# Domain Neural Adaptation

Sentao Chen, Zijie Hong, Mehrtash Harandi, *Member, IEEE,* and Xiaowei Yang

***Abstract*—Domain adaptation is concerned with the problem of generalizing a classification model to a target domain with little or no labeled data, by leveraging the abundant labeled data from a related source domain. The source and target domains possess different joint probability distributions, making it challenging for model generalization. In this paper, we introduce Domain Neural Adaptation (DNA): an approach that exploits nonlinear deep neural network to 1) match the source and target joint distributions in the network activation space and 2) learn the classifier in an end-to-end manner. Specifically, we employ the Relative Chi-Square divergence to compare the two joint distributions, and show that the divergence can be estimated via seeking the maximal value of a quadratic functional over the Reproducing Kernel Hilbert Space. The analytic solution to this maximization problem enables us to explicitly express the divergence estimate as a function of the neural network mapping. We optimize the network parameters to minimize the estimated joint distribution divergence and the classification loss, yielding a classification model that generalizes well to the target domain. Empirical results on several visual data sets demonstrate that our solution is statistically better than its competitors.**

***Index Terms*—Domain adaptation, joint distribution matching, Relative Chi-Square divergence, Reproducing Kernel Hilbert Space, neural network.**

## I. Introduction

DOMAIN adaptation is an important research topic in machine learning and computer vision that has applications in object recognition [1], [2], [3], [4], [5], text classification [6], [7], [8], and semantic segmentation [9], [10], to name a few. A domain is described by a joint probability distribution $P(\boldsymbol{x}, y)$ of the features $\boldsymbol{x}$ and the class label $y$ [11], [12]. Compared with the traditional i.i.d. supervised learning [13], domain adaptation considers a scenario where the training (source) and test (target) data are independently sampled from different but related joint probability distributions [14], [15].

Generally speaking, domain adaptation can be categorized into the unsupervised and semi-supervised settings, depending on whether there are labeled data in the target domain or not. In the unsupervised setting [16], [17], labeled data drawn from the source joint distribution $P^s(\boldsymbol{x}, y)$ and unlabeled data from the target distribution $P^t(\boldsymbol{x}) = \int P^t(\boldsymbol{x}, y)dy$ are accessible. In the semi-supervised setting [3], [7], besides these data, a small amount of labeled data sampled from the target joint distribution $P^t(\boldsymbol{x}, y)$ are also available. See Fig. 1 for an illustration of this semi-supervised domain adaptation setting, where the images are obtained from the Office-Home data set [18]. Given the source and target data from different joint distributions, the ultimate goal of domain adaptation is to learn a classification model that generalizes well to the target domain and correctly predicts the labels for the target data. In this work, we are interested in devising joint distribution neural matching solution to addressing both the unsupervised and semi-supervised domain adaptation problems.

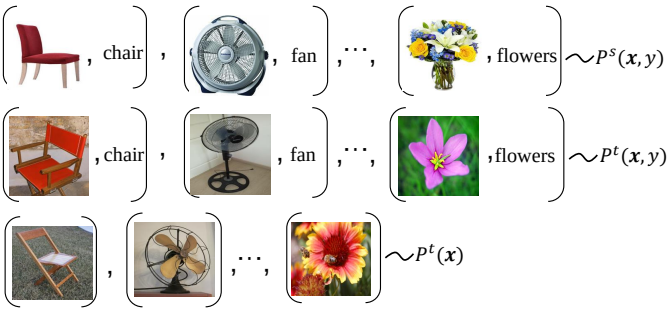



Fig. 1: Semi-supervised domain adaptation for object recognition, where the source and target data are drawn from different joint probability distributions. The image background is a crucial factor that results in the joint distribution mismatch.

In statistical learning, training a classification model can be easily achieved via minimizing the hinge loss, the negative log-likelihood loss (cross-entropy loss), *etc*. Therefore, prior approaches mainly focus on matching the probability distributions to tackle the domain adaptation problem. These approaches can be roughly divided into three categories: 1) marginal distribution matching [19], [20], [21], 2) class-conditional distribution matching [22], [23], [17], and 3) joint distribution matching [24], [25], [12].

Marginal distribution matching matches the source (marginal) distribution $P^s(\boldsymbol{x})$ and the target (marginal) one $P^t(\boldsymbol{x})$ via feature transformation. The transformation is usually performed by a dimension reduction matrix for the shallow methods [19], [20] or a neural network for the deep methods [26], [1], [21], [27]. The distribution similarity metric can be the distribution-ratio-based $f$-divergence (*e.g.*, the Kullback-Leibler (KL) divergence, the Jensen-Shannon (JS) divergence), the distribution-difference-based $L^p$-distance (*e.g.*, $L^1$-distance, $L^2$-distance), the Integral Probability Metrics (IPMs) [28] (*e.g.*, the Maximum Mean Discrepancy (MMD) [29], the Wasserstein distance [30]), to name a few. These metrics are approximated from observed samples via computing the sample means in the Reproducing Kernel Hilbert Space (RKHS) [16], [26], modeling the probability

This work was supported in part by the Technology and Innovation Major Project of the Ministry of Science and Technology of China under Grant 2020AAA0108404; in part by the National Natural Science Foundation of China under Grant 62106137; and in part by the Shantou University under Grant NTF21035. *(Corresponding author: Sentao Chen.)*

S. Chen is with the Department of Computer Science, Shantou University, Shantou 12386, China (e-mail: sentaochen@yahoo.com).

Z. Hong and X. Yang are with the School of Software Engineering, South China University of Technology, Guangzhou 510006, China.

M. Harandi is with the Department of Electrical and Computer Systems Engineering, Monash University, and Data61-CSIRO, Australia.

distributions [19], [31], or maximizing a loss functional with respect to a domain discriminator [1], [21]. Although theoretical analysis by Ben-David *et al*. [32] has shown that minimizing the divergence between the marginal distributions is beneficial to reducing the target classifier's error, neglecting the labels while matching the distributions can in fact harm the performance of the classifier, as highlighted by a series of works [22], [33], [34], [17], [35].

Class-conditional distribution matching matches the source class-conditional distribution $P^s(\boldsymbol{x}|y)$ and the target one $P^t(\boldsymbol{x}|y)$ via learning new feature representations [22], [17], [23], [36]. The aforementioned distribution similarity metrics are seamlessly applied here, since for a fixed class label $y$, quantifying the similarity between $P^s(\boldsymbol{x}|y)$ and $P^t(\boldsymbol{x}|y)$ is in nature the same as measuring the similarity between $P^s(\boldsymbol{x})$ and $P^t(\boldsymbol{x})$. Besides, since estimating the discrepancy between the two class-conditionals requires the target labels, which are not available in unsupervised domain adaptation, some methods [37], [38], [34], [17], [35] make use of the technique introduced by Long *et al*. [39] to endow the target data with pseudo labels. While these methods have been shown to be empirically effective, their class-conditional distribution matching logic does not seem to be closely related to the joint distribution mismatch problem in domain adaptation [12].

Joint distribution matching matches the source joint distribution $P^s(\boldsymbol{x}, y)$ and the target one $P^t(\boldsymbol{x}, y)$ through dimensionality reduction [12], [7], optimal transport [24], or neural network transformation [25]. For instance, in the prior work [7] we introduced the Asymmetric Joint Distribution Matching (AJDM) approach, which employs a couple of asymmetric matrices to reduce the dimensionality of the data and matches the source and target joint distributions in a low dimensional subspace. Obviously, this shallow joint distribution matching solution is directly related to the joint distribution mismatch problem in domain adaptation.

In this work, we further demonstrate the benefits of our direct joint distribution matching idea, and extend our previous shallow implementation [7] to its deep counterpart to handle large and real image to image adaptation problems. To this end, we introduce a Domain Neural Adaptation (DNA) solution, which exploits the flexible and expressive Convolutional Neural Network (CNN) to transform the image features and match the source joint distribution $P^s(\boldsymbol{x}, y)$ and the target one $P^t(\boldsymbol{x}, y)$ in the network activation space[1]. In particular, we quantify the similarity between the source and target joint distributions via the Relative Chi-Square (RCS) divergence [40], which has been demonstrated to be advantageous in various machine learning problems [41], [42], [7]. Importantly, here we show that the RCS divergence can be expressed as the maximal value of a quadratic functional, and when the domain of this functional is the RKHS [43], we further show that the maximal value can be approximated in an analytic way, yielding an explicit estimate for the RCS divergence. As a result, learning a target classification model can be reasonably and straightforwardly formulated as a minimization problem that optimizes the network parameters to minimize the estimated RCS divergence between the source and target joint distributions, and the negative log-likelihood loss (cross-entropy loss) of the classifier. We make use of the minibatch Stochastic Gradient Descent (SGD) algorithm to solve this problem, and expect the resulting classification model to generalize well to the target domain. In a nutshell, the major contributions of this work can be summarized as follows:

1) We introduce a DNA solution to the domain adaptation problem, which exploits the CNN to match the source and target joint distributions under the RCS divergence, and learns a well-performed target classification model.
2) We develop a principled approach to analytically approximate the RCS divergence between joint distributions, and provide theoretical analysis for the approximation error.
3) We evaluate our DNA on large and real object recognition data sets, and show that our solution is statistically better than the existing deep approaches that are based on marginal, class-conditional, or joint distribution matching.

The rest of this paper is structured as follows. Section II formalizes the domain adaptation problem, describes our motivation, approximates the RCS divergence, and presents our DNA model. Section III reviews related domain adaptation methods. Section IV discusses the proposed approach and other related ones from several perspectives. Section V provides comprehensive evaluation results and experimental analysis. Finally, Section VI concludes the paper.

## II. Methodology

### A. Problem Definition

According to [11], [12], we define a domain as a joint probability distribution $P(\boldsymbol{x}, y)$ of the features $\boldsymbol{x} \in \mathcal{X}$ and the label $y \in \mathcal{Y}$, where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the discrete label space. When it is clear from the context, we abbreviate $P(\boldsymbol{x}, y)$ to $P$ for notation convenience. The marginal distribution of a domain is defined as $P(\boldsymbol{x}) = \int P(\boldsymbol{x}, y)dy$. Let $h : \mathcal{X} \to \mathcal{Y}$ be a hypothesis from a hypothesis space $\mathcal{H}$, and $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ a loss function. The generalization error of $h$ with respect to a specific domain $P^\star(\boldsymbol{x}, y)$ is defined as $\mathrm{E}_\star[h] = \mathbb{E}_{(\boldsymbol{x},y)\sim P^\star(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}), y)] = \int \ell(h(\boldsymbol{x}), y)P^\star(\boldsymbol{x}, y)d\boldsymbol{x}dy$. Additionally, we denote an i.i.d. set of $m$ samples drawn from a joint distribution as $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m \sim P(\boldsymbol{x}, y)$. With these preparations, the domain adaptation problem is formally defined as (see for example [11], [12]):

**Definition 1.** *Let $P^s(\boldsymbol{x}, y)$ be a source domain and $P^t(\boldsymbol{x}, y)$ be a related target domain, $P^s(\boldsymbol{x}, y) \neq P^t(\boldsymbol{x}, y)$. In unsupervised domain adaptation, the labeled source data $\mathcal{D}^s = \{(\boldsymbol{x}_i^s, y_i^s)\}_{i=1}^{m_s} \sim P^s(\boldsymbol{x}, y)$ and unlabeled target data $\mathcal{D}^{tu} = \{\boldsymbol{x}_i^t\}_{i=1}^{m_{tu}} \sim P^t(\boldsymbol{x})$ are available. In semi-supervised domain adaptation, besides $\mathcal{D}^s$ and $\mathcal{D}^{tu}$, a small amount of labeled target data $\mathcal{D}^{tl} = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{m_{tl}} \sim P^t(\boldsymbol{x}, y)$ ($m_{tl} \ll m_s$) are also accessible. Given these data, the ultimate goal of unsupervised or semi-supervised domain adaptation is to pick a hypothesis $h^* \in \mathcal{H}$, such that its generalization error with respect to the target domain is minimized, i.e., $h^* = \mathrm{argmin}_{h\in\mathcal{H}}\mathrm{E}_t[h]$. In other words, $h^*$ should correctly predict the labels for the unlabeled set $\mathcal{D}^{tu}$.*

[1]Note that as aforementioned, we refer to a domain as a joint probability distribution, hence the name "Domain Neural Adaptation".

### B. Motivation

We show in the following theorem that under mild assumptions, the target error of a hypothesis $h$ is controlled by its source error and the Relative Chi-Square (RCS) divergence [40] between the source joint distribution $P^s(\boldsymbol{x}, y)$ and the target one $P^t(\boldsymbol{x}, y)$. The RCS divergence between $P^s(\boldsymbol{x}, y)$ and $P^t(\boldsymbol{x}, y)$ is defined as the Chi-Square divergence between $P^s(\boldsymbol{x}, y)$ and the $\alpha$-mixture joint distribution $P^\alpha(\boldsymbol{x}, y) = \alpha P^s(\boldsymbol{x}, y) + (1-\alpha)P^t(\boldsymbol{x}, y)$ for $0 \leq \alpha < 1$, *i.e.*,

$$\mathrm{RCS}_\alpha(P^s, P^t) = \int \Big[\Big(\frac{P^s(\boldsymbol{x}, y)}{P^\alpha(\boldsymbol{x}, y)}\Big)^2 - 1\Big] P^\alpha(\boldsymbol{x}, y) d\boldsymbol{x} dy. \tag{1}$$

This divergence compares distributions $P^s(\boldsymbol{x}, y)$ and $P^t(\boldsymbol{x}, y)$ based on the ratio $\frac{P^s(\boldsymbol{x}, y)}{P^\alpha(\boldsymbol{x}, y)}$. It is non-negative, upper bounded by $\frac{1}{\alpha} - 1$ for $\alpha \in (0, 1)$, and equals to zero if and only if $P^s(\boldsymbol{x}, y) = P^t(\boldsymbol{x}, y)$ [40], [7]. Given the definition of the RCS divergence, we can now present the theorem as follows:

**Theorem 1.** *Assume that the loss $\ell \leq M$ for some $M > 0$. Then, for any hypothesis $h \in \mathcal{H}$,*

$$\mathrm{E}_t[h] \leq \mathrm{E}_s[h] + \frac{\sqrt{2}M}{1-\alpha}\sqrt{\mathrm{RCS}_\alpha(P^s, P^t)}. \tag{2}$$

*Proof.* Please see the supplementary material. □

Theorem 1 suggests that to obtain a classification model that performs well in the target domain (*i.e.*, achieves a small target error), we should try to minimize 1) the source error and 2) the RCS divergence between the source and target joint distributions. Since these two terms are unknown in practice, we therefore minimize their empirical estimates obtained from the observed samples. Specifically, we employ the Convolutional Neural Network (CNN) model that chains feature extraction and classification into a pipeline, and optimize its parameters to simultaneously minimize the classification error, and the estimated RCS divergence between the source and target joint distributions in the network activation space. In the following subsections, we first show how the RCS divergence is estimated, and then introduce our optimization problem for learning the parameters of the CNN model.

**Remark 1.** *We note that the RCS divergence is advantageous over the MMD from the perspective of reducing the joint distribution discrepancy. Particularly, MMD was designed and employed for measuring the discrepancy between marginal distributions (i.e., feature distributions) $P^s(\boldsymbol{x})$ and $P^t(\boldsymbol{x})$ [29], [16], and may not be suitable for quantifying the discrepancy between joint distributions $P^s(\boldsymbol{x}, y)$ and $P^t(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$. By contrast, the RCS divergence can not only be employed for measuring the discrepancy between marginal distributions [40], but also the discrepancy between joint distributions [7]. Therefore, with the RCS divergence, we can better reduce the (feature and label) joint distribution discrepancy between domains.*

### C. Divergence Approximation

We show that the RCS divergence between the source and target joint distributions can be approximated from samples in a direct and analytic manner. In addition to the labeled source data set $\mathcal{D}^s = \{(\boldsymbol{x}_i^s, y_i^s)\}_{i=1}^{m_s} \sim P^s(\boldsymbol{x}, y)$, we assume for the moment that a labeled target set $\mathcal{D}^t = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{m_t} \sim P^t(\boldsymbol{x}, y)$ is also available. Later in the next subsection, we explain how this set is obtained under the settings of unsupervised and semi-supervised domain adaptation.

To be specific, the RCS divergence in Eq. (1) can be alternatively expressed and approximated as

$$\begin{aligned}
&\mathrm{RCS}_\alpha(P^s, P^t) \\
&= \max_r \int \Big(2\frac{P^s(\boldsymbol{x}, y)}{P^\alpha(\boldsymbol{x}, y)} r(\boldsymbol{x}, y) - r(\boldsymbol{x}, y)^2 - 1\Big) P^\alpha(\boldsymbol{x}, y) d\boldsymbol{x} dy \qquad (3)\\
&= \max_r \Big(2\int r(\boldsymbol{x}, y) P^s(\boldsymbol{x}, y) d\boldsymbol{x} dy \\
&\qquad - \alpha \int r(\boldsymbol{x}, y)^2 P^s(\boldsymbol{x}, y) d\boldsymbol{x} dy \\
&\qquad - (1-\alpha)\int r(\boldsymbol{x}, y)^2 P^t(\boldsymbol{x}, y) d\boldsymbol{x} dy\Big) - 1 \qquad (4)\\
&\approx \max_r \Big(\frac{2}{m_s}\sum_{i=1}^{m_s} r(\boldsymbol{x}_i^s, y_i^s) - \frac{\alpha}{m_s}\sum_{i=1}^{m_s} r(\boldsymbol{x}_i^s, y_i^s)^2 \\
&\qquad - \frac{1-\alpha}{m_t}\sum_{i=1}^{m_t} r(\boldsymbol{x}_i^t, y_i^t)^2\Big) - 1. \qquad (5)
\end{aligned}$$

Eq. (3) is obtained from Eq. (1) by leveraging the equation $u^2 - 1 = u^2 - 1 - \min_v (v-u)^2 = \max_v (2uv - v^2 - 1)$, where $\frac{P^s(\boldsymbol{x}, y)}{P^\alpha(\boldsymbol{x}, y)} = u$ and $r(\boldsymbol{x}, y) = v$. This equation also implies that the maximizer and the maximal value of the quadratic functional $F(r)$ in Eq. (3) are the ratio function $\frac{P^s(\boldsymbol{x}, y)}{P^\alpha(\boldsymbol{x}, y)}$ and Eq. (1), respectively. Eq. (4) is obtained by expanding the $\alpha$-mixture joint distribution $P^\alpha(\boldsymbol{x}, y)$. Eq. (5) approximates the three expectations in Eq. (4) by their empirical averages.

Clearly, finding the maximizer of the functional $\widehat{F}(r)$ in Eq. (5) is key to the RCS divergence approximation. For practical implementation, we restrict the hypothesis space from which the function $r$ is searched to the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{R}$ induced by a product kernel $g\big((\boldsymbol{x}_i, y_i), (\boldsymbol{x}_j, y_j)\big) = k(\boldsymbol{x}_i, \boldsymbol{x}_j)\delta(y_i, y_j)$, where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\big(\frac{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\sigma}\big)$ is the Gaussian kernel with kernel width $\sigma$ $(> 0)$, and $\delta(y_i, y_j)$ is the delta kernel that evaluates 1 if $y_i = y_j$ and 0 otherwise. Such restriction is appropriate as the resulting hypothesis space is sufficiently rich, but also amenable to efficient optimization procedures [43]. Hence, the maximizer is obtained by solving the following optimization problem over the RKHS $\mathcal{R}$

$$\widehat{r} = \operatorname*{argmax}_{r \in \mathcal{R}} \widehat{F}(r) - \lambda \|r\|_{\mathcal{R}}^2, \tag{6}$$

where $\lambda\|r\|_{\mathcal{R}}^2$ is a regularization term introduced to avoid overfitting and $\lambda$ $(\geq 0)$ a regularization parameter. According to the representer theorem [43], the maximizer $\widehat{r}$ in Eq. (6) can be represented by a linear combination of samples from the union set $\mathcal{D}^s \cup \mathcal{D}^t$. That is, $\widehat{r} = \sum_{i=1}^{m_{st}} \beta_i g\big((\cdot, \cdot), (\boldsymbol{x}_i, y_i)\big)$, where $m_{st} = m_s + m_t$, $(\boldsymbol{x}_i, y_i) \in \mathcal{D}^s \cup \mathcal{D}^t = \{(\boldsymbol{x}_1^s, y_1^s), \cdots, (\boldsymbol{x}_{m_t}^t, y_{m_t}^t)\} = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_{m_{st}}, y_{m_{st}})\}$, and $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_{m_{st}})^\top$ is the coefficient vector to be

learned. Under such representation, optimization problem (6) becomes

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{m_{st}}}{\operatorname{argmax}} \Big( \frac{2}{m_s}\sum_{i=1}^{m_s} \widehat{r}(\boldsymbol{x}_i^s, y_i^s; \boldsymbol{\beta}) - \frac{\alpha}{m_s}\sum_{i=1}^{m_s} \widehat{r}(\boldsymbol{x}_i^s, y_i^s; \boldsymbol{\beta})^2 - \frac{1-\alpha}{m_t}\sum_{i=1}^{m_t} \widehat{r}(\boldsymbol{x}_i^t, y_i^t; \boldsymbol{\beta})^2 - \lambda \boldsymbol{\beta}^\top \boldsymbol{G}\boldsymbol{\beta} \Big) \tag{7}$$

$$= \underset{\boldsymbol{\beta}\in\mathbb{R}^{m_{st}}}{\operatorname{argmax}} \Big( \frac{2}{m_s}\mathbf{1}_{m_s}^\top \boldsymbol{G}^s\boldsymbol{\beta} - \frac{\alpha}{m_s}\boldsymbol{\beta}^\top (\boldsymbol{G}^s)^\top \boldsymbol{G}^s\boldsymbol{\beta} - \frac{1-\alpha}{m_t}\boldsymbol{\beta}^\top (\boldsymbol{G}^t)^\top \boldsymbol{G}^t\boldsymbol{\beta} - \lambda \boldsymbol{\beta}^\top \boldsymbol{G}\boldsymbol{\beta} \Big) \tag{8}$$

$$= \underset{\boldsymbol{\beta}\in\mathbb{R}^{m_{st}}}{\operatorname{argmax}} \Big( 2\boldsymbol{b}^\top\boldsymbol{\beta} - \boldsymbol{\beta}^\top(\boldsymbol{H}+\lambda\boldsymbol{G})\boldsymbol{\beta} \Big) \tag{9}$$

$$= (\boldsymbol{H}+\lambda\boldsymbol{G})^{-1}\boldsymbol{b}. \tag{10}$$

Eq. (7) is the direct result of plugging the non-parametric form of $\widehat{r}$ into Eq. (6). Eq. (8) rearranges some terms and expresses them in matrix forms, where $\mathbf{1}_{m_s}$ is an $m_s$-dimensional column vector of ones, $\boldsymbol{G}^s \in \mathbb{R}^{m_s\times m_{st}}$, $\boldsymbol{G}^t \in \mathbb{R}^{m_t\times m_{st}}$, and $\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}^s \\ \boldsymbol{G}^t \end{pmatrix} \in \mathbb{R}^{m_{st}\times m_{st}}$. The $(i,j)$-th elements of $\boldsymbol{G}^s$ and $\boldsymbol{G}^t$ are respectively defined as $g_{ij}^s = g\big((\boldsymbol{x}_i^s, y_i^s), (\boldsymbol{x}_j, y_j)\big)$ and $g_{ij}^t = g\big((\boldsymbol{x}_i^t, y_i^t), (\boldsymbol{x}_j, y_j)\big)$. Eq. (9) introduces two notations $\boldsymbol{b} = \frac{1}{m_s}(\boldsymbol{G}^s)^\top \mathbf{1}_{m_s}$ and $\boldsymbol{H} = \frac{\alpha}{m_s}(\boldsymbol{G}^s)^\top\boldsymbol{G}^s + \frac{1-\alpha}{m_t}(\boldsymbol{G}^t)^\top\boldsymbol{G}^t$ to make the unconstrained quadratic optimization problem clearer. Finally Eq. (10) presents the analytic solution to the problem. Consequently, the RCS divergence between $P^s(\boldsymbol{x}, y)$ and $P^t(\boldsymbol{x}, y)$ is explicitly estimated as

$$\widehat{\mathrm{RCS}}_\alpha\big(P^s, P^t\big) = \frac{2}{m_s}\sum_{i=1}^{m_s} \widehat{r}(\boldsymbol{x}_i^s, y_i^s; \widehat{\boldsymbol{\beta}}) - \frac{\alpha}{m_s}\sum_{i=1}^{m_s} \widehat{r}(\boldsymbol{x}_i^s, y_i^s; \widehat{\boldsymbol{\beta}})^2 - \frac{1-\alpha}{m_t}\sum_{i=1}^{m_t} \widehat{r}(\boldsymbol{x}_i^t, y_i^t; \widehat{\boldsymbol{\beta}})^2 - 1 \tag{11}$$

$$= 2\boldsymbol{b}^\top\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\top\boldsymbol{H}\widehat{\boldsymbol{\beta}} - 1. \tag{12}$$

To be comprehensive, we further analyze the approximation error between the estimated RCS divergence in Eq. (12) and its true value in Eq. (1). The following theorem shows that in a probability sense, this approximation error is associated with the hypothesis space $\mathcal{R}$, the number of source samples $m_s$, and the number of target samples $m_t$.

**Theorem 2.** *Assume that for all $r \in \mathcal{R}$, there exists $\widetilde{r} \in \mathcal{R}$, such that $|\widehat{F}(r) - F(r)| \le |\widehat{F}(\widetilde{r}) - F(\widetilde{r})|$. Then, for any $\delta \in (0,1)$, there exists a positive integer $N_\delta$, such that when $\min(m_s, m_t) > N_\delta$, with probability at least $1-\delta$,*

$$\Big|\widehat{\mathrm{RCS}}_\alpha\big(P^s, P^t\big) - \mathrm{RCS}_\alpha\big(P^s, P^t\big)\Big| \le 3\Big(\frac{1}{m_s} + \frac{1}{m_t}\Big) + \max_r F(r) - \max_{r\in\mathcal{R}} F(r). \tag{13}$$

*Proof.* Please see the supplementary material. □

According to Theorem 2, if the hypothesis space $\mathcal{R}$ contains the maximizer of $F(r)$, and $m_s, m_t$ are both sufficiently large, then it is probable that the estimated divergence $\widehat{\mathrm{RCS}}_\alpha\big(P^s, P^t\big)$ is arbitrarily close to the true value $\mathrm{RCS}_\alpha(P^s, P^t)$. In our proposed estimation method, since the RKHS $\mathcal{R}$ that we choose is sufficiently rich [43], we believe that it can at least contain a function that is close to the maximizer of $F(r)$, making $\max_r F(r) - \max_{r\in\mathcal{R}} F(r)$ a small value. Besides, for the sample sizes $m_s$ and $m_t$, in applications like object recognition, which is our interest in this paper, it is feasible to collect enough samples. Therefore, we can expect Eq. (12) to be a reliable estimate of the RCS divergence, and to consequently ensure that our DNA solution, containing this estimate as a crucial component, works well on such applications.

**Remark 2.** *Thanks to the delta kernel $\delta(y_i, y_j)$, the $m_{st}\times m_{st}$ matrix $(\boldsymbol{H}+\lambda\boldsymbol{G})$ in Eq. (10) can be simplified to a block-diagonal matrix, if the union set $\mathcal{D}^s \cup \mathcal{D}^t$ is sorted in a class-wise manner, i.e., the samples belonging to the same class are sequentially put together. For example, suppose there are $c$ classes in this set, the $i$-th class has $m_i$ samples. Then it can be easily verified that $(\boldsymbol{H}+\lambda\boldsymbol{G})$ is a block-diagonal matrix containing $c$ blocks, with the $i$-th block being a $m_i\times m_i$ matrix. Clearly, solving the inverse of this block-diagonal matrix has computational advantage over the original one.*

**Remark 3.** *Theorem 2 presents a simple but easily interpretable probabilistic error bound. To obtain more elaborate results, one can further exploit the Rademacher complexity [44] to quantify the richness of the hypothesis space $\mathcal{R}$, and the concentration inequalities like Talagrand's lemma [45] to bound the divergence approximation error.*

### D. Domain Neural Adaptation

Our DNA solution optimizes the CNN parameters to minimize the classification loss, and the estimated RCS divergence between the source and target joint distributions. Before proceeding to elaborate on the optimization problem, we elucidate how the labeled target set $\mathcal{D}^t$ used in the previous subsection is obtained in domain adaptation. To be specific, we utilize the commonly practiced pseudo labeling technique [39], [35], [7] to endow the target data with pseudo labels. In unsupervised domain adaptation, we set $\mathcal{D}^t = \{(\boldsymbol{x}_i^t, \widehat{y}_i^t)\}_{i=1}^{m_t}$, where $m_t = m_{tu}$ and $\widehat{y}_i^t$ is the pseudo label predicted by a CNN model trained on the labeled source data. In semi-supervised domain adaptation, we set $\mathcal{D}^t = \mathcal{D}^{tl} \cup \{(\boldsymbol{x}_i^t, \widehat{y}_i^t)\}_{i=1}^{m_{tu}}$, where the number of samples in $\mathcal{D}^t$ is $m_t = m_{tl} + m_{tu}$ and $\widehat{y}_i^t$ is the pseudo label predicted by a CNN model trained on both the labeled source and target data.

Combining the negative log-likelihood loss for neural network classification and the estimated RCS divergence between joint distributions, the optimization problem of our Domain Neural Adaptation (DNA) solution to unsupervised domain adaptation is formulated as

$$\min_{\boldsymbol{\Theta}} \frac{-1}{m_s}\sum_{i=1}^{m_s} \log P(y_i^s|\boldsymbol{x}_i^s; \boldsymbol{\Theta}) + \gamma\widehat{\mathrm{RCS}}_\alpha(P^s, P^t), \tag{14}$$

where $\boldsymbol{\Theta}$ is a set containing the CNN model parameters, and $\gamma$ $(>0)$ is a tradeoff parameter between the classification loss and the estimated RCS divergence. For clarity, we graphically reflect this solution in Fig. 2, where the network model could be one of the popular CNNs including AlexNet [46], VGGNet [47], and ResNet [48]. As illustrated in Fig. 2,

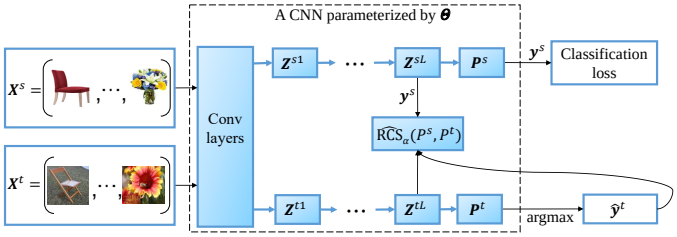



Fig. 2: Illustration of our DNA solution with a CNN model parameterized by $\boldsymbol{\Theta}$. We optimize $\boldsymbol{\Theta}$ to jointly minimize the estimated joint distribution divergence $\widehat{\mathrm{RCS}}_\alpha(P^s, P^t)$ and the classification loss, *i.e.*, the negative log-likelihood loss. In particular, $\widehat{\mathrm{RCS}}_\alpha(P^s, P^t)$ takes as its inputs the source labels $\boldsymbol{y}^s$, the pseudo target labels $\widehat{\boldsymbol{y}}^t$, the source data $\boldsymbol{Z}^{sL}$ and the target data $\boldsymbol{Z}^{tL}$ from the penultimate layer, which is followed by the probabilistic classifier layer that yields the source and target probability matrices $\boldsymbol{P}^s$ and $\boldsymbol{P}^t$ via the softmax transformation.

$\widehat{\mathrm{RCS}}_\alpha(P^s, P^t)$ takes as its inputs the source labels $\boldsymbol{y}^s$, the pseudo target labels $\widehat{\boldsymbol{y}}^t$, the source and target data $\boldsymbol{Z}^{sL}, \boldsymbol{Z}^{tL}$ represented by the activation features from the penultimate layer, which is followed by the probabilistic classifier layer with the softmax transformation. In addition, regarding our DNA solution to semi-supervised domain adaptation, the small amount of labeled target data should also be included when calculating the classification loss and the estimated RCS divergence in optimization problem (14).

We employ the minibatch SGD algorithm to solve minimization problem (14), with the CNN model parameters pretrained on ImageNet. In every iteration of the optimization algorithm, after the network parameters are updated, the pseudo target labels are also updated correspondently. As noted in [35], this refines the pseudo labels and makes them become more accurate.

**Remark 4.** *To a certain extent, our minimization problem* (14) *relates to the previous minimax (adversarial) problems in [1], [21]. In particular, the distribution divergences are all characterized as the maximal values of certain functionals, and these maximal values are minimized with respect to the neural network parameters that control the feature extractors. In our work, because of the RCS divergence and the RKHS, we are able to provide analytic solution to the functional maximization problem, thus converting the minimax problem into a simple minimization problem.*

**Remark 5.** *In the work of Damodaran* et al. *[25], joint distribution matching was also explored in the CNN context, but with the Wasserstein distance being the distribution discrepancy metric. There, the optimization problem not only involves optimizing the CNN parameters, but also an optimal transport matrix to express the Wasserstein distance, both of which are optimized in an alternative manner in every iteration of the optimization algorithm. By contrast, here in our work, since the RCS divergence can be analytically approximated and has an explicit estimate, i.e., Eq.* (12)*, our optimization problem* (14) *is much more straightforward: it only involves optimizing the CNN parameters, which is readily solved via the minibatch SGD algorithm.*

## III. Related Work

We summarize the relevant domain adaptation works from a statistical viewpoint, and refer the readers to [49], [10] for a comprehensive survey of more works.

*1) Marginal Distribution Matching:* Baktashmotlagh *et al.* [31] proposed the Distribution Matching Embedding (DME) framework that projects the unlabeled source and target data into a Grassmannian subspace where the source and target marginal distributions are similar under the MMD distance, the KL divergence, or the Hellinger distance. Ganin *et al.* [1] introduced the Domain-Adversarial Neural Network (DANN) that simultaneously minimizes the negative log-likelihood loss on the labeled source data to learn discriminative features, and the $\mathcal{H}$-divergence between the unlabeled source and target data to learn domain invariant features. Long *et al.* [26] aligned the source and target activation distributions under the Multi-Kernel Maximum Mean Discrepancy (MK-MMD) in a layerwise manner for multiple task-specific layers, and made the deep features more transferable by employing the low-density separation criterion on the unlabeled target data. Additionally, Kang *et al.* [27] matches the source and target activation distributions by minimizing both the JS divergence and the MMD distance between them.

*2) Class-Conditional Distribution Matching:* Quanz *et al.* [22] leveraged the sparse coding technique to match the class-conditional distributions $P^s(\boldsymbol{x}|y)$ and $P^t(\boldsymbol{x}|y)$, as well as the marginal distributions $P^s(\boldsymbol{x})$ and $P^t(\boldsymbol{x})$ under the JS divergence, which is estimated via modeling the distributions with kernel density estimation. Long *et al.* [39] proposed the Joint Distribution Adaptation (JDA) approach, which aims at reducing the dimensionality of the features such that the data variance is maximized, and the marginal and class-conditional distributions are matched between the source and target domains under the linear kernel MMD. It is noteworthy that this work introduces the simple but effective pseudo labeling technique to domain adaptation, which is recently studied from a theoretical perspective [50]. Adaptation Regularization based Transfer Learning (ARTL) [51] simultaneously optimizes the structural risk functional, joint adaptation of both the marginal and class-conditional distributions, and the manifold consistency. Moreover, building on the JDA approach, later works mainly concentrate on improving the discriminativeness of the data [37], [38], [52] or refining the pseudo labeling process [53]. In the deep neural network context, Hu *et al.* [36] consistently aligned the marginal and class-conditional distributions between domains by constraining the gradient of marginal and class-conditional alignment to be synchronous. Cicek *et al.* [17] aligned the source and target class-conditional distributions, encouraged them to have disjoint support, and finally employed semi-supervised learning tools to improve the generalization ability of the classifier. To address the partial and traditional domain adaptation problems, Li *et al.* [54] plugged one residual block into a unified network to mitigate the cross-domain distribution discrepancy, introduced

a weighted class-wise domain alignment loss to encourage accurate class-wise matching across domains, and explored domain-wise knowledge to adapt the global source information to the target.

*3) Joint Distribution Matching:* Nicolas *et al*. [24] presented the Joint Distribution Optimal Transport (JDOT) method, which simultaneously optimizes for a coupling between the source and target joint distributions, and a prediction function that solves the transfer problem. Damodaran *et al*. [25] extended this work to the deep neural network architectures, and minimized the discrepancy between the source and target joint distributions by means of optimal transport. More recently, Chen *et al*. [12] proposed the Joint Distribution Invariant Projections (JDIP) approach that leverages a couple of projections to directly align the source and target joint distributions under the $L^2$-distance, which is approximated via estimating the difference of the two joint distributions.

*4) Other Works:* Apart from the above works, there are also other important ones worth mentioning [55], [6], [56], [57]. For example, Gong *et al*. [58] modeled the source and target subspaces as points on the Grassmann manifold, and integrated all the intermediate subspaces between them along the geodesic to generate new representation for the data. Wang *et al*. [59] proposed the Neural Embedding Matching (NEM) method that enforces cross-domain discriminative embedding matching and preserves the local structural information in the target domain, and developed a progressive learning strategy to improve the proposed method during training.

Our work introduces the DNA approach that chains joint probability distribution matching and classifier learning into a pipeline using the CNN model. It shares similar ideas with prior works [51], [1], [26], [35], [54], [5] that address domain adaptation via distribution matching, but different from them in the sense that our work aims at directly matching the source joint distribution $P^s(\boldsymbol{x}, y)$ and the target one $P^t(\boldsymbol{x}, y)$, and not the marginal or class-conditional distributions. As noted in [24], [25], [12], aligning the source and target joint distributions is beneficial to domain adaptation. In addition, compared with its deep joint distribution matching counterpart [25], our DNA makes use of the RCS divergence, and solves a straightforward minimization problem via the minibatch SGD algorithm to produce a well-performed target predictor.

## IV. Discussion

We discuss our approach and some basic deep domain adaptation methods from the perspectives of model assumption, similarity metric, and optimization problem. These methods include DANN [1], ADDA [21], DAN [26], MADA [23], CDAN [60], GSDA [36], DSAN [35], and DeepJDOP [25]. We first graphically reflect these perspectives in Fig. 3, and then elaborate on the discussions of each perspective.

*1) Model Assumption:* Domain adaptation tackles the learning problem where the source and target joint distributions are different but related, *i.e*., $P^s(\boldsymbol{x}, y) \neq P^t(\boldsymbol{x}, y)$. This relationship is formalized into various model assumptions and expressed via the activation features $\boldsymbol{z}$ from the neural network. DANN, ADDA and DAN assume that $P^s(\boldsymbol{z}) \approx P^t(\boldsymbol{z})$. MADA, CDAN, GSDA, and DSAN presume that $P^s(\boldsymbol{z}|y) \approx P^t(\boldsymbol{z}|y)$. By contrast, DeepJDOT and our DNA assume that $P^s(\boldsymbol{z}, y) \approx P^t(\boldsymbol{z}, y)$, which connects intimately with the problem definition.

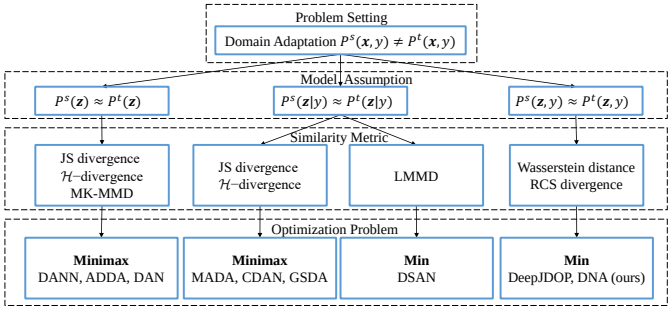



Fig. 3: Discussion of our DNA solution and other approaches from the perspectives of model assumption, similarity metric, and optimization problem.

*2) Similarity Metric:* Based on these model assumptions, deep domain adaptation methods aim to learn the activation features $\boldsymbol{z}$ (parameterized by the network parameters), such that the distributions are similar under a certain metric. DANN and MADA exploit the $\mathcal{H}$-divergence as the distribution similarity metric. ADDA, CDAN, and GSDA use the JS divergence in its variational form [61]. DAN and DSAN utilize the extensions of the MMD distance, *i.e*., MK-MMD [26] and Local Maximum Mean Discrepancy (LMMD) [35]. DeepJDOP relies on the Wasserstein distance. By comparison, our DNA opts for the RCS divergence, which has its advantages in comparing probability distributions [40], [42], [7].

*3) Optimization Problem:* Minimizing the network parameters with respect to different distribution divergences leads to various optimization problems. DANN, ADDA, DAN, MADA, CDAN, and GSDA involve solving a minimax optimization problem, since the JS divergence and the $\mathcal{H}$-divergence utilized by some of the methods are characterized as the maximal values of the loss functionals [61], [32]. By contrast, DSAN, DeepJDOT, and our DNA only involve solving a common minimization problem.

## V. Empirical Evaluation

We evaluate our approach on large and real image to image adaptation problems, and contrast its performance against other deep domain adaptation methods. The Pytorch implementation of our approach will be publicly available on Github when this paper is accepted. Below, we start by describing the data sets, then introduce the experimental setup, present the experimental results with statistical test, and finally finish by conducting experimental analysis for our method.

### A. Data Sets

**Office-Home** [18] is a large and popular visual recognition data set in domain adaptation [60], [25], [35]. It consists of 4 domains: Art (**Ar**, 2421 artistic depictions of objects), Clipart (**Cl**, 4379 clipart images), Product (**Pr**, 4428 objects without a background) and RealWorld (**Rw**, 4357 objects captured with

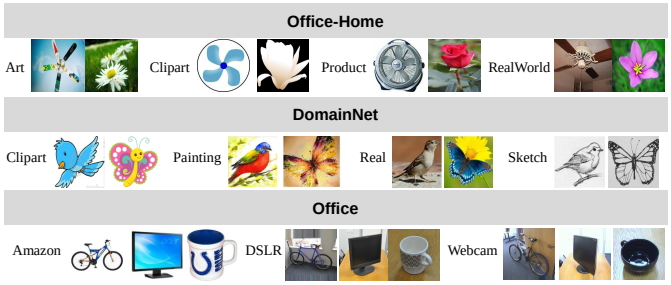



Fig. 4: Image samples from data sets Office-Home, DomainNet, and Office. In different domains, the same category exhibits substantially different visual characteristics.

a regular camera), and 65 categories in each domains. Fig. 4 shows some sample images from this data set.

**DomainNet** [62] is a visual data set recently released for large-scale domain adaptation research. It has 6 domains and 345 classes in total. Since the labels of some domains and classes are very noisy, following the work of Saito *et al*. [3], we select 4 domains and 126 classes from this data set. These domains are Clipart (**Cl**) with 18703 clipart images, Painting (**Pa**) with 31502 artistic depictions of objects in the form of paintings, Real (**Re**) with 70358 photos and real world images, and Sketch (**Sk**) with 24582 sketches of specific objects. See Fig. 4 for some sample images.

**Office** [63] is a classical and widely used data set for domain adaptation in computer vision. It includes images of 31 objects taken from 3 domains: Amazon (**Am**, 2817 images downloaded from amazon.com), DSLR (**Ds**, 498 high-resolution images taken by a digital SLR camera), and Webcam (**We**, 795 low-resolution images recorded by a Web camera). Some sample images from this data set are shown in Fig. 4.

### *B. Experimental Setup*

We compare our solution with the no adaptation deep learning baselines and the deep domain adaptation methods. The former include ResNet [48], VGGNet [47], and AlexNet [46]. The latter are based on several representative domain adaptation strategies: marginal distribution matching, class-conditional distribution matching, and joint distribution matching. In particular, the marginal distribution matching methods contain DAN [26], DANN [1], and ADDA [21]. The class-conditional distribution matching methods consist of MADA [23], CDAN [60], CAT [34], DSAN [35], and GSDA [36]. The joint distribution matching methods include JAN [2] and DeepJDOT [25]. Moreover, we also compare our approach with other important deep methods, including CADA [64], HDAN [65], ADR [66], ENT [67], and MME [3]. The classification accuracy (%) on the unlabeled target data serves as the performance measure for all the methods.

We perform unsupervised domain adaptation experiments on data sets Office-Home, DomainNet, and Office, and semi-supervised experiments on Office-Home and DomainNet. For the unsupervised experiments, following prior works [60], [36], [3], we employ ResNet50 [48] as the backbone for all the deep methods on the Office-Home and Office data sets, and AlexNet [46] on the DomainNet data set. On Office-Home, DomainNet, and Office, we respectively construct 12, 7, and 6 unsupervised domain adaptation tasks, and denote a task as S $\rightarrow$ T with S being a source domain and T a target domain. For the semi-supervised experiments, we adopt the protocol introduced by Saito *et al*. [3], and use VGG16 [47] as the backbone on Office-Home, VGG16 and ResNet34 [48] as the backbone on DomainNet. For each data set, one or three labeled examples per class are randomly selected as labeled target samples, which is termed as the 1-shot or 3-shot setting, another three labeled target examples are used for validation, and the remaining ones serve as the unlabeled target data [3].

We study the RCS divergence with its parameter $\alpha = 0.5$ and $\alpha = 0$, and respectively denote our solution as DNA and $\text{DNA}_0$ under these parameter settings. We employ the minibatch SGD algorithm with a momentum of 0.9 to train our network, where the network parameters are pretrained on ImageNet. Since the ultimate classifier layer is trained from scratch, its learning rate is set to ten times the learning rate of the other layers. Besides, the learning rate annealing strategy in [1], [35] is employed here for training the network. In every iteration of the SGD algorithm, we prepare two minibatches of the same size. One of them consists of the labeled source samples and the other one contains the pseudo labeled target samples. The objective in (14) is computed using these two minibatches. For the tradeoff parameter $\gamma$, following the strategy in [1], [35], we gradually change it from 0 to 1 using the formula $\gamma_p = \frac{2}{1+\exp(-10p)} - 1$, where $p$ is the training progress linearly changing from 0 to 1. According to [1], [35], this strategy can suppress noisy activations at the early stages of the training procedure. For the parameters in the RCS divergence estimate, we set the Gaussian kernel width $\sigma$ to the median squared distance between the minibatch unlabeled source and target samples, and the regularization parameter $\lambda = 10^{-2}$, since preliminary experiments have shown that our solution yields better performance around this value. To ensure fair comparison, other settings regarding the ResNet50 backbone are the same as [60], [35], and the AlexNet, VGG16 and ResNet34 backbones the same as [3].

### *C. Experimental Results with Statistical Test*

We report in Table I and Table II the unsupervised and semi-supervised domain adaptation results on Office-Home, in Table III, Table IV, and Table V the unsupervised and semi-supervised results on DomainNet, and in Table VI the unsupervised results on Office. Note that, since our experimental settings coincide with the prior works, we therefore directly quote the results of the comparison methods in Table I from [35], [36], [68], [65], the results in Table II, Table III, Table IV, and Table V from [3], [65], and the results in Table VI from [34], [35], [36], [68]. In the last row of every table (sub-table), the average accuracy (Avg) over all the tasks is calculated. For every row in the table, the best result is highlighted in **bold**, and the second best is underlined.

From all these 6 tables, we observe that our DNA solution outperforms its competitors on majority of the tasks under both the unsupervised and semi-supervised domain adaptation settings. Additionally, we compare our solution against

TABLE I: Classification accuracy (%) on Office-Home for unsupervised domain adaptation (ResNet50). The best value in each row is highlighted in **bold**. The second best is underlined.

| Task | ResNet50 | DAN | DANN | CDAN | DSAN | GSDA | JAN | DeepJDOT | CADA | HDAN | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ar → Cl | 34.9 | 43.6 | 45.6 | 50.7 | 54.4 | 61.3 | 45.9 | 48.2 | 56.9 | 56.8 | 59.8 | **61.9** |
| Ar → Pr | 50.0 | 57.0 | 59.3 | 70.6 | 70.8 | 76.1 | 61.2 | 69.2 | 76.4 | 75.2 | 79.3 | **79.6** |
| Ar → Rw | 58.0 | 67.9 | 70.1 | 76.0 | 75.4 | 79.4 | 68.9 | 74.5 | 80.7 | 79.8 | 81.5 | **81.8** |
| Cl → Ar | 37.4 | 45.8 | 47.0 | 57.6 | 60.4 | 65.4 | 50.4 | 58.5 | 61.3 | 65.1 | 66.5 | **68.0** |
| Cl → Pr | 41.9 | 56.5 | 58.5 | 70.0 | 67.8 | 73.3 | 59.7 | 69.1 | 75.2 | 73.9 | 77.3 | **79.2** |
| Cl → Rw | 46.2 | 60.4 | 60.9 | 70.0 | 68.0 | 74.3 | 61.0 | 71.1 | 75.2 | 75.2 | 78.1 | **78.9** |
| Pr → Ar | 38.5 | 44.0 | 46.1 | 57.4 | 62.6 | 65.0 | 45.8 | 56.3 | 63.2 | 66.3 | 66.2 | **68.2** |
| Pr → Cl | 31.2 | 43.6 | 43.7 | 50.9 | 55.9 | 53.2 | 43.4 | 46.0 | 54.5 | 56.7 | 55.9 | **56.7** |
| Pr → Rw | 60.4 | 67.7 | 68.5 | 77.3 | 78.5 | 80.0 | 70.3 | 76.5 | 80.7 | 81.8 | 81.9 | **82.7** |
| Rw → Ar | 53.9 | 63.1 | 63.2 | 70.9 | 73.8 | 72.2 | 63.9 | 68.0 | 73.9 | **75.4** | 73.0 | 74.2 |
| Rw → Cl | 41.2 | 51.5 | 51.8 | 56.7 | 60.6 | 60.6 | 52.4 | 52.7 | 61.5 | 59.7 | 61.2 | **63.1** |
| Rw → Pr | 59.9 | 74.3 | 76.8 | 81.6 | 83.1 | 83.1 | 76.8 | 80.9 | 84.1 | 84.7 | 86.0 | **86.9** |
| Avg | 46.1 | 56.3 | 57.6 | 65.8 | 67.6 | 70.3 | 58.3 | 64.3 | 70.3 | 70.9 | 72.2 | **73.4** |

TABLE II: Classification accuracy (%) on Office-Home for semi-supervised domain adaptation (VGG16).

| Setting | Task | VGG16 | DANN | CDAN | ADR | ENT | MME | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|---|
| 1-shot | Ar → Cl | 37.5 | 44.4 | 39.8 | 39.0 | 22.4 | 45.8 | 39.8 | **48.7** |
| | Ar → Pr | 63.6 | 64.3 | 67.7 | 63.9 | 66.0 | 68.6 | 68.7 | **73.8** |
| | Ar → Rw | 69.5 | 68.9 | 64.8 | 68.7 | 70.6 | 72.2 | 70.9 | **75.2** |
| | Cl → Ar | 51.4 | 52.3 | 41.6 | 50.0 | 25.1 | 57.5 | 52.0 | **60.7** |
| | Cl → Pr | 65.9 | 65.3 | 66.2 | 65.2 | 67.7 | 71.3 | 68.6 | **75.2** |
| | Cl → Rw | 64.5 | 64.2 | 58.7 | 64.8 | 62.1 | 68.0 | 67.2 | **73.4** |
| | Pr → Ar | 52.0 | 51.3 | 44.5 | 51.4 | 44.6 | 56.0 | 53.9 | **57.6** |
| | Pr → Cl | 37.0 | 45.9 | 37.4 | 37.2 | 21.3 | 46.2 | 38.3 | **48.5** |
| | Pr → Rw | 71.6 | 72.7 | 69.6 | 71.8 | 74.6 | 74.4 | 76.8 | **78.1** |
| | Rw → Ar | 61.2 | 62.7 | 60.9 | 60.2 | 64.0 | 65.1 | 64.4 | **65.9** |
| | Rw → Cl | 39.5 | **52.0** | 43.3 | 39.7 | 23.7 | 49.1 | 43.1 | 51.9 |
| | Rw → Pr | 75.3 | 75.7 | 75.7 | 76.2 | 77.5 | 78.7 | 81.4 | **81.9** |
| | Avg | 57.4 | 60.0 | 55.9 | 57.3 | 51.6 | 62.7 | 60.4 | **65.9** |
| 3-shot | Ar → Cl | 47.5 | 50.0 | 46.0 | 49.3 | 44.8 | 54.9 | 55.7 | **55.8** |
| | Ar → Pr | 69.4 | 69.5 | 74.7 | 69.9 | 73.0 | 75.7 | 78.3 | **79.2** |
| | Ar → Rw | 73.4 | 72.3 | 71.4 | 73.3 | 75.3 | 75.3 | **78.3** | 77.2 |
| | Cl → Ar | 56.2 | 56.4 | 52.9 | 56.3 | 59.1 | 61.1 | 63.2 | **63.7** |
| | Cl → Pr | 70.4 | 69.8 | 71.2 | 71.4 | 77.0 | 76.3 | 76.7 | **78.7** |
| | Cl → Rw | 69.7 | 68.7 | 65.9 | 69.3 | 72.9 | 72.9 | 74.7 | **75.1** |
| | Pr → Ar | 55.9 | 56.3 | 50.3 | 55.8 | 56.9 | 59.2 | 57.1 | **61.2** |
| | Pr → Cl | 47.2 | 52.4 | 45.1 | 47.8 | 46.8 | 53.6 | 47.1 | **55.6** |
| | Pr → Rw | 72.7 | 73.6 | 70.8 | 73.6 | 76.6 | 76.7 | **78.3** | 78.1 |
| | Rw → Ar | 63.6 | 63.7 | 62.1 | 62.8 | 65.5 | 65.7 | 67.1 | **67.4** |
| | Rw → Cl | 49.6 | 56.1 | 50.2 | 49.0 | 48.3 | **56.9** | 50.3 | 56.8 |
| | Rw → Pr | 78.6 | 77.9 | 80.9 | 78.1 | 81.6 | 82.9 | **85.4** | 84.8 |
| | Avg | 62.9 | 63.9 | 61.8 | 63.1 | 64.8 | 67.6 | 67.7 | **69.4** |

TABLE III: Classification accuracy (%) on DomainNet for unsupervised domain adaptation (AlexNet).

| Task | AlexNet | DANN | CDAN | ADR | ENT | MME | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|
| Re → Cl | 41.1 | 44.7 | 44.2 | 40.2 | 33.8 | 47.6 | 41.5 | **48.1** |
| Re → Pa | 42.6 | 36.1 | 39.1 | 40.1 | 43.0 | 44.7 | 42.5 | **45.2** |
| Pa → Cl | 37.4 | 35.8 | 37.8 | 36.7 | 23.0 | 39.9 | 35.8 | **42.7** |
| Cl → Sk | 30.6 | 33.8 | 26.2 | 29.9 | 22.9 | 34.0 | 30.6 | **36.4** |
| Sk → Pa | 30.0 | **35.9** | 24.8 | 30.6 | 13.9 | 33.0 | 30.7 | 33.0 |
| Re → Sk | 26.3 | 27.6 | 24.3 | 25.9 | 12.0 | 29.0 | 26.4 | **30.4** |
| Pa → Re | 52.3 | 49.3 | 54.6 | 51.5 | 51.2 | 53.3 | 53.2 | **60.9** |
| Avg | 37.2 | 37.6 | 35.9 | 36.4 | 28.5 | 40.2 | 37.3 | **42.4** |

TABLE IV: Classification accuracy (%) on DomainNet for semi-supervised domain adaptation (VGG16).

| Setting | Task | VGG16 | DANN | CDAN | ADR | ENT | MME | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|---|
| 1-shot | Re → Cl | 49.0 | 43.9 | 57.8 | 48.3 | 39.6 | 60.6 | 54.2 | **61.6** |
| | Re → Pa | 55.4 | 42.0 | 57.8 | 54.6 | 43.9 | 63.3 | 59.0 | **65.4** |
| | Pa → Cl | 47.7 | 37.3 | 51.0 | 47.3 | 26.4 | 57.0 | 52.0 | **61.4** |
| | Cl → Sk | 43.9 | 46.7 | 42.5 | 44.0 | 27.0 | 50.9 | 50.1 | **55.6** |
| | Sk → Pa | 50.8 | 51.9 | 51.2 | 50.7 | 29.1 | 60.5 | 56.0 | **62.3** |
| | Re → Sk | 37.9 | 30.2 | 42.6 | 38.6 | 19.3 | 50.2 | 45.0 | **54.8** |
| | Pa → Re | 69.0 | 65.8 | 71.7 | 67.6 | 68.2 | 72.2 | 71.9 | **76.1** |
| | Avg | 50.5 | 45.4 | 53.5 | 50.2 | 36.2 | 59.2 | 55.4 | **62.5** |
| 3-shot | Re → Cl | 52.3 | 56.8 | 58.1 | 50.2 | 50.3 | 64.1 | 57.1 | **65.4** |
| | Re → Pa | 56.7 | 57.5 | 59.1 | 56.1 | 54.6 | 63.5 | 60.1 | **66.1** |
| | Pa → Cl | 51.0 | 49.2 | 57.4 | 51.5 | 47.4 | 60.7 | 56.5 | **63.4** |
| | Cl → Sk | 48.5 | 48.2 | 47.2 | 49.0 | 41.9 | 55.4 | 53.8 | **58.6** |
| | Sk → Pa | 55.1 | 55.6 | 54.5 | 53.5 | 51.0 | 60.9 | 59.0 | **63.5** |
| | Re → Sk | 45.0 | 45.6 | 49.3 | 44.7 | 39.7 | 54.8 | 48.9 | **58.4** |
| | Pa → Re | 71.7 | 70.1 | 74.6 | 70.9 | 72.5 | 75.3 | 73.6 | **77.0** |
| | Avg | 54.3 | 54.7 | 57.2 | 53.7 | 51.1 | 62.1 | 58.4 | **64.6** |

TABLE V: Classification accuracy (%) on DomainNet for semi-supervised domain adaptation (ResNet34).

| Setting | Task | ResNet34 | DANN | CDAN | ADR | ENT | MME | HDAN | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|---|---|
| 1-shot | Re → Cl | 55.6 | 58.2 | 65.0 | 57.1 | 65.2 | 70.0 | 71.7 | 69.6 | **72.3** |
| | Re → Pa | 60.6 | 61.4 | 64.9 | 61.3 | 65.9 | 67.7 | 67.1 | 67.1 | **70.1** |
| | Pa → Cl | 56.8 | 56.3 | 63.7 | 57.0 | 65.4 | 69.0 | 72.8 | 66.4 | **73.0** |
| | Cl →Sk | 50.8 | 52.8 | 53.1 | 51.0 | 54.6 | 56.3 | 63.7 | 57.2 | **64.7** |
| | Sk → Pa | 56.0 | 57.4 | 63.4 | 56.0 | 59.7 | 64.8 | 65.7 | 62.7 | **68.3** |
| | Re → Sk | 46.3 | 52.2 | 54.5 | 49.0 | 52.1 | 61.0 | **69.2** | 55.9 | 64.4 |
| | Pa → Re | 71.8 | 70.3 | 73.2 | 72.0 | 75.0 | 76.1 | 76.6 | 78.4 | **80.3** |
| | Avg | 56.8 | 58.4 | 62.5 | 57.6 | 62.6 | 66.4 | 69.5 | 65.3 | **70.4** |
| 3-shot | Re → Cl | 60.0 | 59.8 | 69.0 | 60.7 | 71.0 | 72.2 | 73.9 | 70.2 | **74.2** |
| | Re → Pa | 62.2 | 62.8 | 67.3 | 61.9 | 69.2 | 69.7 | 69.1 | 69.0 | **71.1** |
| | Pa → Cl | 59.4 | 59.6 | 68.4 | 60.7 | 71.1 | 71.7 | 73.0 | 69.6 | **74.1** |
| | Cl → Sk | 55.0 | 55.4 | 57.8 | 54.4 | 60.0 | 61.8 | **66.3** | 60.5 | 66.0 |
| | Sk → Pa | 59.5 | 59.9 | 65.3 | 59.9 | 62.1 | 66.8 | 67.5 | 66.0 | **68.7** |
| | Re → Sk | 50.1 | 54.9 | 59.0 | 51.1 | 61.1 | 61.9 | **69.5** | 58.7 | 65.0 |
| | Pa → Re | 73.9 | 72.2 | 78.5 | 74.2 | 78.6 | 78.5 | 79.7 | 79.9 | **81.7** |
| | Avg | 60.0 | 60.7 | 66.5 | 60.4 | 67.6 | 68.9 | 71.3 | 67.7 | **71.5** |

more state-of-the-art domain adaptation methods including CCSA [56], FADA [57], and PL-NEM [59]. Following the experimental settings in the work of Wang *et al*. [59], we conduct the semi-supervised domain adaptation experiments on the related Office-Caltech data set [58] with 4 domains Amazon (**Am**), Caltech (**Ca**), DSLR (**Ds**), and Webcam (**We**), and report the experimental results in Table VII, where part of the results are cited from [59]. As before, we observe that our DNA solution yields better classification accuracy and outperforms the other methods on 11 out of the total 12 tasks.

To be strict in a statistical sense, we further conduct statistical test to check our solution is significantly better than the others. We first conduct the Wilcoxon signed-ranks test [69], [8], [12] based on the unsupervised domain adaptation results from Table I, Table III, and Table VI. The test uses a statistic $z$ to compare the performance of two methods over multiple tasks. Specifically, in each task the classification accuracy is adopted as the performance measure of the methods. We fix DNA as a control method, and conduct 8 pairs of tests: DAN versus DNA, DANN versus DNA, CDAN versus DNA, DSAN versus DNA, GSDA versus DNA, JAN versus DNA, DeepJDOT versus DNA, and $DNA_0$ versus DNA. The detailed description of the test procedure is presented in the supplementary material, and the resulting 8 $z$ values are reported in Table VIII. We observe from Table VIII that the $z$ values for these 8 pairs are all below the critical value -1.96. According to

TABLE VI: Classification accuracy (%) on Office for unsupervised domain adaptation (ResNet50).

| Task | ResNet50 | DAN | DANN | ADDA | MADA | CDAN | CAT | DSAN | GSDA | JAN | DeepJDOT | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Am → Ds | 68.9 | 78.6 | 79.7 | 77.8 | 87.8 | 92.9 | 90.6 | 90.2 | **94.8** | 84.7 | 88.2 | 88.8 | 90.4 |
| Am → We | 68.4 | 80.5 | 82.0 | 86.2 | 90.0 | 94.1 | 91.1 | 93.6 | **95.7** | 85.4 | 88.9 | 86.7 | 88.6 |
| Ds → Am | 62.5 | 63.6 | 68.2 | 69.5 | 70.3 | 71.0 | 70.4 | 73.5 | 73.5 | 68.6 | 72.1 | **77.5** | 77.1 |
| Ds → We | 96.7 | 97.1 | 96.9 | 96.2 | 97.4 | 98.6 | 98.6 | 98.3 | 99.1 | 97.4 | 98.5 | **99.2** | **99.2** |
| We → Am | 60.7 | 62.8 | 67.4 | 68.9 | 66.4 | 69.3 | 66.5 | 74.8 | 74.9 | 70.0 | 70.1 | 76.4 | **76.6** |
| We → Ds | 99.3 | 99.6 | 99.1 | 98.4 | 99.6 | **100.0** | 99.6 | **100.0** | **100.0** | 99.8 | 99.6 | **100.0** | **100.0** |
| Avg | 76.1 | 80.4 | 82.2 | 82.8 | 85.3 | 87.7 | 86.1 | 88.4 | **89.7** | 84.3 | 86.2 | 88.1 | 88.7 |

TABLE VII: Classification accuracy (%) on Office-Caltech for semi-supervised domain adaptation.

| Task | DANN | CCSA | FADA | PL-NEM | $DNA_0$ | DNA |
|---|---|---|---|---|---|---|
| Am → Ca | 81.9 | 83.7 | 82.6 | 86.0 | 86.4 | **87.3** |
| Am → Ds | 79.5 | 97.2 | 96.5 | 98.3 | 97.6 | **98.5** |
| Am → We | 75.5 | 94.5 | 95.1 | 97.5 | 98.1 | **98.7** |
| Ca → Am | 84.1 | 90.2 | 91.4 | 92.6 | 92.8 | **94.0** |
| Ca → Ds | 76.7 | 91.7 | 91.3 | 94.1 | 94.5 | **95.8** |
| Ca → We | 66.9 | 89.7 | 90.2 | 97.5 | 97.0 | **98.6** |
| Ds → Am | 78.2 | 91.7 | 90.4 | 94.3 | 95.2 | **96.3** |
| Ds → Ca | 73.8 | 80.8 | 80.2 | 87.1 | 88.4 | **88.8** |
| Ds → We | 97.2 | 98.7 | 97.6 | **99.3** | 98.9 | 99.1 |
| We → Am | 71.9 | 91.2 | 90.7 | 94.0 | 94.7 | **95.6** |
| We → Ca | 65.9 | 77.8 | 76.5 | 86.9 | 87.1 | **87.9** |
| We → Ds | 96.3 | 99.6 | 99.3 | 99.5 | **100.0** | **100.0** |
| Avg | 79.0 | 90.6 | 90.2 | 93.9 | 94.2 | **95.1** |

TABLE VIII: $z$ values of different methods versus DNA on the unsupervised domain adaptation tasks.

| Statistic | DAN | DANN | CDAN | DSAN | GSDA | JAN | DeepJDOT | $DNA_0$ |
|---|---|---|---|---|---|---|---|---|
| $z$ | -3.72 | -4.23 | -4.03 | -3.22 | -2.26 | -3.72 | -3.68 | -4.17 |

[69], [8], [12], this indicates that with a significance level 0.05, DNA is statistically better than the other methods under the unsupervised domain adaptation setting. Subsequently, for the semi-supervised results from Table II, Table IV and Table V, we repeat the Wilcoxon signed-ranks test to check whether DNA is statistically better than DANN, CDAN, ADR, ENT, MME, and $DNA_0$ under the semi-supervised setting. The test again shows that in a statistical sense, our DNA solution performs better than its competitors for semi-supervised domain adaptation.

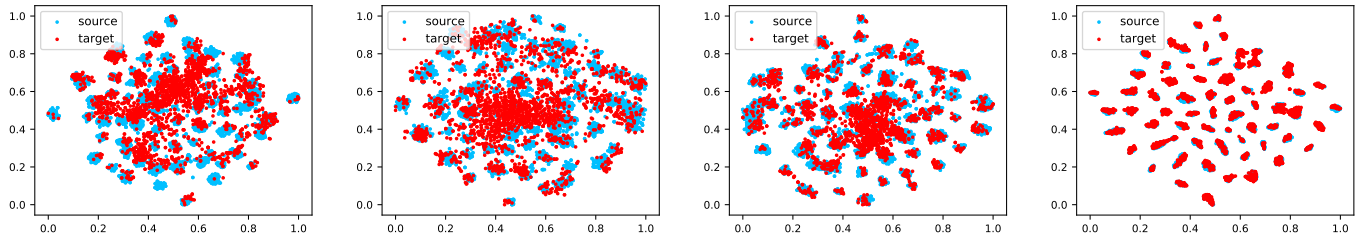
Based on the results of the statistical test, we make the following conclusions. **(1)** For unsupervised and semi-supervised domain adaptation on large and real object recognition data sets, our DNA solution is advantageous over the other deep domain adaptation approaches that are based on marginal distribution matching, class-conditional distribution matching, or other strategies. This resembles the findings in the shallow counterpart of this work [7], and confirms that in the deep neural network context, directly matching the source and target joint distributions can better address the fundamental joint distribution mismatch problem in domain adaptation and consequently fulfill the goal of learning a well-performed target predictor. **(2)** On the tested evaluations, our joint distribution matching solution under the RCS divergence yields better results than the one under the Wassertein distance [25]. To some extent, this indicates that distribution similarity metric matters in domain adaptation and that the RCS divergence is beneficial to comparing distributions in this problem. Furthermore, this also verifies that our divergence approximation method is empirically sound and produces an estimate that well reflects the discrepancy between joint distributions. **(3)** Overall, setting $\alpha = 0.5$ in the RCS divergence can lead to better domain adaptation results than setting $\alpha = 0$, which coincides with the findings in the work [40] that introduces this divergence. We conjecture that this is due to the fact that the RCS divergence with $\alpha = 0.5$ is bounded. With this characteristic, when training our DNA network the divergence term will not dominate the classification loss and result in a negative impact on the discriminant structure of the data.

### D. Experimental Analysis

*1) Feature Visualization:* We exploit the t-SNE visualization tool [70] and visualize in Fig. 5(a)-Fig. 5(d) the network activations of task Cl→Ar generated by ResNet50, DAN, DeepJDOT, and DNA. Comparing Fig. 5(d) against Fig. 5(a), Fig. 5(b), and Fig. 5(c), we observe that our DNA well aligns the source (in blue) and target (in red) data in the network activation space, and the alignment is not only significantly better than the one of the no adaptation network ResNet50, but also superior to the ones of the domain adaptation networks DAN and DeepJDOT. These comparison results suggest that in the deep neural network context, joint distribution matching under the RCS divergence is a powerful approach to domain adaptation.

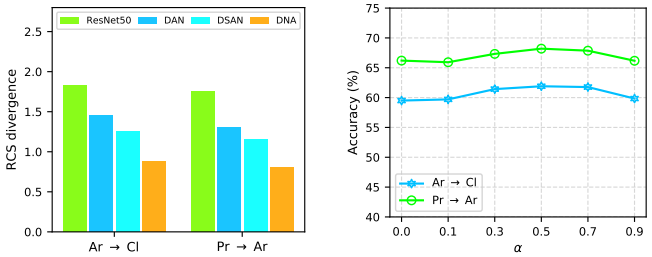
*2) RCS Divergence between Joint Distributions:* We compute the RCS divergence between the source and target joint distributions via Eq. (12) using the activation features from ResNet50, DAN, DSAN, and DNA, and plot in Fig. 6(a) the resulting divergence values on tasks Ar→Cl and Pr→Ar. Clearly, on both tasks the RCS divergence values obtained from our DNA activation features are always smaller than those from the ResNet50, DAN, or DSAN activation features. As implied by Theorem 1, with a smaller RCS divergence and a small source error, our solution thus leads to better classification results in the target domain.

*3) Parameter $\alpha$ of the RCS Divergence:* We study the sensitivity of our solution with respect to different choices of parameter $\alpha$ in the RCS divergence, and plot in Fig. 6(b) the classification accuracy of our solution by varying $\alpha \in \{0.0, 0.1, 0.3, 0.5, 0.7, 0.9\}$ on tasks Ar→Cl and Pr→Ar. From Fig. 6(b), we observe that on the tested evaluations, as the value of $\alpha$ grows, the accuracy of our solution first increases, reaching its peak at $\alpha = 0.5$, and then begins to decrease. According to these sensitivity results, we recommend selecting




(a) ResNet50 (b) DAN (c) DeepJDOT (d) DNA

Fig. 5: T-SNE visualization of the network activations from ResNet50, DAN, DeepJDOT, and DNA on the task Cl→Ar, where the source samples are colored in blue and the target ones in red.




(a) (b)

Fig. 6: Analysis of the RCS divergence. (a) RCS divergence between the source and target joint distributions. (b) Classification accuracy of our solution with respect to different values of parameter $\alpha$ in the RCS divergence.

parameter $\alpha = 0.5$ for better application of the RCS divergence in our deep domain adaptation solution.

## VI. Conclusion

We devise a deep domain adaptation method named DNA, which exploits the neural network mapping to match the source and target joint distributions for addressing the fundamental problem in unsupervised and semi-supervised domain adaptation. The method compares the source and target joint distributions under the RCS divergence. We show that this divergence can be directly approximated from samples via solving an unconstrained quadratic maximization problem, leading to a nice analytic solution. By simultaneously minimizing the RCS divergence between joint distributions and the negative log-likelihood loss for classification, we obtain a neural network classifier that generalizes well to the target domain. In our experiments on large and real object recognition data sets, we find that the proposed DNA solution is statistically better than the reference methods under both the unsupervised and semi-supervised domain adaptation settings. In the future, we intend to extend the ideas in this work to tackle other related problems including domain generalization and multi-task learning.

## Acknowledgment

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

## References

[1] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research*, vol. 17, no. 59, pp. 1–35, 2016.

[2] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *International Conference on Machine Learning*, 2017, pp. 2208–2217.

[3] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko, "Semi-supervised domain adaptation via minimax entropy," in *IEEE International Conference on Computer Vision*, 2019, pp. 8049–8057.

[4] A.-J. Gallego, J. Calvo-Zaragoza, and R. B. Fisher, "Incremental unsupervised domain-adversarial training of neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 11, pp. 4864–4878, 2021.

[5] S. Li, F. Lv, B. Xie, C. H. Liu, J. Liang, and C. Qin, "Bi-classifier determinacy maximization for unsupervised domain adaptation," in *AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 8455–8464.

[6] L. Li and Z. Zhang, "Semi-supervised domain adaptation by covariance matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2724–2739, 2019.

[7] S. Chen, M. Harandi, X. Jin, and X. Yang, "Semi-supervised domain adaptation via asymmetric joint distribution matching," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5708–5722, 2021.

[8] S. Chen, L. Han, X. Liu, Z. He, and X. Yang, "Subspace distribution adaptation frameworks for domain adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 12, pp. 5204–5218, 2020.

[9] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "CyCADA: Cycle-consistent adversarial domain adaptation," in *International Conference on Machine Learning*, vol. 80, 10–15 Jul 2018, pp. 1989–1998.

[10] S. Zhao, X. Yue, S. Zhang, B. Li, H. Zhao, B. Wu, R. Krishna, J. E. Gonzalez, A. L. Sangiovanni-Vincentelli, S. A. Seshia, and K. Keutzer, "A review of single-source deep unsupervised visual domain adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 473–493, 2022.

[11] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1414–1430, July 2017.

[12] S. Chen, M. Harandi, X. Jin, and X. Yang, "Domain adaptation by joint distribution invariant projections," *IEEE Transactions on Image Processing*, vol. 29, pp. 8264–8277, 2020.

[13] V. N. Vapnik, *Statistical Learning Theory*. John Wiley & Sons, 1998.

[14] J. Jiang, "A literature survey on domain adaptation of statistical classifiers," *URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey*, vol. 3, pp. 1–12, 2008.

[15] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, "Domain adaptation under target and conditional shift," in *International Conference on Machine Learning*, 2013, pp. 819–827.

[16] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann, "Unsupervised domain adaptation by domain invariant projection," in *IEEE International Conference on Computer Vision*, Dec 2013, pp. 769–776.

[17] S. Cicek and S. Soatto, "Unsupervised domain adaptation via regularized conditional alignment," in *IEEE International Conference on Computer Vision*, 2019, pp. 1416–1425.

[18] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5385–5394.

[19] S. Si, D. Tao, and B. Geng, "Bregman divergence-based regularization for transfer subspace learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 7, pp. 929–942, July 2010.

[20] M. Baktashmotlagh, M. Harandi, and M. Salzmann, "Distribution-matching embedding for visual domain adaptation," *Journal of Machine Learning Research*, vol. 17, no. 108, pp. 1–30, 2016.

[21] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[22] B. Quanz, J. Huan, and M. Mishra, "Knowledge transfer with low-quality data: A feature extraction issue," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 10, pp. 1789–1802, Oct 2012.

[23] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *AAAI Conference on Artificial Intelligence*, 2018, pp. 3934–3941.

[24] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," in *Advances in Neural Information Processing Systems*, 2017, pp. 3730–3739.

[25] B. Bhushan Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, "Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation," in *European Conference on Computer Vision*, 2018, pp. 447–463.

[26] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 3071–3085, 2019.

[27] Q. Kang, S. Yao, M. Zhou, K. Zhang, and A. Abusorrah, "Effective visual domain adaptation via generative adversarial distribution matching," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 9, pp. 3919–3929, 2021.

[28] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet, "Hilbert space embeddings and metrics on probability measures," *Journal of Machine Learning Research*, vol. 11, pp. 1517–1561, 2010.

[29] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[30] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1853–1865, Sep. 2017.

[31] M. Baktashmotlagh, M. Harandi, and M. Salzmann, "Learning domain invariant embeddings by matching distributions," in *Domain adaptation in computer vision applications*. Springer, 2017, pp. 95–114.

[32] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1, pp. 151–175, May 2010.

[33] M. Gong, K. Zhang, T. Liu, D. Tao, C. Glymour, and B. Schölkopf, "Domain adaptation with conditional transferable components," in *International Conference on Machine Learning*, 2016, pp. 2839–2848.

[34] Z. Deng, Y. Luo, and J. Zhu, "Cluster alignment with a teacher for unsupervised domain adaptation," in *IEEE International Conference on Computer Vision*, 2019, pp. 9944–9953.

[35] Y. Zhu, F. Zhuang, J. Wang, G. Ke, J. Chen, J. Bian, H. Xiong, and Q. He, "Deep subdomain adaptation network for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1713–1722, 2021.

[36] L. Hu, M. Kan, S. Shan, and X. Chen, "Unsupervised domain adaptation with hierarchical gradient synchronization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4043–4052.

[37] J. Zhang, W. Li, and P. Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp. 5150–5158.

[38] S. Li, S. Song, G. Huang, Z. Ding, and C. Wu, "Domain invariant and class discriminative feature learning for visual domain adaptation," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4260–4273, Sep. 2018.

[39] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *IEEE International Conference on Computer Vision*, Dec 2013, pp. 2200–2207.

[40] M. Yamada, T. Suzuki, T. Kanamori, H. Hachiya, and M. Sugiyama, "Relative density-ratio estimation for robust distribution comparison," *Neural Computation*, vol. 25, no. 5, pp. 1324–1370, 2013.

[41] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Networks*, vol. 43, pp. 72 – 83, 2013.

[42] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "On the effectiveness of least squares generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 2947–2960, Dec 2019.

[43] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[44] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, pp. 463–482, 2002.

[45] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2012.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2014, pp. 1–14.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[49] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 766–785, 2021.

[50] Y. Chen, C. Wei, A. Kumar, and T. Ma, "Self-training avoids using spurious features under domain shift," in *Advances in Neural Information Processing Systems*, 2020, pp. 21 061–21 071.

[51] M. Long, J. Wang, G. Ding, S. J. Pan, and P. S. Yu, "Adaptation regularization: A general framework for transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1076–1089, 2014.

[52] J. Li, M. Jing, K. Lu, L. Zhu, and H. T. Shen, "Locality preserving joint transfer for domain adaptation," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 6103–6115, 2019.

[53] S. Li, C. H. Liu, L. Su, B. Xie, Z. Ding, C. L. P. Chen, and D. Wu, "Discriminative transfer feature and label consistency for cross-domain image classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4842–4856, 2020.

[54] S. Li, C. H. Liu, Q. Lin, Q. Wen, L. Su, G. Huang, and Z. Ding, "Deep residual correction network for partial domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2329–2344, 2021.

[55] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European Conference on Computer Vision Workshops*, 2016, pp. 443–450.

[56] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *IEEE International Conference on Computer Vision*, 2017, pp. 5716–5726.

[57] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto, "Few-shot adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2017, pp. 6670–6680.

[58] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 2066–2073.

[59] Z. Wang, B. Du, and Y. Guo, "Domain adaptation with neural embedding matching," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2387–2397, 2020.

[60] M. Long, Z. Cao, J. Wang, and M. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2018, pp. 1640–1650.

[61] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," in *Advances in Neural Information Processing Systems*, 2016, pp. 271–279.

[62] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *IEEE International Conference on Computer Vision*, 2019, pp. 1406–1415.

[63] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *European Conference on Computer Vision*, 2010, pp. 213–226.

[64] V. Kurmi, S. Kumar, and V. Namboodiri, "Attending to discriminative certainty for domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 491–500.

[65] S. Cui, X. Jin, S. Wang, Y. He, and Q. Huang, "Heuristic domain adaptation," in *Advances in Neural Information Processing Systems*, 2020, pp. 7571–7583.
[66] K. Saito, Y. Ushiku, T. Harada, and K. Saenko, "Adversarial dropout regularization," in *International Conference on Learning Representations*, 2018, pp. 1–15.
[67] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Advances in Neural Information Processing Systems*, 2005, pp. 529–536.
[68] R. Xu, P. Liu, L. Wang, C. Chen, and J. Wang, "Reliable weighted optimal transport for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4394–4403.
[69] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.
[70] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.