Open Set Domain Adaptation: Theoretical Bound and Algorithm

Zhen Fang, Jie Lu, *Fellow, IEEE*, Feng Liu, *Graduate Student Member, IEEE*, Junyu Xuan, *Member, IEEE*, and Guangquan Zhang

***Abstract*—The aim of unsupervised domain adaptation is to leverage the knowledge in a labeled (source) domain to improve a model's learning performance with an unlabeled (target) domain—the basic strategy being to mitigate the effects of discrepancies between the two distributions. Most existing algorithms can only handle unsupervised closed set domain adaptation (UCSDA), i.e., where the source and target domains are assumed to share the same label set. In this article, we target a more challenging but realistic setting: unsupervised open set domain adaptation (UOSDA), where the target domain has unknown classes that are not found in the source domain. This is the first study to provide learning bound for open set domain adaptation, which we do by theoretically investigating the risk of the target classifier on unknown classes. The proposed learning bound has a special term, namely, open set difference, which reflects the risk of the target classifier on unknown classes. Furthermore, we present a novel and theoretically guided unsupervised algorithm for open set domain adaptation, called *distribution alignment with open difference* (DAOD), which is based on regularizing this open set difference bound. The experiments on several benchmark data sets show the superior performance of the proposed UOSDA method compared with the state-of-the-art methods in the literature.**

***Index Terms*—Domain adaptation, machine learning, open set recognition, transfer learning.**

I. Introduction

STANDARD supervised learning relies on the assumption that both the training and the testing samples are drawn from the same distribution. Unfortunately, this assumption does not hold in many applications since the process of collecting samples is prone to data set bias [1], [2]. In object recognition, for example, there can be a discrepancy in the distributions between training and testing as a result of the given conditions, the device type, the position, orientation, and so on. To address this problem, *unsupervised domain adaptation* (UDA) [3], [4] has been proposed as a way of transferring relevant knowledge from a source domain that has an abundance of labeled samples to an unlabeled domain (the target domain).

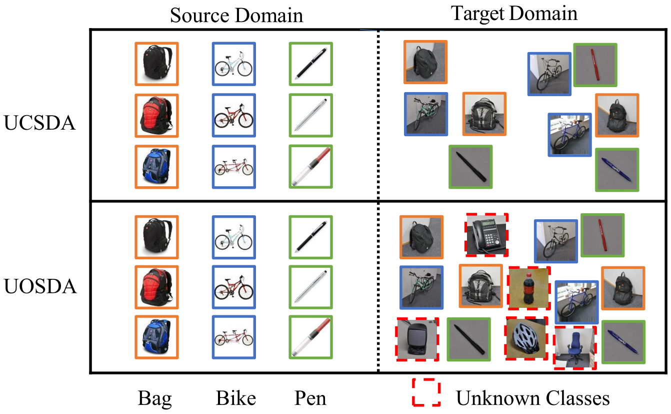

Fig. 1. UOSDA problem, where the target domain contains "unknown" classes that are not contained in the label set of the source domain.

The aim of UDA is to minimize the discrepancy between the distributions of two domains. Existing work on UDA falls into two main categories: 1) feature matching, which seeks a new feature space where the marginal distributions or conditional distributions from the two domains are similar [5]–[7] and 2) instance reweighting, which estimates the weights of the source domain so that the distributional discrepancy is minimized [8], [9]. There is an implicit *assumption* in most existing UDA algorithms [10]–[16] that the source and target domains share the same label set. Under this assumption, UDA is also regarded as *unsupervised closed set domain adaptation* (UCSDA) [17] (see Fig. 1).

However, this assumption in UCSDA algorithms is not realistic in an unsupervised setting (i.e., there are no labels in the target domain) since it is not known whether the classes of target samples are from the label set of the source domain. It may be that the target domain contains additional classes (*unknown classes*) that do not exist in the label set of the source domain [18]. For example, in the Syn2Real task [19], there may be more classes for the real-world objects in the target domain than the synthetic objects contained in the source domain. Therefore, if existing UCSDA algorithms were to be used to solve the UDA problem without the assumption in UCSDA, the potential mismatches between unknown and

Manuscript received June 14, 2019; revised December 4, 2019, March 17, 2020, June 23, 2020, and August 12, 2020; accepted August 13, 2020. This work was supported by the Australian Research Council (ARC) under Grant DP170101632 and Grant FL190100149. *(Corresponding author: Jie Lu.)*

The authors are with the Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: zhen.fang@student.uts.edu.au; jie.lu@uts.edu.au; feng.liu@uts.edu.au; junyu.xuan@uts.edu.au; guangquan.zhang@uts.edu.au).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3017213

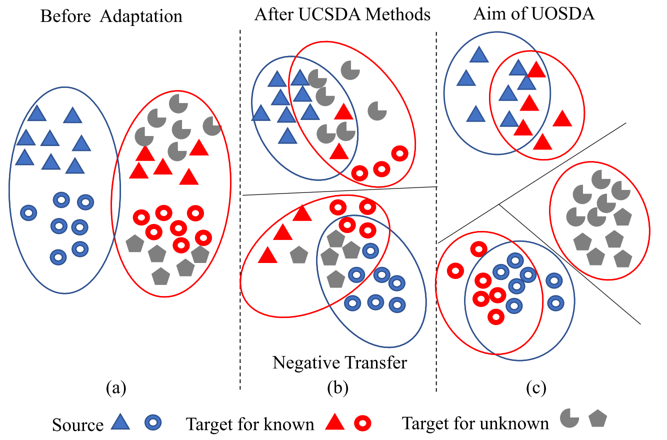

Fig. 2. Aim of UOSDA. (a) Original source and target samples are given. (b) UCSDA algorithm matches the source and target samples, leading to negative transfer. Because the unknown target samples interfere with distribution matching. (c) UOSDA algorithm classifies known target samples into the correct known classes and recognizes the unknown target samples as unknown.

known classes would likely result in negative transfer [20] [see Fig. 2(b)].

To address the UDA problem *without* the assumption, Busto and Gall [17] and Saito *et al.* [18] recently proposed a new problem setting, *unsupervised open set domain adaptation* (UOSDA), in which the unlabeled target domain contains unknown classes that do not belong to the label set of the source domain (see Fig. 1). There are two key challenges [18] in addressing the UOSDA problem. The first challenge is that there is not enough knowledge in the target domain to classify the unknown samples. So how these samples should be labeled? The solution is to mine deeper information in the target domain to delineate a boundary between the known and unknown classes. The second challenge in UOSDA is the difference in distributions. The unknown target samples should not be matched when the overall distribution is matched; otherwise, the negative transfer may occur.

Only a small number of algorithms have been proposed to solve the UOSDA problem [17], [18], [21]–[23]. The first proposed UOSDA algorithm is *assign-and-transform-iteratively* (ATI) [17], which recognizes unknown target samples using constraint integer programming. It then learns a linear map to match the source domain with the target domain by excluding the predicted unknown target samples. However, ATI carries the assumption that the source domain contains unknown classes that are not in the target domain. Hence, the first proposed deep UOSDA algorithm, *open set back propagation* (OSBP) [18] was developed to address the UOSDA problem without this assumption. It rejects unknown target samples by training a binary cross entropy loss.

Although ATI and OSBP are designed to solve the UOSDA problem, neither is based on a theoretical analysis of UOSDA. Moreover, no work has yet given learning bound for open set domain adaptation problems. To fill this gap, this article presents a theoretical exploration of UOSDA. In studies, the risk of the target classifier on unknown classes, we discovered the risk is closely related to a special term called *open set difference* which can be estimated from the unlabeled samples. Minimizing the open set difference helps us to classify unknown target samples, addressing the first challenge.

Following our theory, we design a principle-guided UOSDA algorithm referred to as *distribution alignment with open difference* (DAOD). This algorithm can accurately classify unknown target samples while minimizing the discrepancy between the two domains for known classes. DAOD learns the target classifier by simultaneously optimizing the structural risk function [24], the joint distribution alignment, the manifold regularization [25], and open set difference. The reason DAOD is able to avoid negative transfer lies in its ability to minimize the open set difference, which enables the unknown target samples to be classified accurately as unknown. By excluding these recognized unknown target samples, the source and target domains can be precisely aligned, addressing the second challenge.

As mentioned, there is no theoretical work in the literature for open set domain adaptation. The closest theoretical work is by Ben-David *et al.* [26], who gives VC-dimension-based generalization bounds. Unfortunately, this work has several restrictions: 1) the theoretical analysis only covers closed settings and 2) the work only solves binary classification tasks, rather than the multiclass problems common to open settings. A significant contribution of this article is that the theoretical work gives learning bound for open set domain adaptation.

The contributions of this article are summarized as follows.

1) We provide the theoretical bound for open set domain adaptation. The closed set domain adaptation theory [26] is a special case of our theoretical results. To the best of our knowledge, this is the first work on open set domain adaptation theory.
2) We develop an unsupervised novel open set domain adaptation algorithm, DAOD, which is based on the open set learning bound proposed. The algorithm enables the unknown target samples to be separated from samples using an open set difference.
3) We conduct 38 real-world UOSDA tasks (including 20 face recognition tasks and 18 object recognition tasks) for evaluating DAOD and existing UOSDA algorithms. Extensive experiments demonstrate that DAOD outperforms the state-of-the-art UOSDA algorithms ATI and OSBP.

This article is organized as follows. Section II reviews existing work on UCSDA, open set recognition, and UOSDA. Section III presents the definitions, important notations, and our problem. Section IV provides the main theoretical result and our proposed algorithm. Comprehensive evaluation results and analyses are provided in Section V. Finally, Section VI concludes this article.

II. Related Work

In this section, we present relevant work related to UCSDA algorithms, open set recognition, and UOSDA.

A. Closed Set Domain Adaptation

Ben-David *et al.* [26] proposed learning bounds for closed set domain adaptation, where the bounds show that the performance of the target classifier depends on the performance of the source classifier and the discrepancy between the source and target domains. Many UCSDA algorithms [7], [11], [27], [28] have been proposed based on theoretical bounds with the objective of minimizing the discrepancy between domains. These algorithms can be roughly divided into two categories: feature matching and instance reweighting.

Feature matching aims to reduce the distribution discrepancy by learning a new feature representation. *transfer component analysis* (TCA) [5] learns a new feature space to match distributions by employing the *maximum mean discrepancy* (MMD) [29]. *Joint distribution adaptation* (JDA) [6] improves TCA by jointly matching marginal distributions and conditional distributions. *Adaptation Regularization Transfer Learning* (ARTL) [30] considers a manifold regularization term [25] to learn the geometric relations between domains, while matching distributions. *joint geometrical and statistical alignment* (JGSA) [31] not only considers the distribution discrepancy but also matches the geometric shift. Recent advances show that deep networks can be successfully applied to closed set domain adaptation tasks. *Deep adaptation networks* [32] considers three adaptation layers for matching distributions and applies multiple kernels MMD [33] for adapting deep representations. *Wasserstein distance guided representation learning* [34] minimizes the distribution discrepancy by employing *Wasserstein distance* in neural networks.

In the other category, instance reweighting algorithms reduce the distribution discrepancy by weighting samples in the source domain. *Kernel mean matching* [8] defines the weights as the density ratio between the source domain and the target domain. Yu and Szepesvári [9] have provided a theoretical analysis for important instance reweighting algorithms. However, with a very great domain discrepancy, a large number of effective source samples are down weighted and useful information is lost.

Unfortunately, the algorithms mentioned above cannot be applied to open set domain adaptation because unknown target samples would be included in the distribution matching process, leading to negative transfer.

B. Open Set Recognition

When the source domain and target domain for known classes share the same distribution, open set domain adaptation becomes *open set recognition*. A common method for handling open set recognition relies on the use of threshold-based classification strategies [35]. Establishing a threshold for the similarity score means distant samples are removed from the training samples. *Open set nearest neighbor* (OSNN) [36] recognizes whether a sample is from an unknown class by comparing the threshold with a ratio: the similarity score of the sample to the two classes most similar to that sample. Another research stream relies on modifying *support vector machines* (SVMs) [37]–[39]. *Multiclass open set SVM* [39] uses a multiclass SVM as a basis to learn the unnormalized posterior probability, which is used to reject unknown samples.

TABLE I

INTRODUCTION OF DATA SETS

Dataset	Type	#Sample	#Feature	#Class	Domain
Office-31	Object	4,110	4,096	31	A,W,D
Office-Home	Object	15,500	2,048	65	Ar,Cl,Pr,Rw
PIE	Face	1,1554	1,024	68	P1,...,P5

C. Open Set Domain Adaptation

The open set domain adaptation problem was proposed by ATI [17]. Using ℓ_2 distance between each target sample and the center of each source class, ATI constructs a constraint integer programming to recognize unknown target samples S_u, then learns a linear transformation to match the source domain and target domain excluding S_u. However, ATI requires the help of unknown source samples, which are unavailable in our setting. Recently, a deep learning algorithm, OSBP [18], is a recent contribution to addressing UOSDA. OSBP relies on an adversarial neural network and a binary cross entropy loss to learn the probability of the target samples. It then uses the estimated probability to separate samples of known and unknown classes in the target. However, we have not found any article that considers the learning bound for open set domain adaptation. In this article, we aim to fill in the blanks of the open set domain adaptation theory.

III. PRELIMINARIES

In this section, we formally define the problem set for this article and introduce some fundamental concepts to domain adaptation and, therefore, this study. The notations used throughout this article are summarized in Table I in the Supplementary Material.

A. Definitions and Problem Setting

Important definitions are presented as follows.

Definition 1 (Domain): Given a feature (input) space $\mathcal{X} \subset \mathbb{R}^d$ and a label (output) space $\mathcal{Y}$, a *domain* is a joint distribution $P(X, Y)$, where random variables $X \in \mathcal{X}$, $Y \in \mathcal{Y}$.

To be exact, a random variable is a measurable map. In Definition 1, $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ mean that the image sets of X and Y are contained in the spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. We normally name the random variable X from the feature space $\mathcal{X}$ as feature vector while the random variable Y as a label. The label Y can either be continuous (in a regression task) or discrete (in a classification task). In this article, we have fixed it as a discrete variable with a fixed number of items. Based on this definition, we have in the following.

Definition 2 (Domains for Open Set Domain Adaptation): Given a feature space $\mathcal{X} \subset \mathbb{R}^d$ and the label spaces $\mathcal{Y}^s, \mathcal{Y}^t$, the source and target domains have different joint distributions $P(X^s, Y^s)$ and $P(X^t, Y^t)$, where the label space $\mathcal{Y}^s \subset \mathcal{Y}^t$, and random variables $X^s, X^t \in \mathcal{X}$, $Y^s \in \mathcal{Y}^s$, $Y^t \in \mathcal{Y}^t$.

From Definitions 1 and 2, we can see that: 1) X^s and X^t are from the same space because our focus is on homogeneous situations and 2) $\mathcal{Y}^s$ is a subset of $\mathcal{Y}^t$. The classes from $\mathcal{Y}^t \backslash \mathcal{Y}^s$ are the *unknown target classes*. The classes from $\mathcal{Y}^s$ are the *known classes*. Thus, the problem to be solved as follows.

Problem 3 (Unsupervised Open Set Domain Adaptation): Given labeled samples $\mathcal{S}$ drawn from the source domain $P(X^s, Y^s)$ i.i.d. and unlabeled samples $\mathcal{T}_X$ drawn from the target marginal distribution $P(X^t)$ i.i.d., the aim of UOSDA is to find a target classifier $f^t : \mathcal{X} \rightarrow \mathcal{Y}^t$ such that

1) f^t classifies known target samples into the correct known classes.
2) f^t classifies unknown target samples as unknown.

It is worth noting that, with UOSDA tasks, the algorithm only needs to classify unknown samples as unknown and classify known target samples into the correct known classes. Classifying unknown target samples into correct unknown classes is not necessary. Hence, we consider all unknown target samples are allocated to one big unknown class. Without loss of generality, we assume that $\mathcal{Y}^s = \{\mathbf{y}_c\}_{c=1}^{C}$, $\mathcal{Y}^t = \{\mathbf{y}_c\}_{c=1}^{C+1}$, where the label $\mathbf{y}_{C+1}$ represents the unknown target classes and the label $\mathbf{y}_c \in \mathbb{R}^{(C+1)\times 1}$ is a one-hot vector, whose cth coordinate is 1 and other coordinates are 0. The label $\mathbf{y}_c$ represents the cth class.

B. Concepts and Notations

Before introducing our main results, we need to introduce the following necessary concepts and notations in this field. Unless otherwise specified, all the following notations are used consistently throughout this article without further explanations. More detail on these notations is provided in Appendix A in the Supplementary Material.

1) Notations for Distributions: For the sake of simplicity, we use the notations $P_{X^sY^s}$ and $P_{X^tY^t}$ to denote the joint distributions $P(X^s, Y^s)$ and $P(X^t, Y^t)$, respectively, and also denote P_{X^s} and P_{X^t} as the marginal distributions $P(X^s)$ and $P(X^t)$, respectively.

$P_{X^s|\mathbf{y}_c}$ and $P_{X^t|\mathbf{y}_c}$ represent the conditional distributions for the cth class $P(X^s|Y^s = \mathbf{y}_c)$ and $P(X^t|Y^t = \mathbf{y}_c)$, while π_c^t represents the target class-prior probability for cth class $P(Y^t = \mathbf{y}_c)$. Hence, $\pi_{C+1}^t = P(Y^t = \mathbf{y}_{C+1})$ is the class-prior probability for the unknown target classes.

Lastly, $P_{X^t|\mathcal{Y}^s}$ represents the target conditional distribution for the known classes $P(X^t|Y^t \in \mathcal{Y}^s)$, which can be evaluated by

$$\frac{P(X^t, Y^t \in \mathcal{Y}_s)}{P(Y^t \in \mathcal{Y}_s)} = \frac{\sum_{c=1}^{C} P(X^t|Y^t = \mathbf{y}_c)\pi_c^t}{1 - \pi_{C+1}^t}.$$

The notation $\widehat{P}$ denotes the corresponding empirical distribution to any distribution P. For example, $\widehat{P}_{X^sY^s}$ represents the empirical distribution corresponding to $P_{X^sY^s}$.

2) Risks and Partial Risks: Risks and partial risks are two important concepts in learning theory, which are briefly explained in the following and later used in our theorems.

Following the notations in [40], we consider a multiclass classification task with a *hypothesis space* $\mathcal{H}$ of the *scoring functions*

$$\begin{aligned} \boldsymbol{h} :\ & \mathcal{X} \rightarrow \mathbb{R}^{|\mathcal{Y}^t|} = \mathbb{R}^{C+1} \\ & \mathbf{x} \rightarrow [h_1(\mathbf{x}), \ldots, h_{C+1}(\mathbf{x})]^T \end{aligned} \tag{1}$$

where the output $h_c(\mathbf{x})$ indicates the confidence in the prediction of the label $\mathbf{y}_c$. Let $\ell : \mathbb{R}^{C+1} \times \mathbb{R}^{C+1} \rightarrow \mathbb{R}_+$ be a *symmetric loss function*. Then, the *risks* of $\boldsymbol{h} \in \mathcal{H}$ w.r.t. ℓ under $P_{X^sY^s}$ and $P_{X^tY^t}$ are given by

$$\begin{aligned} R^s(\boldsymbol{h}) &:= \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim P_{X^sY^s}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) = \mathbb{E}\, \ell(\boldsymbol{h}(X^s), Y^s)) \\ R^t(\boldsymbol{h}) &:= \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim P_{X^tY^t}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) = \mathbb{E}\, \ell(\boldsymbol{h}(X^t), Y^t)). \end{aligned} \tag{2}$$

The *partial risk* of $\boldsymbol{h} \in \mathcal{H}$ for the known target classes is

$$R_*^t(\boldsymbol{h}) := \frac{1}{1 - \pi_{C+1}^t} \int_{\mathcal{X}\times\mathcal{Y}^s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X^tY^t}(\mathbf{x}, \mathbf{y}) \tag{3}$$

and the *partial risk* of $\boldsymbol{h} \in \mathcal{H}$ for the unknown target classes is

$$\begin{aligned} R_{C+1}^t(\boldsymbol{h}) &:= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X^t|\mathbf{y}_{C+1}}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{C+1}) \\ &= \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{C+1}) \mathrm{d}P_{X^t|\mathbf{y}_{C+1}}(\mathbf{x}). \end{aligned} \tag{4}$$

According to (2)–(4), we have

$$R^t(\boldsymbol{h}) = \pi_{C+1}^t R_{C+1}^t(\boldsymbol{h}) + (1 - \pi_{C+1}^t) R_*^t(\boldsymbol{h}). \tag{5}$$

The proof can be found in Appendix A in the Supplementary Material.

Finally, we denote

$$\begin{aligned} R_{u,C+1}^s(\boldsymbol{h}) &:= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X^s}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{C+1}) = \mathbb{E}\, \ell(\boldsymbol{h}(X^s), \mathbf{y}_{C+1}) \\ R_{u,C+1}^t(\boldsymbol{h}) &:= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X^t}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{C+1}) = \mathbb{E}\, \ell(\boldsymbol{h}(X^t), \mathbf{y}_{C+1}) \end{aligned} \tag{6}$$

as the *risks* that the samples are regarded as the unknown classes.

Given a risk $R(\boldsymbol{h})$, it is convenient to use notation $\widehat{R}(\boldsymbol{h})$ as the empirical risk that corresponds to $R(\boldsymbol{h})$. Hence, notations $\widehat{R}^s(\boldsymbol{h})$, $\widehat{R}_{u,C+1}^s(\boldsymbol{h})$, and $\widehat{R}_{u,C+1}^t$ represent the empirical risks corresponding to the risks $R^s(\boldsymbol{h})$, $R_{u,C+1}^s(\boldsymbol{h})$, and $R_{u,C+1}^t(\boldsymbol{h})$ respectively.

3) Discrepancy Distance and Maximum Mean Discrepancy: One challenge of domain adaptation is the mismatch between the distributions of the source and target domains. To mitigate this effect, two famous distribution distances have been proposed as the measures of the distribution difference.

The first one is discrepancy distance presented as follows.

Definition 4 (Discrepancy Distance [41]): Let the hypothesis space $\mathcal{H}$ be a set of functions defined in a feature space $\mathcal{X}$, ℓ be a loss function and P_1, P_2 be distributions on space $\mathcal{X}$. The discrepancy distance $d_{\mathcal{H}}^{\ell}(P_1, P_2)$ between the distributions P_1 and P_2 over $\mathcal{X}$ is

$$\sup_{\boldsymbol{h},\boldsymbol{h}^*\in\mathcal{H}} |\mathop{\mathbb{E}}_{\mathbf{x}\sim P_1} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x})) - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x}))|.$$

If ℓ in the definition is the zero-one loss, the discrepancy distance is known as the $\mathcal{H}\Delta\mathcal{H}$ distance [26]. The discrepancy distance is symmetric and satisfies the triangle inequality, but it does not define a distance in general: $d_{\mathcal{H}}^{\ell}(P_1, P_2) = 0$ does not mean $P_1 = P_2$.

The second distance is MMD.

Definition 5 (Maximum Mean Discrepancy [29]): Given a feature space $\mathcal{X}$ and a class of function $\mathcal{F}$ $(f : \mathcal{X} \rightarrow \mathbb{R})$. The MMD between the distributions P_1 and P_2 is

$$\mathrm{MMD}[\mathcal{F}, P_1, P_2] := \sup_{f\in\mathcal{F}} |\mathop{\mathbb{E}}_{\mathbf{x}\sim P_1} f(\mathbf{x}) - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2} f(\mathbf{x})|.$$

To ensure that MMD is a metric, one must identify a function class $\mathcal{F}$ that is rich enough to uniquely identify whether $P_1 = P_2$. Gretton *et al.* [29], therefore, propose as MMD function class $\mathcal{F}$ the unit ball in a *reproducing kernel Hilbert space* (RKHS) $\mathcal{H}_k$ [42] (the subscript k represents the reproducing kernel and is used to distinguish the hypothesis set $\mathcal{H}$ from the RKHS $\mathcal{H}_k$).

For convenience, we have used the notation $\text{MMD}_{\mathcal{H}_k}(\cdot,\cdot)$ to represent $\text{MMD}[\mathcal{F},\cdot,\cdot]$, when $\mathcal{F} = \{f \in \mathcal{H}_k : \|f\|_{\mathcal{H}_k} \le 1\}$ [29]. Note that $\text{MMD}_{\mathcal{H}_k}$ is symmetric and satisfies the triangle inequality. When the kernel k is a *universal kernel*, $\text{MMD}_{\mathcal{H}_k}(P_1, P_2) = 0$ if and only if $P_1 = P_2$, which implies that $\text{MMD}_{\mathcal{H}_k}$ is a metric.

Though the MMD distance is powerful, it is not convenient to be optimized as a regularization term in shallow domain adaptation algorithms. The *projected MMD* [5], [42], [43] has been proposed to transform the MMD distance into a proper regularization term. Given a scoring function $\boldsymbol{h} = [h_1, \ldots, h_{C+1}]^T$, where $h_c \in \mathcal{H}_k, c = 1, \ldots, C+1$, the projected MMD is defined as follows:

$$D_{\boldsymbol{h},k}(P_1, P_2) = \left\| \int_{\mathcal{X}} \boldsymbol{h}(\mathbf{x})\mathrm{d}P_1(\mathbf{x}) - \int_{\mathcal{X}} \boldsymbol{h}(\mathbf{x})\mathrm{d}P_2(\mathbf{x}) \right\|_2$$

where $\|\cdot\|_2$ is the ℓ_2 norm.

4) Manifold Regularization: The idea of manifold regularization has a rich machine learning history going back to transductive learning and truly semi-supervised learning [25]. Manifold regularization is specifically designed to control the complexity as measured by the geometry of the distribution. Given samples $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, the manifold regularization is

$$\sum_{i,j=1}^{n} \|\boldsymbol{h}(\mathbf{x}_i) - \boldsymbol{h}(\mathbf{x}_j)\|_2^2 \mathbf{W}_{ij}$$

where $[\mathbf{W}_{ij}]$ is the pairwise affinity matrix and $\mathbf{W}_{ij}$ estimates the similarity of $\mathbf{x}_i, \mathbf{x}_j$.

By the manifold assumption [25], if two samples from the support set of the distributions P_{X^s}, P_{X^t} are close, then the scores of the two samples are similar. To extract geometric relationship between domains, the manifold regularization has been used by many closed set domain adaptation algorithms [30], [43]–[48].

IV. Proposed Algorithm

A. Main Theoretical Result and Open Set Difference

Before introducing the main theorem, we first define open set difference, one of the main contributions of this article.

Definition 6 (Open Set Difference): Given risks $R^s_{u,C+1}(\boldsymbol{h})$ and $R^t_{u,C+1}(\boldsymbol{h})$ defined in (6), the open set difference is

$$\Delta_o = \frac{R^t_{u,C+1}(\boldsymbol{h})}{1-\pi^t_{C+1}} - R^s_{u,C+1}(\boldsymbol{h})$$

where π^t_{C+1} is the class-prior probability for the unknown target classes.

The following theorem provides an open set domain adaptation bound according to discrepancy distance and open set difference.

Theorem 7: Given a symmetric loss function ℓ satisfying triangle inequality and a hypothesis $\mathcal{H}$ with a mild condition that the constant vector value function $\boldsymbol{g} := \mathbf{y}_{C+1} \in \mathcal{H}$, then for any $\boldsymbol{h} \in \mathcal{H}$, we have

$$\frac{R^t(\boldsymbol{h})}{1-\pi^t_{C+1}} \le \overbrace{R^s(\boldsymbol{h})}^{\text{Source Risk}} + 2\overbrace{d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s})}^{\text{Distribution Discrepancy}} + \Lambda + \underbrace{\frac{R^t_{u,C+1}(\boldsymbol{h})}{1-\pi^t_{C+1}} - R^s_{u,C+1}(\boldsymbol{h})}_{\text{Open Set Difference}\,\Delta_o}$$

where $R^s(\boldsymbol{h})$ and $R^t(\boldsymbol{h})$ are the risks defined in (2), $R^s_{u,C+1}(\boldsymbol{h})$, and $R^t_{u,C+1}(\boldsymbol{h})$ are the risks defined in (6), $R^t_*(\boldsymbol{h})$ is the partial risk defined in (3), and $\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} R^s(\boldsymbol{h}) + R^t_*(\boldsymbol{h})$.

Proof: Here, we provide a proof sketch. Detailed proof is given in Appendix A in the Supplementary Material. According to (5), we have

$$\frac{R^t(\boldsymbol{h})}{1-\pi^t_{C+1}} - R^s(\boldsymbol{h}) = R^t_*(\boldsymbol{h}) - R^s(\boldsymbol{h}) + \frac{\pi^t_{C+1}}{1-\pi^t_{C+1}} R^t_{C+1}(\boldsymbol{h}). \tag{7}$$

Then, we can check that

$$R^t_*(\boldsymbol{h}) - R^s(\boldsymbol{h}) \le \Lambda + d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}) \tag{8}$$

$$\frac{\pi^t_{C+1}}{1-\pi^t_{C+1}} R^t_{C+1}(\boldsymbol{h}) \le d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}) + \Delta_o. \tag{9}$$

Combining (8) and (9) with (7), we have

$$\frac{R^t(\boldsymbol{h})}{1-\pi^t_{C+1}} \le R^s(\boldsymbol{h}) + 2d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}) + \Lambda + \Delta_o.$$

□

Remark 8: The condition $\mathbf{y}_{C+1} \in \mathcal{H}$ can be replaced by a weaker condition that there exists a sequence $\{\boldsymbol{h}_i\}_{i=1}^{+\infty}$ such that $\boldsymbol{h}_i$ converges uniformly to $\mathbf{y}_{C+1}$. Note that the hypothesis space $\mathcal{H}$ used in our algorithm satisfies the weaker condition automatically, and thus, the condition $\mathbf{y}_{C+1} \in \mathcal{H}$ can be removed when we use the $\mathcal{H}$ applied in our algorithm.

The open set difference Δ_o is the crucial term to bound the risk of $\boldsymbol{h}$ on unknown target classes, since

$$R^t_{C+1}(\boldsymbol{h}) \le \frac{1-\pi^t_{C+1}}{\pi^t_{C+1}}(\Delta_o + d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s})). \tag{10}$$

The risk of $\boldsymbol{h}$ on unknown target classes is intimately linked to the open set difference Δ_o

$$\left|\pi^t_{C+1} R^t_{C+1}(\boldsymbol{h}) - (1-\pi^t_{C+1})\Delta_o\right| \le d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}).$$

When $\pi^t_{C+1} = 0$, Theorem 7 degenerates into the closed set scenario with this theoretical bound [26]

$$R^t(\boldsymbol{h}) \le R^s(\boldsymbol{h}) + 3d^\ell_{\mathcal{H}}(P_{X^t}, P_{X^s}) + \Lambda.$$

This is because when $\pi^t_{C+1} = 0$, the open set difference is

$$\Delta_o \le d^\ell_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}) = d^\ell_{\mathcal{H}}(P_{X^t}, P_{X^s}).$$

The significance of Theorem 7 is twofold. First, it highlights that the open set difference Δ_o is the main term for controlling performance in open set domain adaptation. Second, the bound shows a direct connection with closed set domain adaptation theory.

In addition, the open set difference Δ_o consists of two parts: a positive term $R^t_{u,C+1}(\boldsymbol{h})$ and a negative term $R^s_{u,C+1}(\boldsymbol{h})$. A larger positive term implies more target samples are classified as unknown samples. The negative term is used to prevent the source samples from being classified as unknown. According to (10), the negative term and the distance discrepancy jointly prevent all target samples from being recognized as unknown classes. In addition, Corollary 9 also tells us that the positive term and the negative term can be estimated from unlabeled samples. Using the *Natarajan dimension theory* [49] to bound the source risk $R^s(\boldsymbol{h})$, risks $R^t_{u,C+1}(\boldsymbol{h})$, and $R^s_{u,C+1}(\boldsymbol{h})$ by empirical estimates $\widehat{R}^s(\boldsymbol{h})$, $\widehat{R}^t_{u,C+1}(\boldsymbol{h})$, and $\widehat{R}^s_{u,C+1}(\boldsymbol{h})$ respectively, we have the following result.

Corollary 9: Given a symmetric loss function ℓ satisfying the triangle inequality and bounded by B, and a hypothesis $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathcal{Y}^t\}$ with conditions: 1) the constant vector value function $\boldsymbol{g} := \mathbf{y}_{C+1} \in \mathcal{H}$ and 2) the Natarajan dimension of $\mathcal{H}$ is d, if a random labeled sample of size n^s is generated by source joint distribution $P_{X^s Y^s}$-i.i.d. and a random unlabeled sample of size n^t is generated by target marginal distribution P_{X^t}-i.i.d., then for any $\boldsymbol{h} \in \mathcal{H}$ and $\delta \in (0, 1)$ with probability at least $1 - 3\delta$, we have

$$\begin{aligned}\frac{R^t(\boldsymbol{h})}{1-\pi^t_{C+1}} \le{}& \widehat{R}^s(\boldsymbol{h}) + 2d^{\ell}_{\mathcal{H}}(P_{X^t|\mathcal{Y}^s}, P_{X^s}) + \widehat{\Delta}_o + \Lambda\\ &+ 4B\sqrt{\frac{8d\log n^s + 16d\log(C+1) + 2\log 2/\delta}{n^s}}\\ &+ 2B\sqrt{\frac{8d\log n^t + 16d\log(C+1) + 2\log 2/\delta}{\left(1-\pi^t_{C+1}\right)^2 n^t}}\end{aligned}$$

where $\mathcal{X}$ is the feature space, $\mathcal{Y}^t$ is the target label space, $\widehat{R}^s(\boldsymbol{h})$ is the empirical source risk, $R^t(\boldsymbol{h})$ is the target risk, $\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} R^s(\boldsymbol{h}) + R^t_*(\boldsymbol{h})$, and empirical open set difference $\widehat{\Delta}_o = (\widehat{R}^t_{u,C+1}(\boldsymbol{h})/1 - \pi^t_{C+1}) - \widehat{R}^s_{u,C+1}(\boldsymbol{h})$, here $R^s(\boldsymbol{h})$ are the risks defined in (2), $\widehat{R}^s_{u,C+1}(\boldsymbol{h})$ and $\widehat{R}^t_{u,C+1}(\boldsymbol{h})$ are the empirical risks corresponding to $R^s_{u,C+1}(\boldsymbol{h})$ and $R^t_{u,C+1}(\boldsymbol{h})$ defined in (6).

Proof: The proof is given in Appendix D in the Supplementary Material. □

Next, we employ the open set difference Δ_o to construct our model—DAOD.

B. Algorithm

The importance of Theorem 7 is that it tells us the relationships between the three terms (i.e., the source risk, the distribution discrepancy, and the open set difference) and the bound of the open set domain adaptation. Inspired by these relationships, our initial focus is the following optimization problem for UOSDA:

$$\begin{aligned}\boldsymbol{h}^* = \mathop{\arg\min}_{\boldsymbol{h}\in\mathcal{H}}\ & \widehat{R}^s(\boldsymbol{h}) + \lambda d^{\ell}_{\mathcal{H}}(\widehat{P}_{X^s}, \widehat{P}_{X^t|\mathcal{Y}^s})\\ &+ \gamma\left(\frac{1}{1-\pi^t_{C+1}}\widehat{R}^t_{u,C+1}(\boldsymbol{h}) - \widehat{R}^s_{u,C+1}(\boldsymbol{h})\right) \qquad (11)\end{aligned}$$

where the hypothesis space $\mathcal{H}$ is defined as a subset of functional space $\{\boldsymbol{h} = [h_1, \ldots, h_{C+1}]^T : h_c \in \mathcal{H}_k\}$ and λ and γ are two free hyperparameters.

As proven by JDA [6], ARTL [30] and Manifold Embedded Distribution Alignment (MEDA) [44], incorporating conditional distributions into the original marginal distribution discrepancy can lead to superior domain adaptation performance. Hence, we have also added an additional conditional distribution discrepancy to the optimization problem in (11). Hence, the new problem becomes

$$\begin{aligned}\boldsymbol{h}^* = \mathop{\arg\min}_{\boldsymbol{h}\in\mathcal{H}}\ & \widehat{R}^s(\boldsymbol{h}) + \lambda\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s}, \widehat{P}_{X^t|\mathcal{Y}^s})\\ &+ \lambda(1-\mu)\sum_{c=1}^{C} D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s|\mathbf{y}_c}, \widehat{P}_{X^t|\mathbf{y}_c})\\ &+ \gamma\left(\frac{1}{1-\pi^t_{C+1}}\widehat{R}^t_{u,C+1}(\boldsymbol{h}) - \widehat{R}^s_{u,C+1}(\boldsymbol{h})\right)\end{aligned}$$

where $\mu \in [0, 1]$ is the adaptive factor [44] to convexly combine the contributions from both the empirical marginal distribution alignment and the empirical conditional distribution alignment. Note that the original $d^{\ell}_{\mathcal{H}}(\cdot, \cdot)$ is replaced with the projected MMD $D_{\boldsymbol{h},k}(\cdot, \cdot)$ in the new problem, because $d^{\ell}_{\mathcal{H}}(\cdot, \cdot)$ is difficult to estimate. This results in a gap with Theorem 1 where the discrepancy distance is used to measure the distribution discrepancy, rather than projected MMD. To mitigate this gap, we also give a similar theoretical bound using MMD distance (see Theorem 4 in Appendix C in the Supplementary Material for details). Specifically, for proving Theorem 4, we need an additional condition that the loss ℓ is *squared loss* $\ell(\mathbf{y}, \mathbf{y}') = \|\mathbf{y} - \mathbf{y}'\|_2^2$. Thus, we use the squared loss to design our algorithm.

Furthermore, we have added the manifold regularization [25] to learn the geometric structure of the source and target domains. With this regularization, our algorithm can consistently achieve good performance when the setting degrades into a closed set domain adaptation (i.e., where there are no unknown classes). This is because state of the art closed set algorithm ARTL [30] is a special case of DAOD when there is no open set difference.

Thus, the optimization problem can be rewritten as follows:

$$\begin{aligned}\boldsymbol{h}^* = \mathop{\arg\min}_{\boldsymbol{h}\in\mathcal{H}}\ & \widehat{R}^s(\boldsymbol{h}) + \lambda\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s}, \widehat{P}_{X^t|\mathcal{Y}^s})\\ &+ \lambda(1-\mu)\sum_{c=1}^{C} D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s|\mathbf{y}_c}, \widehat{P}_{X^t|\mathbf{y}_c})\\ &+ \alpha\widehat{R}^t_{u,C+1}(\boldsymbol{h}) - \gamma\,\widehat{R}^s_{u,C+1}(\boldsymbol{h})\\ &+ \rho M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_X) + \sigma\|\boldsymbol{h}\|_k^2 \qquad (12)\end{aligned}$$

where $\alpha := \gamma/(1 - \pi^t_{C+1})$, ρ and σ are three free hyperparameters, $\mathcal{T}_X$ denotes unlabeled target samples, $\mathcal{S}_X$ denotes source samples without labels, $M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_X)$ is the manifold regularization, and $\|\boldsymbol{h}\|_k^2$ is the squared norm of $\boldsymbol{h}$ in $\mathcal{H}_k$ to avoid overfitting.

Next, we show how to formulate equation (12) using given samples. First, following the representer theorem, if the optimization problem (12) has a minimizer $\boldsymbol{h}^*$, then $\boldsymbol{h}^*$ can be written as

$$\boldsymbol{h}^*(\mathbf{x}) = \sum_{i=1}^{n^s+n^t} \boldsymbol{\beta}_i k(\mathbf{x}_i, \mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{X}$$

where $\mathbf{x}_i \in \mathcal{S}_X \cup \mathcal{T}_X$ and $\boldsymbol{\beta}_i \in \mathbb{R}^{(C+1)\times 1}$ is the parameter. With this form of $\boldsymbol{h}^*$, we explain the computation of terms in (12). The notations used in this section are summarized in Table II in the Supplementary Material.

1) Distribution Alignment: Since there are no labels for the target samples, we cannot directly compute

$$\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s}, \widehat{P}_{X^t|\mathcal{Y}^s}) + (1-\mu)\sum_{c=1}^{C} D^2_{\boldsymbol{h},k}(\widehat{P}_{X^s|\mathbf{y}_c}, \widehat{P}_{X^t|\mathbf{y}_c}). \quad (13)$$

Therefore, the pseudotarget labels are used to help compute $\widehat{P}_{X^t|\mathcal{Y}^s}$ and $\widehat{P}_{X^t|\mathbf{y}_c}$ instead. Given the pseudotarget samples for known classes $\mathcal{T}_{X,K}$, the pseudotarget samples for cth class $\mathcal{T}_{X,c}$ and the source samples for cth class $\mathcal{S}_{X,c}$ we can compute (13) by the representer theorem and kernel trick as follows:

$$\operatorname{tr}(\boldsymbol{\beta}^T \mathbf{KMK}\boldsymbol{\beta}) \quad (14)$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{n^s+n^t}]^T \in \mathbb{R}^{(C+1)\times(n^s+n^t)}$, $\mathbf{K}$ is the $(n^s+n^t)\times(n^s+n^t)$ kernel matrix $[k(\mathbf{x}_i, \mathbf{x}_j)]$, here $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{S}_X \cup \mathcal{T}_X$, and $\mathbf{M} = \mu\mathbf{M}_0 + (1-\mu)\sum_{c=1}^{C}\mathbf{M}_c$ is the MMD matrix:

$$(\mathbf{M}_0)_{ij} = \begin{cases} \dfrac{1}{(n^s)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{S}_X \\ \dfrac{1}{\left(n^t_K\right)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{T}_{X,K} \\ 0, \quad \mathbf{x}_i \text{ or } & \mathbf{x}_j \in \mathcal{T}_X \setminus \mathcal{T}_{X,K} \\ -\dfrac{1}{n^s n^t_K}, & \text{otherwise} \end{cases}$$

$$(\mathbf{M}_c)_{ij} = \begin{cases} \dfrac{1}{\left(n^s_c\right)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{S}_{X,c} \\ \dfrac{1}{\left(n^t_c\right)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{T}_{X,c} \\ -\dfrac{1}{n^s_c n^t_c}, & \mathbf{x}_i \in \mathcal{S}_{X,c}, \mathbf{x}_j \in \mathcal{T}_{X,c} \\ -\dfrac{1}{n^s_c n^t_c}, & \mathbf{x}_j \in \mathcal{S}_{X,c}, \mathbf{x}_i \in \mathcal{T}_{X,c} \\ 0 & \text{otherwise} \end{cases}$$

where $n^s := |\mathcal{S}_X|$, $n^t_K := |\mathcal{T}_{X,K}|$, $n^s_c := |\mathcal{S}_{X,c}|$, and $n^t_c := |\mathcal{T}_{X,c}|$.

2) Manifold Regularization: The pair-wise affinity matrix is denoted as

$$\mathbf{W}_{ij} = \begin{cases} \operatorname{sim}(\mathbf{x}_i, \mathbf{x}_j), & \mathbf{x}_i \in \mathcal{N}_p(\mathbf{x}_j) or \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}_i) \\ 0, & \text{otherwise} \end{cases}$$

where $\operatorname{sim}(\mathbf{x}_i, \mathbf{x}_j)$ is the similarity function, such as cosine similarity, $\mathcal{N}_p(\mathbf{x}_i)$ denotes the set of p-nearest neighbors to point $\mathbf{x}_i$, and p is a free parameter. The manifold regularization can then be evaluated as follows:

$$\begin{aligned} M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_X) &= \sum_{i,j=1}^{n^s+n^t} \|\boldsymbol{h}(\mathbf{x}_i) - \boldsymbol{h}(\mathbf{x}_j)\|_2^2 \mathbf{W}_{ij} \\ &= \sum_{c=1}^{C+1}\sum_{i,j=1}^{n^s+n^t} h_c(\mathbf{x}_i)\mathbf{L}_{ij} h_c(\mathbf{x}_j) \end{aligned}$$

where $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{S}_X \cup \mathcal{T}_X$, $\mathbf{L}$ is the Laplacian matrix, which can be written as $\mathbf{D} - \mathbf{W}$, here $\mathbf{D}$ is a diagonal matrix, and $\mathbf{D}_{ii} = \sum_{j=1}^{n^s+n^t}\mathbf{W}_{ij}$. Using the representer theorem and kernel trick, the manifold regularization $M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_X)$ can be written as

$$\operatorname{tr}(\boldsymbol{\beta}^T \mathbf{KLK}\boldsymbol{\beta}). \quad (15)$$

3) Open Set Loss Function: We use a matrix to rewrite the loss function and open set difference. Let the label matrix be $\mathbf{Y} \in \mathbb{R}^{(C+1)\times(n^s+n^t)}$

$$\mathbf{Y}_{ij} = \begin{cases} 1, & \mathbf{x}_j \in \mathcal{S}_{X,i} \\ 0, & \text{otherwise} \end{cases} \text{when} \quad i \le C \quad (16)$$

$$\mathbf{Y}_{ij} = \begin{cases} 1, & \mathbf{x}_j \in \mathcal{T}_{X,C+1} \\ 0, & \text{otherwise} \end{cases} \text{when} \quad i = C+1. \quad (17)$$

The label matrix $\widetilde{\mathbf{Y}} \in \mathbb{R}^{(C+1)\times(n^s+n^t)}$ is

$$\widetilde{\mathbf{Y}}_{ij} = 1 \text{ iff } i = C+1 \text{ and } \mathbf{x}_j \in \mathcal{S}_X, \text{ otherwise } \widetilde{\mathbf{Y}}_{ij} = 0. \quad (18)$$

Then

$$\begin{aligned} &\widehat{R}^s(\boldsymbol{h}) + \alpha\widehat{R}^t_{u,C+1}(\boldsymbol{h}) - \gamma\widehat{R}^s_{u,C+1}(\boldsymbol{h}) + \sigma\|\boldsymbol{h}\|_k^2 \\ &\quad = \|(\mathbf{Y} - \boldsymbol{\beta}^T\mathbf{K})\mathbf{A}\|_F^2 - \gamma\|(\widetilde{\mathbf{Y}} - \boldsymbol{\beta}^T\mathbf{K})\widetilde{\mathbf{A}}\|_F^2 + \sigma\operatorname{tr}(\boldsymbol{\beta}^T\mathbf{K}\boldsymbol{\beta}) \end{aligned} \quad (19)$$

where $\mathbf{A}$ is a $(n^s+n^t)\times(n^s+n^t)$ diagonal matrix with $\mathbf{A}_{ii} = ((1/n^s))^{1/2}$ if $\mathbf{x}_i \in \mathcal{S}_X$, $\mathbf{A}_{ii} = ((\alpha/n^t))^{1/2}$ if $\mathbf{x}_i \in \mathcal{T}_X$; $\widetilde{\mathbf{A}}$ is a $(n^s+n^t)\times(n^s+n^t)$ diagonal matrix with $\widetilde{\mathbf{A}}_{ii} = ((1/n^s))^{1/2}$ if $\mathbf{x}_i \in \mathcal{S}_X$, $\widetilde{\mathbf{A}}_{ii} = 0$ if $\mathbf{x}_i \in \mathcal{T}_X$, and $\|\cdot\|_F$ is the Frobenius norm.

4) Overall Reformulation: Finally, based on (14), (15), and (19), the optimization problem in (12) is reformulated as

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}\in\mathbb{R}^{(n^s+n^t)\times(C+1)}}{\arg\min}\ \mathcal{L}(\boldsymbol{\beta})$$

where

$$\begin{aligned} \mathcal{L}(\boldsymbol{\beta}) := &\|(\mathbf{Y} - \boldsymbol{\beta}^T\mathbf{K})\mathbf{A}\|_F^2 - \gamma\|(\widetilde{\mathbf{Y}} - \boldsymbol{\beta}^T\mathbf{K})\widetilde{\mathbf{A}}\|_F^2 \\ &+ \operatorname{tr}(\boldsymbol{\beta}^T\mathbf{K}(\lambda\mathbf{M} + \rho\mathbf{L})\mathbf{K}\boldsymbol{\beta}) + \sigma\operatorname{tr}(\boldsymbol{\beta}^T\mathbf{K}\boldsymbol{\beta}). \end{aligned} \quad (20)$$

C. Training

There is a negative term in $\mathcal{L}(\boldsymbol{\beta})$; hence, it may be not correct to compute the optimizer by solving the equation $(\partial\mathcal{L}(\boldsymbol{\beta})/\partial\boldsymbol{\beta}) = \mathbf{0}$ directly. Maybe the "minimizer" solved by $(\partial\mathcal{L}(\boldsymbol{\beta})/\partial\boldsymbol{\beta}) = \mathbf{0}$ is a maximum point or a saddle point. Fortunately, the following theorem shows that there exists a unique optimizer that can be solved by $(\partial\mathcal{L}(\boldsymbol{\beta})/\partial\boldsymbol{\beta}) = \mathbf{0}$.

Theorem 10: If the coefficient γ of $\widehat{R}^s_{u,C+1}(\boldsymbol{h})$ is smaller than 1 and the kernel k is universal, then the $\mathcal{L}(\boldsymbol{\beta})$ defined in (20) has a unique minimizer, which can be written as

$$\left((\mathbf{A}^2 - \gamma\widetilde{\mathbf{A}}^2 + \lambda\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{I}\right)^{-1}(\mathbf{A}^2\mathbf{Y}^T - \gamma\widetilde{\mathbf{A}}^2\widetilde{\mathbf{Y}}^T). \quad (21)$$

The proof can be found in Appendix B in the Supplementary Material.

To compute the true value of (21), it is best to use the ground truth labels of the target domain. However, our focus is on unsupervised task, which means that it is impossible to obtain

```
Algorithm 1: DAOD
Input: Data $\mathcal{S}, \mathcal{T}_X$; #iterations $T$; #neighbor $p$ and
       parameters $\lambda, \sigma, \rho, \alpha, \gamma, \mu$; threshold $t$; universal
       kernel function $k(\cdot, \cdot)$.
1. $\widetilde{Y}_t \leftarrow \mathrm{OSNN}^{\mathrm{cv}}(\mathcal{S}, \mathcal{T}_X, t)$;% Predict pseudo labels;
2. Compute $\mathbf{L}, \mathbf{K}$ using $\mathcal{S}$, $\mathcal{T}_X$, and $\widetilde{Y}_t$;
3. $i \leftarrow 1$;
while $i < T + 1$ do
    4. Compute $\mathbf{M}$ using $\mathcal{S}$, $\mathcal{T}_X$, and $\widetilde{Y}_t$;
    5. Compute $\boldsymbol{\beta}$ by formula (21);
    6. $\widetilde{Y}_t \leftarrow \boldsymbol{\beta}^T \mathbf{K}$;%Predict pseudo labels;
    7. $i \leftarrow i + 1$;
Output: Predicted target labels $\widetilde{Y}_t$, classifier $\boldsymbol{\beta}^T \mathbf{K}$.
```

any true target labels and, as mentioned, pseudolabels can be used instead. These pseudolabels are generated by applying an open set classifier that has been trained on the source samples to the target samples.

In this article, we used *OSNN for class verification-t* ($\mathrm{OSNN}^{\mathrm{cv}}$-*t*) [36] to help us learn the pseudolabels. We select the two nearest neighbors $\mathbf{v}, \mathbf{u}$ from the test sample $\mathbf{s}$. If both nearest neighbors have the same label $\mathbf{y}_c$, $\mathbf{s}$ is classified with the label $\mathbf{y}_c$. Otherwise, the following ratio is calculated $\|\mathbf{v}-\mathbf{s}\|_2/\|\mathbf{u}-\mathbf{s}\|_2$, on the assumption that $\|\mathbf{v}-\mathbf{s}\|_2 \leq \|\mathbf{u}-\mathbf{s}\|_2$. If the ratio is smaller than or equal to a predefined threshold t, $0 < t < 1$, $\mathbf{s}$ is classified with the same label as $\mathbf{v}$. Otherwise, $\mathbf{s}$ is recognized as the unknown sample.

To make the pseudolabels more accurate, we use the iterative pseudolabel refinement strategy, proposed by JDA [6]. The implementation details are demonstrated in Algorithm 1 (https://github.com/fang-zhen/Open-set-domain-adaptation).

V. Experiments and Evaluations

In this section, we first utilized real-world data sets to verify the performance of DAOD. We then conducted experiments to examine the behavior of the parameters.

A. Real World Data Sets

We evaluated our algorithm on three cross-domain recognition tasks: object recognition (**Office-31** and **Office-Home**), and face recognition (**PIE**). Table I lists the statistics of these data sets.

Office-31 [50] consists of three real-world object domains: AMAZON (**A**), DSLR (**D**) and WEBCAM (**W**). It has 4 652 images with 31 common categories. This means that there are six domain adaptation tasks: **A** → **D**, **A** → **W**, **D** → **A**, **W** → **A**, **D** → **W**, and **W** → **D**. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from the fully connected layer-7 (fc7) of the AlexNet [51]. We introduced an open set protocol for this data set by taking classes 1–10 as the shared classes in alphabetical order. The classes 21–31 were used as the unknown classes in the target domain.

Office-Home [52] consists of four different domains: Artistic (**Ar**), Clipart (**Cl**), Product (**Pr**), and Real-World (**Rw**). Each domain contains images from 65 object classes. We constructed 12 OSDA tasks: **Ar** → **Cl**, **Ar** → **Pr**,…, **Rw** → **Ar**. In alphabetical order, we used the first 25 classes as the known classes and classes 26–65 as the unknown classes. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from ResNet-50.

PIE [53] contains 41 368 facial images of 68 people in various poses, illuminations, and expression changes. The face images are captured by 13 synchronized cameras (different poses) and 21 flashes (different illuminations and/or expressions). We focused on 5 of 13 poses, i.e., **PIE1** (C05, left pose), **PIE2** (C07, upward pose), **PIE3** (C09, downward pose), **PIE4** (C27, frontal pose), and **PIE5** (C29, right pose). These facial images were cropped to a size of 32 × 32. We took classes 1–20 as the known classes and classes 21–68 as the unknown classes in the target domain. Twenty tasks were tested: **PIE1→PIE2**, **PIE1→PIE3**,…, **PIE5→PIE4**.

B. Baseline Algorithms

The baseline algorithms selected for comparison with DAOD were as follows.

1) *No Transfer:*
   a) OSNN [36]. OSNN recognizes a sample as unknown by computing the ratio of similarity scores to the two most similar classes of the sample and then comparing the ratio with a predefined threshold.
2) *Closed Set:*
   a) TCA [5] + OSNN. The aim of implementing TCA is to show that if the UCSDA algorithm is used to solve the UOSDA problem, the negative transfer will occur, leading to poor performance.
3) *Open Set:*
   a) JDA [6] + OSNN. We extended JDA into the open set setting. Joint distribution matching is the main step in JDA. Thus, we simply matched the known samples predicted by OSNN when the JDA algorithm was implemented.
   b) JGSA [31] + OSNN. We extended JGSA into the open set setting. First, for learning new features, we implemented JGSA using the source samples and the known target samples predicted by OSNN. Then, we used OSNN to predict pseudolabels. We repeated the process until convergence.
   c) ATI [17] + OSNN. ATI was the first UOSDA algorithm, but it requires the unknown source samples to implement. Therefore, to implement ATI under our setting, we used ATI to select the outliers, and then learned the new features for matching the source domain and target domain excluding selected outliers. Finally, OSNN was used to predict the labels.
   d) OSBP [18]. OSBP utilizes adversarial neural networks and a binary cross entropy loss to learn the probability for the target samples and then uses

the estimated probability to recognize the unknown samples.

C. Experimental Setup

Before reporting the detailed evaluation results, it is important to explain how DAOD's hyperparameters are tuned. DAOD has several hyperparameters: 1) the choice of the kernel function k; 2) the adaptation parameters $\lambda, \sigma, \rho, p, \mu$; 3) the open set parameters α, γ; and 4) #iterations T and the threshold $t \in (0, 1)$. Each parameter is discussed one by one next.

1) Kernel Function k: As suggested in [29] and [44], we chose the Gaussian kernel

$$k(\mathbf{a}, \mathbf{b}) = \exp\left(-\frac{\|\mathbf{a} - \mathbf{b}\|_2^2}{2r^2}\right) \tag{22}$$

where the kernel bandwidth r is median$(\|\mathbf{a} - \mathbf{b}\|_2)$, $\forall \mathbf{a}, \mathbf{b} \in \mathcal{S}_X \cup \mathcal{T}_X$.

2) Adaptive Factor μ: The adaptive factor μ expresses the relative importance of the marginal distributions and conditional distributions. Wang *et al.* [44] made the first attempt to compute μ by employing $\mathcal{A}$-distance [26], which is the special case $d_{\mathcal{H}}^{0-1}$ of the discrepancy distance $d_{\mathcal{H}}^{\ell}$. According to [26], the $\mathcal{A}$-distance can also be defined as the error of building a binary classifier from hypothesis set $\mathcal{H}$ to discriminate between the two domains. Wang *et al.* [44] used the linear hypothesis set to estimate $\mathcal{A}$-distance. Let $\epsilon(\boldsymbol{h})$ be the error of the linear classifier h discriminating source samples $\mathcal{S}_X$ and target samples $\mathcal{T}_X$. Then, the $\mathcal{A}$-distance

$$d_{\mathcal{A}}(\mathcal{S}_X, \mathcal{T}_X) = 2(1 - \epsilon(\boldsymbol{h})).$$

We adopted the same algorithm as [44] to estimate μ by

$$\mu = 1 - \frac{d_0}{d_0 + \sum_{c=1}^{C} d_c}$$

where $d_0 := d_{\mathcal{A}}(\mathcal{S}_X, \mathcal{T}_{X,K})$, $d_c := d_{\mathcal{A}}(\mathcal{S}_{X,c}, \mathcal{T}_{X,c})(c = 1, \cdots, C)$. Here, $\mathcal{T}_{X,K}$ is the set of the target samples predicted as known samples. This estimation has to be computed at every iteration of DAOD, since the predicted conditional distributions for the target may vary each time.

3) Open Set Parameters α and γ: As shown in Figs. 3 and 4, DAOD is able to achieve consistently good performance within the same range $\alpha \in [0.2, 0.4]$ and $\gamma \in [0.15, 0.5]$, which shows the relative stability of DAOD given the correct tuning of these two parameters. Tuning should be done according to the following rules. First, the positive term $R_{u,C+1}^t$ and the negative term $R_{u,C+1}^s$ in the open set difference are inferred from each other. A larger positive term means that more samples are recognized as unknown classes. A larger negative term implies that more samples are classified as known classes. To ensure that the balance of positive and negative terms, the difference $|\alpha - \gamma|$ should not be too large. Furthermore, the parameter α should be larger than γ, since the positive term's coefficient $1/(1 - \pi_{C+1}^t)$ is larger than 1. In this article, we set $\alpha = 0.4$ for all tasks and: 1) $\gamma = 0.2$ for **Office-31** and 2) $\gamma = 0.25$ for the **Office-Home** and **PIE** data sets.

4) Other Hyperparameters: We ran DAOD with a wide range of parameter values for λ, ρ, p, σ, t, and T in Section V-G. The results are shown in Fig. 4. These results indicate that DAOD can provide robust performance with a wide range of hyperparameter values.

From our tests, the best choices of parameters were: $\lambda \in [50, 1000]$, $\rho \in [0, 1]$, $p \in [2, 32]$, $\sigma \in [0.2, 1.6]$, and $t \in [0, 0.9]$, and DAOD can converge within 10 iterations. To sum up, the performance of DAOD stays robust with a large range of parameter choice. Therefore, the parameters do not need to be significantly fine tuned in practical applications. In this article, we fixed $p = 10$, $\rho = 1$ and $\sigma = 1$, $T = 10$, $t = 0.5$ and set: 1) $\lambda = 50$ for **Office-31** and 2) $\lambda = 500$ for the **Office-Home** and **PIE** data sets.

Although DAOD is easy to use, and its parameters do not have to be fine tuned, we did explore how to further tune these parameters for research purposes. We chose the parameters according to the following the rules: 1) The regularization term $\|\boldsymbol{h}\|_k^2$ is very important, and therefore, we tended to choose a slightly larger σ ($\sigma = 1$) to prevent DAOD from degenerating. 2) We chose ρ by following [25]. 3) p is set by following [25], [54]. 4) Distribution alignment is inevitable for DAOD, and therefore, we chose a larger λ ($\lambda \geq 50$) to make it count.

We used two types of accuracy [17], [18] to evaluate DAOD

$$\text{Acc(OS)} = \frac{1}{C+1} \sum_{c=1}^{C+1} \frac{|x : x \text{ from class } c \bigwedge \widetilde{f}(\mathbf{x}) = c|}{|x : x \text{ from class } c|} \tag{23}$$

and

$$\text{Acc(OS}^*) = \frac{1}{C} \sum_{c=1}^{C} \frac{|x : x \text{ from class } c \bigwedge \widetilde{f}(\mathbf{x}) = c|}{|x : x \text{ from class } c|} \tag{24}$$

where $\widetilde{f}$ is the predicted classifier. Note that Acc(OS) is the main index for evaluating the performance of the UOSDA algorithms [17].

D. Experimental Results

The classification accuracy of the UOSDA tasks is shown in Table II. The following facts can be observed from Table II: 1) the closed set algorithm TCA performed poorly on most tasks, even worse than the standard OSNN algorithm, indicating that negative transfer occurred. 2) All open set algorithms achieved better classification accuracy than OSNN on most tasks. This is because the source samples and the known target samples have different distributions. 3) DAOD achieved much better performance Acc(OS) than the six baseline algorithms on most tasks (24 out of 38). The average classification accuracy (Acc(OS), Acc(OS*)) of DAOD on the 38 tasks was 69.3% and 70.4%, respectively, gaining a performance improvement of 3.4% and 3.6% compared with the best baseline OSBP. 4) In general, JDA + OSNN, JGSA + OSNN, and ATI + OSNN algorithms did not perform and also DAOD. A major limitation of these algorithms may be that they omit the selected unknown target samples when they construct a latent space to match the distributions for the known classes. This may result in the unknown samples being mixed with the known samples in the latent space. In DAOD, the negative

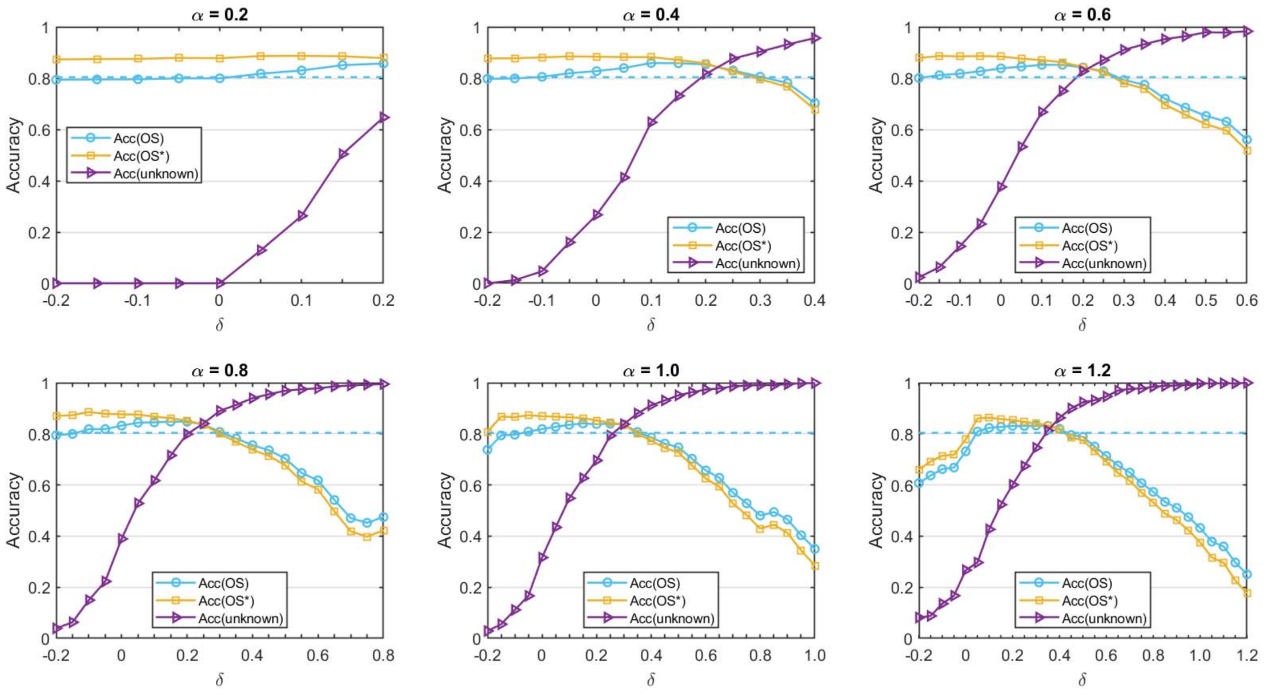

Fig. 3. Horizontal axis is the difference in the open set parameters $\delta = \alpha - \gamma$. Difference δ is not larger than α, since the parameter γ is required to be larger than or equal to 0. If $\delta > 0$, α is larger than γ. If $\delta < 0$, γ is larger.

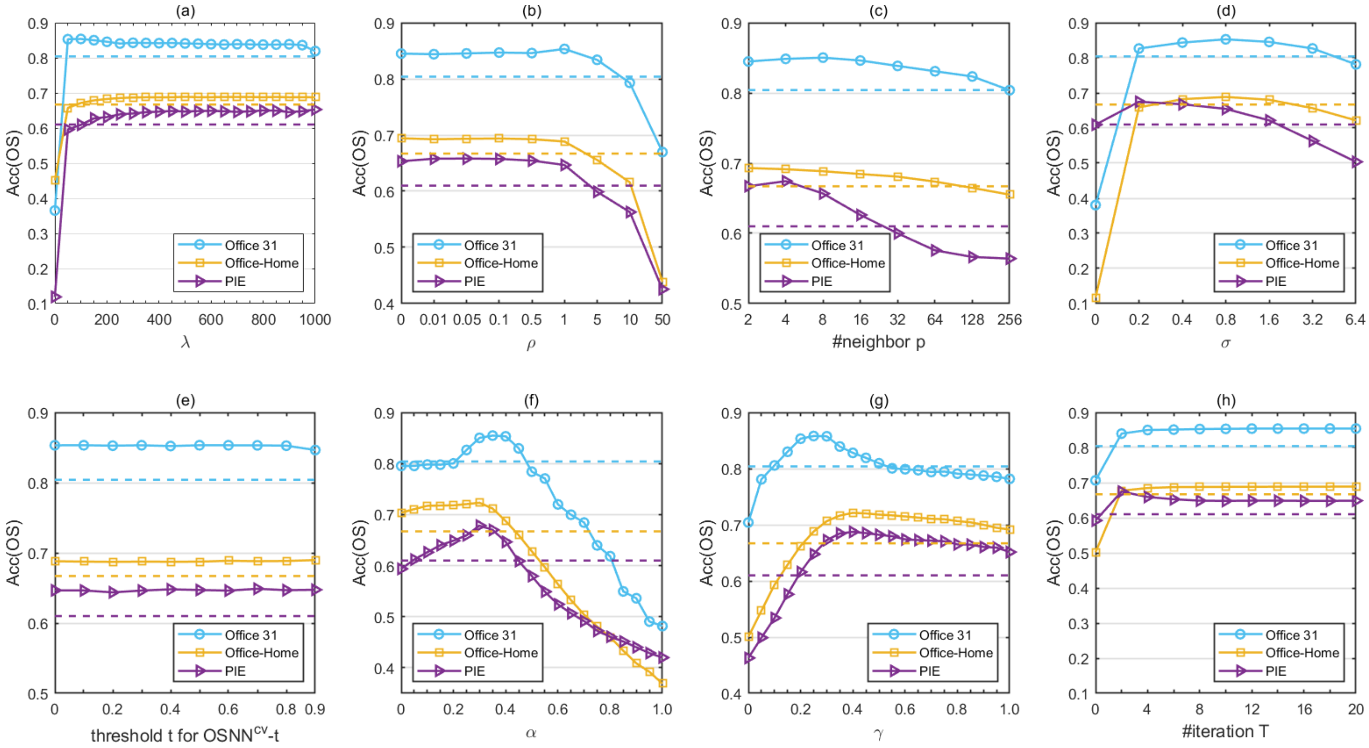

Fig. 4. Parameter sensitivity study, ablation study, and convergence analysis of the proposed DAOD algorithm: (a) is the parameter analysis for λ; (b) is the parameter analysis for ρ; (c) is the parameter for ρ; (d) is the parameter analysis for σ; (e) is the parameter analysis for t; (f) is the analysis for α; (g) is the analysis for γ; and (h) is the analysis for T.

term $R^s_{u,C+1}$ helps DAOD to avoid the problem suffered by JDA, JGSA, and ATI. 5) The performance of the OSPB algorithm was generally worse than that of DAOD. The main reasons may be that: 1) OSBP only matches the marginal distributions, not the joint distributions and 2) OSBP does not keep the unknown target samples away from the known source samples, with the result that many unknown target samples are recognized as known samples. DAOD, however, uses the negative term $R^s_{u,C+1}$ to separate the source samples and unknown target samples.

E. Open Set Parameters Analysis

From our analysis of the open set parameters α and γ, we find that the relationship between α and γ is closely

TABLE II

ACC(OS*) AND ACC(OS) (%) ON **OFFICE-31**, **OFFICE-HOME**, AND **PIE** DATA SETS

Dataset	OSNN		TCA		JDA		JGSA		ATI		OSBP		DAOD	
	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS
A→W	56.0	54.0	54.8	54.8	63.0	64.8	75.7	75.2	70.6	69.7	69.1	70.1	**84.2**	**84.2**
A→D	75.4	71.9	68.0	67.1	70.1	70.6	74.8	73.3	85.9	84.0	76.4	76.6	**89.8**	**88.5**
D→A	62.6	60.3	53.4	52.7	60.4	60.7	62.4	61.5	68.3	67.6	62.3	62.5	**71.8**	**72.6**
D→W	93.0	88.1	84.6	80.9	98.4	94.7	98.0	93.2	95.8	94.1	94.6	**98.9**	**98.0**	96.0
W→A	58.6	56.8	56.1	55.6	62.5	62.6	64.0	62.9	64.0	62.8	**82.2**	**82.3**	72.9	74.2
W→D	99.3	93.3	97.8	94.8	99.3	96.1	**100.0**	94.4	97.8	94.5	96.8	**96.9**	97.5	96.3
Average	74.2	70.7	69.2	68.5	75.6	74.9	79.2	76.7	80.4	78.8	80.2	80.4	**85.7**	**85.3**
Ar→Pr	39.4	40.6	37.7	37.9	59.7	59.0	64.1	63.3	70.4	68.6	69.2	68.4	**72.6**	**71.8**
Ar→Cl	32.1	33.7	24.4	24.1	39.1	39.6	45.9	46.0	54.2	53.1	53.3	53.1	**55.3**	**55.4**
Ar→Rw	56.6	57.0	55.7	55.3	67.5	66.4	74.1	72.8	78.1	77.3	**79.1**	**78.0**	78.2	77.6
Cl→Ar	32.3	34.0	31.3	32.1	41.9	42.1	43.8	44.5	59.1	57.8	58.2	57.9	**59.1**	**59.2**
Cl→Pr	39.1	40.3	34.8	34.8	49.1	48.9	55.8	55.8	68.3	66.7	**72.4**	**71.6**	70.8	70.1
Cl→Rw	46.9	47.7	41.4	41.2	59.7	59.1	62.8	62.5	75.3	74.3	72.3	71.4	**77.8**	**77.0**
Rw→Ar	51.4	52.1	49.4	49.2	55.8	55.1	56.9	56.4	70.8	70.0	68.2	66.5	**71.3**	**70.5**
Rw→Cl	38.0	39.2	34.9	34.1	44.1	43.9	48.7	48.6	55.4	55.2	**59.2**	**57.8**	58.4	**57.8**
Rw→Pr	59.2	59.2	57.3	56.5	68.0	68.2	66.5	65.3	79.4	78.3	80.8	78.6	**81.8**	**80.6**
Pr→Ar	38.5	39.7	33.2	33.4	48.4	48.0	55.8	55.5	62.6	61.2	61.0	59.6	**66.7**	**65.8**
Pr→Cl	35.0	36.3	35.8	36.1	41.2	41.1	44.1	44.4	54.1	53.9	56.9	55.7	**60.0**	**59.1**
Pr→Rw	59.6	59.7	58.3	57.5	70.4	68.9	73.5	72.3	81.1	79.9	83.9	82.1	**84.1**	**82.2**
Average	44.0	45.0	41.2	41.0	53.8	53.4	57.7	57.3	67.4	66.4	67.9	66.7	**69.6**	**68.9**
P1→P2	32.1	34.3	20.6	21.4	42.1	41.3	55.4	54.4	44.0	41.9	**66.6**	**64.2**	57.3	56.5
P1→P3	46.5	48.3	20.2	20.3	50.0	49.1	54.4	53.5	56.3	53.6	**69.1**	**66.4**	53.1	52.2
P1→P4	60.1	61.2	30.7	30.5	62.3	61.2	63.2	61.8	67.9	64.6	80.0	76.2	**85.2**	**82.4**
P1→P5	22.9	26.1	10.6	11.5	28.3	28.2	35.8	35.7	45.4	43.3	**50.2**	**49.1**	47.3	46.1
P2→P1	35.6	37.9	25.4	25.5	47.9	47.3	68.5	67.2	59.5	56.7	54.2	52.9	**69.7**	**68.1**
P2→P3	61.5	62.5	38.8	38.3	62.9	61.4	62.5	61.3	56.3	53.6	63.5	61.5	**71.7**	**69.9**
P2→P4	71.0	71.4	49.3	48.5	71.6	69.6	78.6	76.9	77.1	73.5	81.3	87.6	**91.2**	**88.2**
P2→P5	28.5	31.2	20.4	20.7	37.3	37.1	49.0	48.0	36.7	34.9	44.2	41.2	**49.8**	**49.4**
P3→P1	43.3	45.2	20.1	20.4	51.1	50.6	66.9	65.5	**68.4**	**66.9**	61.0	61.3	**68.3**	66.6
P3→P2	53.5	54.8	37.3	36.5	64.2	62.5	66.9	65.2	55.0	52.4	64.6	64.1	**70.4**	**68.5**
P3→P4	64.9	65.4	34.6	34.2	68.5	66.6	75.6	73.8	74.0	70.5	76.9	74.7	**87.1**	**83.9**
P3→P5	34.6	37.0	12.7	13.0	39.2	39.0	42.5	41.8	47.1	44.8	46.7	46.3	**53.3**	**52.3**
P4→P1	56.5	57.7	24.8	24.6	64.2	62.4	75.8	73.9	66.8	63.7	68.7	67.2	**87.1**	**84.4**
P4→P2	78.1	78.0	64.0	62.1	75.2	72.4	78.3	76.1	78.1	74.4	**85.0**	82.2	84.8	**82.4**
P4→P3	78.3	78.3	33.8	33.3	81.5	78.9	**81.3**	**79.1**	61.7	58.7	67.6	66.9	80.0	77.6
P4→P5	43.1	44.8	17.1	17.7	52.1	50.9	**65.8**	**64.4**	48.5	46.2	63.8	59.9	61.3	59.9
P5→P1	23.2	25.7	11.6	12.8	29.6	30.2	46.4	45.9	23.5	30.2	**66.6**	**64.2**	60.6	59.2
P5→P2	26.5	28.4	18.3	18.3	31.0	31.1	**44.0**	**43.6**	36.7	34.9	35.8	35.4	34.8	35.0
P5→P3	31.0	32.7	12.3	13.3	33.1	32.9	**55.4**	**54.6**	41.9	39.9	46.3	45.1	44.4	44.6
P5→P4	37.2	38.9	19.4	20.0	49.7	49.1	63.8	62.7	58.6	55.8	53.5	52.2	**70.3**	**68.6**
Average	46.4	48.0	26.2	26.1	52.1	51.1	61.5	60.3	55.2	53.0	62.2	61.0	**66.4**	**64.8**
All avg	50.0	50.6	37.7	37.5	56.3	55.6	63.1	61.9	63.0	61.3	66.8	65.9	**70.4**	**69.3**

related to another parameter, the difference $\delta := \alpha - \gamma$. We conducted experiments on the **Office-31** data set with α ranging from 0.2 to 1.2 and δ ranging from -0.2 to α. Due to space limitations, the average results on **Office-31** are reported in Fig. 3. According to Fig. 3, we made the following observations.

1) As δ increased, the accuracy of the unknown classes also increased, since a larger positive term $R^t_{u,C+1}(\boldsymbol{h})$ means that more samples are recognized as unknown.
2) When $\delta < 0$ ($\alpha < \gamma$), for almost all $\alpha \in [0.2, 1.2]$, the performance Acc(OS) was poorer than the best baseline algorithm (dashed line). This is because when $\delta < 0$, more samples are recognized as known classes. Our theoretical results support this observation (Theorem 1) since the positive term's coefficient $1/(1-\pi^t_{C+1})$ is larger than the negative term's coefficient 1. Thus, δ should be larger than 0 ($\alpha > \gamma$).
3) All figures in Fig. 3 are similar for almost all α from 0.4 to 1.2, which implies that α maybe not the most important factor influencing the performance of DAOD. Rather, the difference δ is likely to be more important.
4) Performance Acc(OS) begins to decrease when δ is larger than 0.25 because more known samples are classified as unknown with a larger δ.
5) When α ranged between 0.2 to 1.2 and δ was chosen from $[0.05, 0.2]$, the performance Acc(OS) of DAOD δ was superior to the best baseline.
6) Although α is not the main factor influencing the performance of DAOD, we compare $\alpha < 1.0$ with $\alpha \geq 1.0$ in Fig. 3 and find that a smaller α achieves slightly better performance than a larger α. In general, we select α from $[0.2, 0.4]$ and δ from $[0.05, 0.25]$.

F. Parameter Sensitivity, Ablation Study, and Convergence Analysis

These studies were conducted on different types of data sets to demonstrate that: 1) a wide range of parameter values can be chosen to obtain satisfactory performance and 2) the open set difference and distribution alignment term are important and necessary. We evaluated important parameters $\lambda, \sigma, \rho, p, t, \alpha, \gamma$, and T, reporting the average results for data sets **Office-31**, **Office-Home**, and **PIE**, respectively. The dashed line denotes the results of the best baseline algorithm with each data set.

1) Distribution Alignment λ: We ran DAOD with varying values of λ. Fig. 4(a) plots the classification accuracy w.r.t. to different values of λ. From Fig. 4(a), we can see that: 1) when $\lambda = 0$, the performance was the worse than the baseline. 2) After the increasing of the λ from 0 to 50, the performance dramatically increased to equal that of the baseline. 3) From 50 to 1000, DAOD was stable with values of around $0.85, 0.7$, and 0.65 on the three data sets. Overall, the performance of DAOD with most values of λ was better than the baselines. We also found that larger values of λ resulted in a better distribution alignment, and, if we chose λ from $[50, 1000]$, we obtained better results than the best baseline algorithm.

2) Manifold Regularization ρ: In these experiments, we ran DAOD with varying values of ρ. Larger values of ρ increase the importance of manifold consistency in DAOD. From Fig. 4(b), we can see that: 1) DAOD's performance was steady and consistently good when $\rho \in [0, 1]$. 2) But, after the increasing of ρ from 1 to 5, its performance dramatically dropped below the baseline. 3) Furthermore, DAOD's continued to fall below the baseline from 5 to 50. The reason for this poor performance at $\rho \in [5, 50]$ is that when ρ is large, DAOD mainly focuses on the geometric information of the samples and ignores other information. Choosing λ from $[0, 1]$, however, provides the best results.

3) Nearest Neighbors p: We ran DAOD with varying values of p. If $p \to +\infty$, two samples which are not at all similar are connected. If $p \to 0$, limited similarity information between samples is captured, and thus, p should not be too large or too small. Fig. 4(c) shows that if p is selected from $[2, 32]$, the performance of our algorithm is better than the baseline. When $p > 32$, the performance of **PIE** was worse than the baseline. One reason may be that when p is large, the samples from different classes are connected, resulting in that samples from different classes share similar scores. From Fig. 4(c), p can be selected from $[2, 32]$.

4) Regularization σ: We ran DAOD with varying values of σ and plotted the classification accuracy, as shown in Fig. 4(d). Theoretically, when $\sigma \to 0$, the classifier degenerates and overfitting occurs. When $\sigma \to +\infty$, the classifier obtains a trivial result. From Fig. 4(d), we can see that: 1) When $\sigma = 0$, the performance was the worst and also much worse than the baseline. 2) However, after increasing of σ from 0 to 0.2, performance dramatically increased commensurate with the baseline. 3) From 0.2 to 1.6, DAOD was stable with values at around $0.85, 0.7$, and 0.65 on three data sets. 4) When $\sigma > 1.6$, the performance dramatically dropped again to below the baseline. According to Fig. 4(d), we can choose $\sigma \in [0.2, 1.6]$.

5) Threshold t: Fig. 4(b) shows the classification accuracy with varying values of t. Theoretically, the threshold t is determined by the openness $\mathbb{O}$. When openness $\mathbb{O} \to 1$, $t \to 0$. When openness $\mathbb{O} \to 0$, $t \to 1$. However, according to Fig. 4(e), DAOD performed steadily when the threshold t varies from $[0, 0.9]$. This is because: 1) As the number of iterations T increases, the effect of t tapers off. 2) OSNN$^{\text{cv}}$-t is not sensitive to t.

6) Open Set Parameter α: Fig. 4(f) plots the classification accuracy w.r.t. different values. Theoretically, when $\alpha \to 0$, the classifier can not recognize unknown samples, whereas, when $\alpha \to +\infty$, the classifier classifies all samples as unknown. These conjectures are verified by the results in Fig. 4(f), where the performance reaches its maximal point at around $\alpha = 0.3$ and then gradually drops as α increases. The performance was worst and lower than the baselines when $\alpha > 0.4$ because, at this parameter setting, many samples from known classes are classified as unknown. In general, we can choose α from $[0.2, 0.4]$.

7) Open Set Parameter γ: The classification accuracy w.r.t. different values of γ is shown in Fig. 4(g). Theoretically, when $\gamma \to +\infty$, DAOD keeps the unknown target samples away from known source samples. As a result, few samples are classified as unknown classes. When $\gamma \to 0$, more samples are classified as unknown, and when $\gamma < 0.15$, its performance was worse than the baselines. Conversely, as γ increased, DAOD's performance dramatically increased, reaching its maximal value at around $\gamma = 0.3$ before gradually dropping again as γ continues to increase. In general, we can choose $\gamma \in [0.15, 0.5]$.

8) Ablation Study: 1) α and γ are the two parameters that control the contribution of the open set difference. As shown in Fig. 4(f), setting α closer to 0 reduces the contribution of the open set difference and performance degrades compared with the optimal value of about $\alpha = 0.3$. Furthermore, as shown in Fig. 4(g), setting γ closer to 0 also reduces the contribution of the open set difference, and again performance degrades compared with the optimal value of about $\gamma = 0.3$. Therefore, we can safely draw the conclusion that our proposed open set difference is a necessary term for open set domain adaption. 2) λ is the parameter that controls the contribution of the distribution discrepancy. As shown in Fig. 4(a), when λ is 0, performance is much worse than at other values, which shows that this term also makes a significant contribution to the final domain adaptation performance. 3) ρ controls the contribution of the manifold regularization. Fig. 4(b) shows there is no significant change in performance when ρ is set in the range 0 to 1. These results indicate that ρ makes no significant contributions to DAOD and may even negatively affect its performance with values from 5 to 50. Though the contribution of manifold regularization is not significant, more experiments in Appendix E in the Supplementary Material show that the manifold regularization is necessary. 4) σ is used to avoid overfitting. As shown in Fig. 4(d), performance drops significantly when σ is set to 0. Thus, the term $\|\boldsymbol{h}\|_k^2$ is important to our algorithm.

9) Convergence Analysis: The results of the convergence analysis on the number of iterations T are provided in Fig. 4(h). As shown, DAOD reached a steady performance in only a few iterations ($T < 10$). This is a clear indication of the advantages of DAOD's ability to be trained in UOSDA tasks.

VI. Conclusion

To the best of our knowledge, this is the first work to present a theoretical analysis for open set domain adaptation. In deriving a theoretical bound, we discovered a special term, open set difference, which is crucial for recognizing unknown target samples. Using this open set difference, we then constructed an UOSDA algorithm, called DAOD. Extensive experiments show that DAOD outperforms several competitive algorithms.

In the future, we will mainly focus on *universal domain adaptation* [55], which is a unified domain adaptation framework that includes closed set domain adaptation, open set domain adaptation, and partial domain adaptation [56].

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful comments.

References

[1] J. Q. Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.

[2] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowl.-Based Syst.*, vol. 80, pp. 14–23, May 2015.

[3] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proc. AAAI*, 2008, pp. 677–682.

[4] S. Jialin Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[5] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[6] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2200–2207.

[7] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer joint matching for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1410–1417.

[8] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *Proc. NeurIPS*, 2007, pp. 601–608.

[9] Y. Yu and C. Szepesvári, "Analysis of kernel mean matching under covariate shift," in *Proc. ICML*, 2012, pp. 607–614.

[10] P. Wei, Y. Ke, and C. K. Goh, "Feature analysis of marginalized stacked denoising autoenconder for unsupervised domain adaptation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1321–1334, May 2019.

[11] S. Li, S. Song, and G. Huang, "Prediction reweighting for domain adaptation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1682–1695, Jul. 2017.

[12] N. Passalis and A. Tefas, "Unsupervised knowledge transfer using similarity embeddings," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 946–950, Mar. 2019.

[13] Z. Fang, J. Lu, F. Liu, and G. Zhang, "Unsupervised domain adaptation with sphere retracting transformation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[14] F. Liu, J. Lu, and G. Zhang, "Unsupervised heterogeneous domain adaptation via shared fuzzy equivalence relations," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 6, pp. 3555–3568, Dec. 2018.

[15] F. Liu, G. Zhang, and J. Lu, "Heterogeneous domain adaptation: An unsupervised approach," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, Mar. 3, 2020, doi: 10.1109/TNNLS.2020.2973293.

[16] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy regression transfer learning in Takagi–Sugeno fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1795–1807, Dec. 2017.

[17] P. P. Busto and J. Gall, "Open set domain adaptation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 754–763.

[18] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," in *Proc. ECCV*, Sep. 2018, pp. 156–171.

[19] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko, "Syn2real: A new benchmark for synthetic-to-real visual domain adaptation," 2018, *arXiv:1806.09755*. [Online]. Available: https://arxiv.org/abs/1806.09755

[20] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *Proc. NeurIPS*, 2005, pp. 1–4.

[21] M. Baktashmotlagh, M. Faraki, T. Drummond, and M. Salzmann, "Learning factorized representations for open-set domain adaptation," in *Proc. ICLR*, 2019, pp. 1–11.

[22] H. Liu, Z. Cao, M. Long, J. Wang, and Q. Yang, "Separate to adapt: Open set domain adaptation via progressive separation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2927–2936.

[23] H. Zhang, A. Li, X. Han, Z. Chen, Y. Zhang, and Y. Guo, "Improving open set domain adaptation using image-to-image translation," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2019, pp. 1258–1263.

[24] V. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.

[25] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[26] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. NeurIPS*, 2006, pp. 137–144.

[27] W.-Y. Deng, A. Lendasse, Y.-S. Ong, I. W.-H. Tsang, L. Chen, and Q.-H. Zheng, "Domain adaption via feature selection on explicit feature map," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1180–1190, Apr. 2019.

[28] F. Liu, J. Lu, B. Han, G. Niu, G. Zhang, and M. Sugiyama, "Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation," in *Proc. NeurIPS LTS Workshop*, 2019, pp. 1–21.

[29] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.

[30] M. Long, J. Wang, G. Ding, S. J. Pan, and P. S. Yu, "Adaptation regularization: A general framework for transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1076–1089, May 2014.

[31] J. Zhang, W. Li, and P. Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5150–5158.

[32] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. ICML*, 2015, pp. 97–105.

[33] A. Gretton *et al.*, "Optimal kernel choice for large-scale two-sample tests," in *Proc. NeurIPS*, 2012, pp. 1205–1213.

[34] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proc. AAAI*, 2018, pp. 4058–4065.

[35] P. J. Phillips, P. Grother, and R. J. Micheals, "Evaluation methods in face recognition," in *Handbook of Face Recognition*, 2nd ed. London, U.K.: Springer, 2011, pp. 551–574.

[36] P. R. M. Júnior *et al.*, "Nearest neighbors distance ratio open-set classifier," *Mach. Learn.*, vol. 106, no. 3, pp. 359–386, Mar. 2017.

[37] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2317–2324, Nov. 2014.

[38] R. Vareto, S. Silva, F. Costa, and W. R. Schwartz, "Towards open-set face recognition using hashing functions," in *Proc. IEEE Int. Joint Conf. Biometrics (IJCB)*, Oct. 2017, pp. 634–641.

[39] L. P. Jain, W. J. Scheirer, and T. E. Boult, "Multi-class open set recognition using probability of inclusion," in *Proc. ECCV*, 2014, pp. 393–409.

[40] Y. Zhang, T. Liu, M. Long, and M. I. Jordan, "Bridging theory and algorithm for domain adaptation," in *Proc. ICML*, 2019, pp. 7404–7413.

[41] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. COLT*, 2009, pp. 1–12.

[42] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1414–1430, Jul. 2017.

[43] B. Quanz and J. Huan, "Large margin transductive transfer learning," in *Proc. 18th ACM Conf. Inf. Knowl. Manage. CIKM*, 2009, pp. 1327–1336.

[44] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu, "Visual domain adaptation with manifold embedded distribution alignment," in *Proc. ACM Multimedia Conf. Multimedia Conf. MM*, 2018, pp. 402–410.

[45] M. Long, J. Wang, G. Ding, D. Shen, and Q. Yang, "Transfer learning with graph co-regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 7, pp. 1805–1818, Jul. 2014.

[46] J. Zhang, W. Li, and P. Ogunbona, "Unsupervised domain adaptation: A multi-task learning-based method," *Knowl.-Based Syst.*, vol. 186, Dec. 2019, Art. no. 104975.

[47] L. Li and Z. Zhang, "Semi-supervised domain adaptation by covariance matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2724–2739, Nov. 2019.

[48] C. Wang and S. Mahadevan, "Heterogeneous domain adaptation using manifold alignment," in *Proc. IJCAI*, T. Walsh, Ed., 2011, pp. 1541–1546.

[49] B. K. Natarajan, *Mach. Learning: A Theoretical Approach*. San Mateo, CA, USA: Morgan Kaufmann, 1991.

[50] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. ECCV*, 2010, pp. 213–226.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NeurIPS*, 2012, pp. 1106–1114.

[52] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5385–5394.

[53] T. Sim, S. Baker, and M. Bsat, "The CMU pose, illumination, and expression database," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1615–1618, Dec. 2003.

[54] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1548–1560, Aug. 2011.

[55] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2720–2729.

[56] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2724–2732.

Zhen Fang received the M.Sc. degree in pure mathematics from the School of Mathematical Sciences Xiamen University, Xiamen, China, in 2017. He is currently pursuing the Ph.D. degree with the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia.

He is currently a member of the Decision Systems and e-Service Intelligence Research Laboratory, Centre for Artificial Intelligence, University of Technology Sydney. His research interests include transfer learning and domain adaptation.

Jie Lu (Fellow, IEEE) received the Ph.D. degree from Curtin University, Perth, WA, Australia, in 2000.

She is currently a Distinguished Professor and the Director of the Center for Artificial Intelligence, University of Technology Sydney, Ultimo, NSW, Australia. She has received over 20 Australian Research Council (ARC) Laureate, ARC discovery projects, and government and industry projects. She has authored or coauthored six research books and over 450 articles in refereed journals and conference proceedings. Her main research interests are in the areas of fuzzy transfer learning, concept drift, decision support systems, and recommender systems.

She is a fellow of the IFSA and Australian Laureate. She has received various awards, such as the UTS Medal for Research and Teaching Integration in 2010, the Computer Journal Wilkes Award in 2018, the UTS Medal for Research Excellence in 2019, the IEEE TRANSACTIONS ON FUZZY SYSTEMS Outstanding Paper Award in 2019, and the Australian Most Innovative Engineer Award in 2019. She serves as the Editor-in-Chief for *Knowledge-Based Systems* (Elsevier) and the *International Journal of Computational Intelligence Systems*. She has delivered over 25 keynote speeches at international conferences and chaired 15 international conferences.

Feng Liu (Graduate Student Member, IEEE) received the B.Sc. degree in pure mathematics and the M.Sc. degree in probability and statistics from the School of Mathematics and Statistics, Lanzhou University, Lanzhou, China, in 2013 and 2015, respectively. He is currently pursuing the Ph.D. degree with the Centre for Artificial intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia. His research interests include domain adaptation and two-sample test.

Mr. Liu served as a Senior Program Committee Member for European Conference on Artificial Intelligence (ECAI) and a Program Committee Member for NeurIPS, International Conference on Machine Learning (ICML), International Joint Conferences on Artificial Intelligence (IJCAI), Conference on Information and Knowledge Management (CIKM), IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), The International Joint Conference on Neural Networks (IJCNN), and Conferences on Computational Intelligence System (ISKE). He has received the UTS Research Publication Award in 2018, the UTS-Faculty of Engineering and IT (FEIT) Higher Degree Research (HDR) Research Excellence Award in 2019, and the Best Student Paper Award of FUZZ-IEEE in 2019. He serves as a Reviewer for IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON FUZZY SYSTEMS (TFS), and IEEE TRANSACTIONS ON CYBERNETICS (TCYB).

Junyu Xuan (Member, IEEE) is currently a Post-Doctoral Research Fellow with the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia. He has authored or coauthored almost 40 articles in high-quality journals and conferences, including *Artificial Intelligence*, *Machine Learning*, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the *ACM Computing Surveys*, the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING (TKDE), the *ACM Transactions on Information Systems* (TOIS), and the IEEE IEEE Transactions on Cybernetics (TCYB). His main research interests include machine learning, Bayesian nonparametric learning, text mining, and web mining.

Guangquan Zhang received the Ph.D. degree in applied mathematics from Curtin University, Perth, WA, Australia, in 2001.

He is currently a Professor and the Director of the Decision Systems and e-Service Intelligent Research Laboratory, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia. He has received seven Australian Research Council (ARC) discovery project grants and many other research grants. He has authored 4 monographs, 5 textbooks, and 350 articles, including 160 refereed international journal articles. His research interests include fuzzy machine learning, fuzzy optimization, machine learning, and data analytics.

Dr. Zhang has served as a member for the editorial boards of several international journals. He received the ARC QEII Fellowship in 2005. He was the co-chair of several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering. He has also served as a guest editor for eight special issues of the IEEE TRANSACTIONS and other international journals.