Unsupervised Domain Adaptation with Sphere Retracting Transformation

[1st] Zhen Fang
Centre for Artificial Intelligence
FEIT, Univervisity of Technology Sydney
Sydney, Australia
Zhen.Fang@student.uts.edu.au

[2nd] Jie Lu
Centre for Artificial Intelligence
FEIT, Univervisity of Technology Sydney
Sydney, Australia
Jie.Lu@uts.edu.au

[3rd] Feng Liu
Centre for Artificial Intelligence
FEIT, Univervisity of Technology Sydney
Sydney, Australia
Feng.Liu-2@student.uts.edu.au

[4th] Guangquan Zhang
Centre for Artificial Intelligence
FEIT, Univervisity of Technology Sydney
Sydney, Australia
Guangquan.Zhang@uts.edu.au

***Abstract*—Unsupervised domain adaptation aims to leverage the knowledge in training data (source domain) to improve the performance of tasks in the remaining unlabeled data (target domain) by mitigating the effect of the distribution discrepancy. Existing approaches resolve this problem mainly by 1) mapping data into a latent space where the distribution discrepancy between two domains is reduced; or 2) reducing the domain shift by weighting the source domain. However, most of these approaches share a common issue that they neglect inter-class margins while matching distributions, which has a significant impact on classification performance. In this paper, we analyze the issue from the theoretical aspect and propose a novel unsupervised domain adaptation approach: Sphere Retracting Transformation (SRT), which reduces the distribution discrepancy and increases inter-class margins. We implement SRT, according to our theoretical analysis by (1) assigning class-specific weights for data in the source domain, and (2) minimizing the intra-class variations. Experiments confirm that the SRT approach outperforms several competitive approaches for standard domain adaptation benchmarks.**

***Index Terms*—unsupervised domain adaptation, maximum mean discrepancy, instance weighting, feature matching**

I. Introduction

The dramatic successes of standard supervised learning machines derive in large part from the availability of abundant labeled datasets. However, manual labeling is a time-consuming, costly process and thus prohibitive. To reduce the cost of manual labeling, it is important to develop a learning algorithm that leverages rich labeled data from the source domain to the target domain. To address this problem, domain adaptation [1], [2] was proposed to link two related domains with different distributions ($\mathcal{P}_s \neq \mathcal{P}_t$) [3]. Unsupervised domain adaptation approaches transfer the related knowledge from the source domain, which has abundant labeled data, to an unlabeled domain (target domain). In this paper, we focus on the homogeneous unsupervised domain adaptation problem [4], where the source and target domain share the same feature space but have different distributions, and the labeling functions f_s and f_t for the source domain and the target domain are similar.

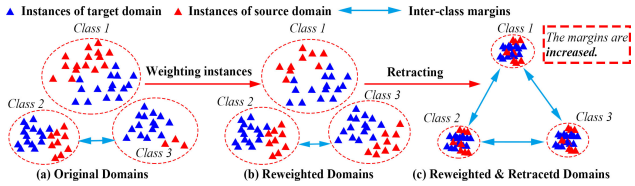

Fig. 1. Weighting source instances according to a class-specific weighting strategy and increasing the inter-class margins by reducing the intra-class variations.

A main problem of unsupervised domain adaptation is how to reduce the discrepancy between the distributions of the source and target domains. The existing work falls into two main categories: (1) feature matching, which seeks a new feature space in which the marginal distributions or conditional distributions from two domains are similar [5]–[7], or performs subspace alignment by exploiting subspace geometrical structure or statistical properties to reduce distribution discrepancy [8]–[11]; (2) instance reweighting, which estimates the weights of the source domain so that the distribution discrepancy can be minimized [12]–[16]. Although many feature matching approaches address the domain adaptation tasks well, those approaches [2], [17], [18] only focus on constructing a latent space where the discrepancy between distributions of two domains is minimized. They overlook the inter-class margins, which could lead an issue that instances belonging to different classes are disordered in the constructed latent space.

In this paper, we analyze the issue in theory and then design an approach to address the issue according to our theoretical analysis. We first discover and prove that the distribution discrepancy between source and target domains depends on

the intra-class variations and weights of the data in source domain. This indicates that reducing the intra-class variations and weighting the data in the source domain not only increases the margins between classes but also reduces the distribution discrepancy between the two domains. Using this discovery, we propose a novel unsupervised domain adaptation approach, referred to as Sphere Retracting Transformation (SRT), by minimizing the intra-class variations with a new retracting sphere transformation and weighting the data in the source domain using a class-specific weighting strategy. Considering the curse of dimensionality, a new spherical dimension reduction is developed to preserve the geometric information of the data. The main contributions of this paper are as follows:

- A theorem is proven and shows that the discrepancy between two distributions can be reduced by reducing the intra-class variations while assigning class-specific weights for the source domain.
- This paper presents a novel domain adaptation approach, Sphere Retracting Transformation (SRT), which matches distributions by jointly reducing intra-class variations and weighting the source domain. Extensive experiments demonstrate that SRT outperforms several competitive domain adaptation approaches.

II. Related Work

We review the previous literatures [18]–[21] and roughly separate the domain adaptation approaches into two categories: feature matching and instance reweighting.

Feature matching aims to reduce the distribution discrepancy by learning a new feature representation. The feature matching method can be summarized as: (1) Transforming data into a new space, where distance measures are minimized. Transfer component analysis (TCA) [22] learns a new feature space to match distributions by employing the Maximum Mean Discrepancy (MMD) [23]. Joint distribution adaptation (JDA) [17] improves TCA by jointly matching marginal distributions and conditional distributions. Scatter component analysis (SCA) [24] extends TCA and JDA, and considers the between and within class scatter. Recent advances show that deep networks can be successfully applied to domain adaptation tasks. Domain Adaptive Neural Networks (DaNN) [25] adds an adaptation layer in neural networks to reduce the distribution discrepancy. Deep Adaptation Networks (DAN) [26] considers three adaptation layers for matching distributions and applies multiple kernels (MK-MMD) [27] for adapting deep representations. Wasserstein Distance Guided Representation Learning (WDGRL) [28] minimizes the distribution discrepancy by employing Wasserstein Distance in neural networks. However, these approaches have a strong assumption that there is a unified transformation to map the source domain and target domain into a common space in which the feature representations are domain invariant. (2) Extracting intermediate features to minimize distribution discrepancy by projecting data onto subspaces or transforming a source subspace into a new subspace, which is related to the target subspace. Geodesic Flow Kernel (GFK) [8] considers the intermediate information in geodesic curve between the source domain and target domain on a Grassmann manifold. Subspace alignment (SA) [9] maps a source PCA subspace into a new subspace which is well-aligned with the target subspace. Correlation Alignment (CORAL) [10] matches the covariance matrix of the source subspace and target subspace. However, the subspace alignment approaches could fail to match the feature distributions when the domain discrepancy is substantially large.

The instance reweighting approach reduces data bias by weighting the source data. The Kernel mean matching (KMM) [12] defines the weights as the density ratio between the source domain and the target domain. Moreover, Mohri [13] and Yu and Szepesvári [14] provided theoretical analysis for important instance reweighting approaches. However, when the cross-domain discrepancy is substantially large, a large number of effective source data will be down-weighted, resulting in the loss of effective information.

The most similar works to our proposed SRT approach are Transfer Joint Matching (TJM) [18] and Distribution Matching Machines (DMM) [21], which both perform feature matching and instance reweighting. However, SRT clearly contrasts with TJM and DMM in two aspects: (1) SRT attempts to reduce the intra-class variations instead of directly matching the distributions; (2) the weights in SRT are class-specific and are determined before training.

III. Sphere Retracting Transformation

In unsupervised domain adaptation, we are given a source domain $\mathcal{X}_s = \{(x_i, y_i)\}_{i=1}^{n_s}$ with n_s labeled instances, and a target domain $\mathcal{X}_t = \{x_j\}_{j=1}^{n_t}$ with n_t unlabeled instances, where the source domain and target domain have different distributions $\mathcal{P}_s$ and $\mathcal{P}_t$. In this paper, we propose a new approach $-$ Sphere Retracting Transformation (SRT), based on our theoretical analysis that the distribution difference can be reduced by (1) minimizing the intra-class variations, and (2) weighting the source domain with class-specific weights. Our approach also guarantees that the inter-class margins are increased. SRT is based on three assumptions:

(1) the labeling function $\mathcal{P}_s(\mathcal{Y}|x) = \mathcal{P}_t(\mathcal{Y}|x)$ $(f_s = f_t)$;

(2)$\mathcal{P}_s(\mathcal{Y} = c|x) \propto 1/\|x - \frac{1}{n_s^c}\sum_{j=1}^{n_s} x_{s,j}^c\|$;

(3) $\frac{1}{n_s^k}\sum_{j=1}^{n_s^k} x_{s,j}^k \neq \frac{1}{n_s^l}\sum_{j=1}^{n_s^l} x_{s,j}^l$, for $k \neq l$, $k, l \in \mathcal{Y}$.

Assumption 1 [22], [29], [30] is the basic assumption in homogeneous unsupervised domain adaptation setting . Assumption (2) implies that the class of an instance x has a higher probability of being c if x is closer to the center of class c. Assumption (3) indicates that each class for the source domain has a different class center.

This section first presents theoretical analysis to show that the distribution discrepancy of related domains can be minimized by (1) assigning class-specific weights for the data in the source domain, and (2) minimizing the intra-class variations. Next, according to our theoretical analysis, we design a loss for seeking a transformation T to simultaneously (1) minimize

the distribution discrepancy, and (2) increase the margins between classes. We then propose an approach for dimensionality reduction to avoid the curse of dimensionality. Last, we employ neural networks to minimize our loss and obtain the best transformation T. Notions and their descriptions are summarized in Table I.

TABLE I
NOTATIONS AND THEIR DESCRIPTIONS.

Notation	Description
$\mathcal{X}_s, \mathcal{X}_t$	source/target domain
n_s, n_t	number of source/target instances
$n+1$	the feature dimension
$\mathcal{Y}$	C-cardinality label set $\{1, \cdots, C\}$
$\mathcal{P}_s, \mathcal{P}_t$	probability distribution for source/target domain
f_s, f_t	labeling function for source/target domain
$\mathcal{X}_s^c$	sub-domain $\{x_{s,j}^c\}_{j=1}$, instances in $\mathcal{X}_s$ with label c
$\widehat{\mathcal{X}}_t^c$	sub-domain $\{x_{t,j}^c\}_{j=1}$, instances in $\mathcal{X}_t$ with pseudo label c
n_s^c	number of instances in $\mathcal{X}_s^c$
$\widehat{n}_t^c$	number of instances in $\widehat{\mathcal{X}}_t^c$
X_s^c, X_t^c	data matrices $\mathcal{X}_s^c$, $\widehat{\mathcal{X}}_t^c$ with size $(n_s^c, n+1)$,$(\widehat{n}_t^c, n+1)$
X_s, X_t	$X_s = [X_s^1; \cdots; X_s^C]$ and $X_t = [X_t^1; \cdots; X_t^C]$
k	the feature dimension after dimensionality reduction
$\widetilde{X}_s^c, \widetilde{X}_t^c$	data matrix for $\mathcal{X}_s^c$, $\widehat{\mathcal{X}}_t^c$ after dimensionality reduction
$\widetilde{X}_s, \widetilde{X}_t$	$\widetilde{X}_s = [\widetilde{X}_s^1; \cdots; \widetilde{X}_s^C]$ and $\widetilde{X}_t = [\widetilde{X}_t^1; \cdots; \widetilde{X}_t^C]$
$\|\cdot\|$	l_2 norm
$\|\cdot\|_F$	Frobenius norm
S^n	the unit sphere $\{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$
$\varphi(\cdot), k(\cdot,\cdot)$	kernel feature map and kernel function induced by $\varphi(\cdot)$

A. Theoretical Analysis

In this section, we prove that the distribution discrepancy can be reduced by (1) assigning class-specific weights for the data in the source domain, and (2) minimizing the intra-class variations. We first introduce a **class-specific weighting strategy**.

The weights are defined as follows: for arbitrary $c \in \mathcal{Y}$, the weight W_c is defined in the c class $f_s^{-1}(c)$ and W_c is

$$\frac{\mathcal{P}_t(f_t^{-1}(c))}{\mathcal{P}_s(f_s^{-1}(c))}. \tag{1}$$

We denote $W\mathcal{P}_s$ as the source distribution after instance reweighting.

for arbitrary $c \in \mathcal{Y}$, the weight W_c is defined in the c class $f_s^{-1}(c)$ and W_c is

$$\frac{Q(f_t^{-1}(c))}{P(f_s^{-1}(c))}. \tag{2}$$

To estimate the domain discrepancy, we use Maximum Mean Discrepancy (MMD) [23], which is an effective non-parametric distance. Let H be the reproducing kernel Hilbert space induced by a kernel function $k = \langle \varphi, \varphi \rangle$, where $\varphi(\cdot)$ is a nonlinear feature map. Given two distributions $\mathcal{P}$ and $\mathcal{Q}$, the MMD distance between $\mathcal{P}$ and $\mathcal{Q}$ is defined as:

$$d_{MMD}(\mathcal{P}, \mathcal{Q}) = \Big\| \int \varphi d\mathcal{P} - \int \varphi d\mathcal{Q} \Big\|_H. \tag{3}$$

We employ the l_2 distance to compute the intra-class variation for class c:

$$\max_{x,y \in f_s^{-1}(c)} \|x - y\|. \tag{4}$$

Theorem 1. *Assume the MMD kernel map φ is a Lipschitz function with a Lipschitz constant L, and $f_s = f_t$, if there is a transformation T to make the l_2 distance for every class $T(f_s^{-1}(c)), c \in \mathcal{Y}$, is smaller than $\epsilon > 0$, then*

$$d_{MMD}(T(W\mathcal{P}_s), T(\mathcal{P}_t)) \leqslant 2L\epsilon. \tag{5}$$

The proof is given in the Appendix.

Theorem 1 tells us that how much the distribution discrepancy can be reduced depends on how small the intra-class variations are for the source domain. According to this theorem, therefore, it is the key to seek a transformation T to minimize the intra-class variations for the source domain. We call this transformation T as the retracting transformation.

B. Model

In this section, we design our loss based on Theorem 1.

1) Retracting Loss: The retracting transformation T encourages tight clusters for the source data in the same class. When the assumption $f_s = f_t$ is satisfied and the intra-class variations in the source domain are small, the intra-class variations in the target domain are also small. We also require that the retracting transformation T maps the data into a new feature space, where the instances in the same class for the target domain are close.

Based on this objective, we design a loss to learn the retracting transformation T so that the data of the same class in the source domain and the target domain will be mapped into a fixed point α_c:

$$\sum_{c=1}^{C} \Big(\sum_{j=1}^{n_s^c} \frac{\|T(x_{s,j}^c) - \alpha_c\|^p}{n_s^c} + \sum_{j=1}^{\widehat{n}_t^c} \frac{\|T(x_{t,j}^c) - \alpha_c\|^p}{\widehat{n}_t^c} \Big), \tag{6}$$

for $c = 1 \cdots, C$, and $p \geqslant 1$. We set $\alpha_c = \frac{1}{n_s^c} \sum_{j=1}^{n_s^c} x_{s,j}^c$ based on our assumption (2). If we assume T is related to a parameter θ, then formula (6) is a nonlinear function related to θ. Gradient descent is used to optimize formula (6). It is desirable, according to assumption (2), that the data $T(x_{s,j}^c)$ should converge to α_c quickly. Therefore, we set $p = 1$, since

$$\lim_{T(x_{s,j}^c) \to \alpha_c} \frac{|\frac{\mathrm{d}}{\mathrm{d}\theta} \|T(x_{s,j}^c) - \alpha_c\||}{|\frac{\mathrm{d}}{\mathrm{d}\theta} \|T(x_{s,j}^c) - \alpha_c\|^p|} \to +\infty, p > 1, \tag{7}$$

which means the gradient of $\|T(x_{s,j}^c) - \alpha_c\|$ could be far larger than the gradient of $\|T(x_{s,j}^c) - \alpha_c\|^p$, $p > 1$.

However, when there exists an instance $x_{s,j}^c$ such that $T(x_{s,j}^c) - \alpha_c = 0$, the derivative of formula (6) with $p = 1$ does not exist. To solve the problem, we consider the case that the dataset is distributed in the unit sphere S^n. If we assume the transformation

$$T : S^n \to S^n, \tag{8}$$

then we can ensure that the derivative of formula (6) exists, because $\alpha_c \bar{\in} S^n$, if there exist two different data in $\mathcal{X}_s^c$. One method of data preprocessing, normalization, directly maps the dataset into the unit sphere. A large number of datasets are therefore suitable for our method.

Since $\alpha_i \neq \alpha_j$, for $i \neq j$, we should note that when formula (6) is minimized, different classes are separated, which implies that the inter-class margins are increased.

We use the groundtruth labels of the target domain to compute the true values of loss (6); however, the labeled data from the target domain are unavailable. Inspired by the JDA method [17], we use pseudo labels instead of the groundtruth labels. Pseudo labels can be generated by applying a classifier h trained on the source data to the target data. To make the pseudo labels more accurate, we use the iterative pseudo label refinement strategy, proposed by JDA [17].

2) Distribution Matching Loss: When source data X_s, target data X_t and the pseudo labels of X_t are given, $d_{MMD}(T(W\mathcal{P}_s), T(\mathcal{P}_t))^2$ can be written as:

$$tr(WKWL), \tag{9}$$

where

$$K = \begin{bmatrix} K_{s,s} & K_{s,t} \\ K_{t,s} & K_{t,t} \end{bmatrix}, \tag{10}$$

is a $(n_s + n_t) \times (n_s + n_t)$ kernel matrix, $K_{s,s}, K_{t,t}$ and $K_{s,t}$ respectively are the kernel matrices defined by kernel K on the data $T(X_s)$ and $T(X_t)$; W is a $(n_s + n_t) \times (n_s + n_t)$ diagonal matrix, defined as follows:

$$W = \begin{bmatrix} A_{n_s \times n_s} & \mathbf{0} \\ \mathbf{0} & I_{n_t \times n_t} \end{bmatrix}, \tag{11}$$

where

$$A_{n_s \times n_s} = \begin{bmatrix} W_1 I_{n_s^1 \times n_s^1} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & W_C I_{n_s^C \times n_s^C} \end{bmatrix}, \tag{12}$$

with $W_i = \frac{n_s \widehat{n}_t^i}{n_t n_s^i}$, for $i = 1, \cdots, C$; and $L = [L_{ij}]$ with $L_{ij} = \frac{1}{n_s^2}$ if $x_i, x_j \in X_s$; $L_{ij} = \frac{1}{n_t^2}$ if $x_i, x_j \in X_t$; otherwise, $-\frac{1}{n_s n_t}$.

IV. Dimensionality Reduction

To avoid the curse of dimensionality and speed up training, we propose an approach for dimensionality reduction, Spherical Dimension Reduction (SDR), which satisfies two conditions:

(1) the dataset is still in the unit sphere after dimension reduction;

(2) the geometric information (e.g., the angle and distance) after dimension reduction can be preserved.

The inspiration of this approach comes from stereographic projection, which maps sphere $S^n \subseteq \mathbb{R}^{n+1}$ into $\mathbb{R}^n \bigcup \infty$ while preserving the angle of intersect curves in S^n after mapping. The stereographic projection maps a $n+1$ dimensional vector x into a n dimensional linear subspace $V \subseteq \mathbb{R}^{n+1}$, which implies that the feature dimension of x is reduced to n. To construct a stereographic projection related to a vector p ($\|p\| = 1$), stereographic projection can be written as:

$$F_p(x) = \frac{x - \langle x, p \rangle p}{1 - \langle x, p \rangle}, \tag{13}$$

where x is a point in $S^n \backslash \{p\}$ and $\langle \cdot, \cdot \rangle$ is the l_2 inner product.

To make the dataset satisfy condition (1) above, we normalize the image $F_p(X)$. The final form of the dimensionality reduction can then be written as:

$$T_p(x) = P_p\Big(\frac{x - \langle x, p \rangle p}{\|x - \langle x, p \rangle p\|}\Big), \tag{14}$$

where x is a point in $S^n \backslash \{p\}$, and P_p is the orthogonal projection from R^{n+1} to R^n such that $P_p(p) = 0$. Then transformation T_p maps $S^n \backslash \{p\} \subseteq \mathbb{R}^{n+1}$ into $S^{n-1} \subseteq \mathbb{R}^n$.

Moreover, we can prove

Theorem 2. *Let $[M_{ij}]$ be the l_2 distance matrix $[|x_i - x_j|]$, $[T_p M_{ij}]$ be the l_2 distance matrix $[|T_p(x_i) - T_p(x_j)|]$, then*

$$\frac{1}{m^2} \sum_{i<j} (M_{ij} - T_p M_{ij})^2 \leqslant \frac{4}{\sqrt{m}} (\sum_{i=1} |\langle x_i, p \rangle|^2)^{\frac{1}{2}}, \tag{15}$$

where m is the sample size of the dataset X.

The proof is given in the Appendix.

To preserve the distance matrix $[M_{ij}]$ after dimensionality reduction according to Theorem 2, we need to solve an optimization problem:

$$\min_{p \subseteq S^n} (\sum_{i=1} |\langle x_i, p \rangle|^2)^{\frac{1}{2}}. \tag{16}$$

The optimization problem (16) can be written as:

$$\max_{A^T A = I} tr(A^T X^T X A), \tag{17}$$

where A is a $(n+1) \times n$ matrix.

We solve the problem (17) by singular value decomposition (SVD) . Let $\sum = X^T X$, then $\sum = USV$ by SVD, where S is a diagonal matrix with eigenvalues λ_i in decreasing order on the diagonal. We can write U as $[u_1, u_2, \cdots, u_{n+1}]_{(n+1)\times(n+1)}$, then $p = u_{n+1}$, $A = [u_1, u_2, \cdots, u_n]$, and $T_p(x) = xA/\|xA\|$.

To reduce the feature dimension to k, we need to seek $n + 1 - k$ vectors $\{p_i\}_{i=1}^{n+1-k}$, which are obtained by solving optimization problems:

$$p_i = \underset{p \subseteq S^{n-i+1}}{\arg\min} (\sum_{j=1} |\langle T_{p_1, p_2 \cdots, p_{i-1}}(x_j), p \rangle|^2)^{\frac{1}{2}}, \tag{18}$$

$i = 1, \cdots, n + 1 - k$, where $T_{p_1, p_2 \cdots, p_{i-1}} = T_{p_{i-1}} \circ \cdots \circ T_{p_1}$.

Then $T_{p_1, p_2 \cdots, p_{n+1-k}}$ maps dataset $X \subseteq S^n$ into $T_{p_1, p_2 \cdots, p_{n+1-k}}(X) \subseteq S^{k-1} \subseteq \mathbb{R}^k$. The implementation details of SDR are demonstrated in Algorithm 1.

V. Final model

A. Transformation and Neural Networks

In this section, we assume that the feature dimension of the dataset has been reduced to k by SDR.

Many proposed methods look for new representations using linear mapping or kernel mapping. However, linear mapping and kernel mapping are insufficient when transformation T is required to map S^{k-1} to S^{k-1}. Motivated by the development of deep learning, we use neural network to find transformation T.

We use a neural network architecture with a hidden layer whose size is k. According to the assumption that the transformation maps S^{k-1} to S^{k-1}, the size of the input layer and output layer is k. Therefore, T can be written as:

$$T(x;\theta) = g_2((g_1(xW_1 + b_1)W_2) + b_2), \tag{19}$$

where $\theta = \{W_i, b_i\}_{i=1}^2$ is a parameter set, W_i and $b_i, i = 1, 2$, are the $k \times k$ weight matrices and $1 \times k$ bias, g_2 is the l_2 normalization function, and g_1 is the activation function. In this paper, the softplus function and the tanh function are chosen as the standard activation function g_1.

When we fix the neural network architecture and the activation functions g_1, g_2, the class $\mathcal{T}$ of transformations is also fixed. We also desire that the final transformation T is deformed from an initial transformation T_0. In this paper, we set T_0 by selecting special weight matrices and bias as follows:

$$W_{i0} = I_{k\times k}, \quad b_{i0} = \mathbf{0}_{1\times k}, \ i = 1, 2.$$

To avoid overfitting, we use a regularization technique, which can be written as:

$$\frac{1}{2(n_S + n_T)} \sum_{i=1}^{2} \|W_i - W_{i0}\|_F^2. \tag{20}$$

The geometric meaning of term (20) is that we search the transformation T from a cylinder $H_r(T_0) \equiv \{T \in \mathcal{T} | \sum_{i=1}^2 ||W_i - W_{i0}||_F^2 \leqslant r^2, b^i \in \mathbb{R}^1, i = 1, 2\}$, where r is a large constant.

B. Sphere Retracting Transformation

We formulate the SRT approach by incorporating the above three formulas (6, 9 and 20) as follows:

$$\mathcal{L}(\theta) = \lambda tr(WKWL) + \frac{\mu}{2(n_S + n_T)} \sum_{i=1}^{2} \|W_i - W_{i0}\|_F^2 + \sum_{c=1}^{C} \Big(\sum_{j=1}^{n_s^c} \frac{\|T(\widetilde{x}_{s,j}^c; \theta) - \alpha_c\|}{n_s^c} + \sum_{j=1}^{\widehat{n}_t^c} \frac{\|T(\widetilde{x}_{t,j}^c; \theta) - \alpha_c\|}{\widehat{n}_t^c} \Big), \tag{21}$$

where λ, μ are regularization parameters, K is related to the parameter set θ, and $\{\widetilde{x}_{s,j}^c\}_{j,c=1}$, $\{\widetilde{x}_{t,j}^c\}_{j,c=1}$ are the source data and target data after dimensionality reduction.

In the implementation of minimizing (21), we run SRT iteratively according to the iterative pseudo label refinement strategy. In iteration i, the terminal step t_i for Adam Gradient Descent is

$$\begin{aligned} t &= \min\{t : d(t) < 10^{-6}\}, && \text{if } i = 1; \\ t &= \min\{t : t > t_{i-1}, d(t) < 10^{-6}\}, && \text{if } i > 1; \end{aligned}$$

where $d(t) = \frac{|\text{loss}(t) - \text{loss}(t-1)|}{\text{loss}(t-1)}$; here loss$(t)$ is the loss (21) after step t in Adam Gradient Descent. The condition $t_i < t_{i+1}$ is used to ensure convergence. Algorithm 2 shows details of SRT.

Algorithm 1: Spherical dimension reduction (SDR)

Input: Source data, target data: X_s, X_t; dimension k;
$Z \leftarrow [X_s^T, X_t^T]^T$;
$Z_0 \leftarrow l_2$-normalization(Z); %Normalize each row of Z
$n \leftarrow$ rank of Z_0;
Decompose $Z_0^T Z_0 = USV$ by SVD;
$Z_0 \leftarrow Z_0 U[:, 1:n]$;
$i \leftarrow 0$;
while $i < n - k$ **do**
  Use SVD to decompose $Z_i^T Z_i = U_i S_i V_i$;
  $Z_{i+1} \leftarrow l_2$-normalization$(Z_i U_i[:, 1:(n-i-1)])$;
  $i \leftarrow i + 1$;
$\widetilde{X}_s \leftarrow Z_{n-k}[1:n_s, :]$,
$\widetilde{X}_t \leftarrow Z_{n-k}[n_s + 1 : n_s + n_t, :]$.
Output: Source data, target data: $\widetilde{X}_s, \widetilde{X}_t$.

Algorithm 2: Sphere Retracting Transformation (SRT)

Input: Source domain, target data: $[X_s, Y_s], X_t$;
  dimension k, the number of iterations T;
  parameters λ, μ, kernel K; learning rate α;
$\widetilde{X}_s, \widetilde{X}_t \leftarrow$ SDR(X_s, X_t, k);
Use $[\widetilde{X}_s, Y_s]$ to train a classifier h;
$\widehat{Y}_{t,0} \leftarrow h(\widetilde{X}_t)$, $t \leftarrow 0$, $s \leftarrow 0$, $i \leftarrow 0$;
while $i < T$ **do**
  Initialize $W_j = I_{k\times k}$, $b_j = \mathbf{0}_{1\times k}$, $j = 1, 2$;
  Use the pseudo label $\widehat{Y}_{t,i}$ to obtain sub-domain $\widehat{\mathcal{X}}_t^c = \{\widetilde{x}_{t,j}^c\}_{j=1} \subseteq \widetilde{X}_t$, $c = 1, 2, \cdots, C$;
  Use source data $\widetilde{X}_s$ in loss (21), and update target data $\{\widetilde{x}_{t,j}^c\}_{j=1}$, $\widehat{n}_t^c$ in loss (21);
  $t \leftarrow 0$;
  while $t < +\infty$ **do**
    $t \leftarrow t + 1$;
    Update W_j and b_j, $j = 1, 2$, via Adam Gradient Descent with learning rate α;
    Update loss (21) and compute $d(t)$;
    If $d(t) < 10^{-6}$ and $t > s$:
      $\widehat{X}_s \leftarrow g_2(g_1(\widetilde{X}_s W_1 + b_1)W_2 + b_2)$,
      $\widehat{X}_t \leftarrow g_2(g_1(\widetilde{X}_t W_1 + b_1)W_2 + b_2)$;
      Use $[\widehat{X}_s, Y_s]$ to train a classifier h;
      $\widehat{Y}_{t,(i+1)} \leftarrow h(\widehat{X}_t)$;
      $s \leftarrow t$;
      Break;
  $i \leftarrow i + 1$;
$\widehat{Y}_t \leftarrow \widehat{Y}_{t,i}$.
Output: Predicted target label $\widehat{Y}_t$ and classifier h;

VI. Experiments

In this section, we first utilize the real-world datasets to verify the performance of SRT. We then conduct experiments on a synthetic dataset to understand the behavior of the learned features compared to other algorithms. Tables III-VI

show the results of our method for a range of cross-domain object recognition tasks and digital recognition tasks. Figure 2 visualizes the performance of SRT on synthetic data.

A. Real World Datasets

TABLE II
INTRODUCTION OF THE FIVE DATASETS.

Dataset	Type	#Sample	#Feature	#Class	Domain
COIL20	Object	1,440	1,024	20	CO1,CO2
USPS	Digit	1,800	256	10	U
MNIST	Digit	2,000	256	10	M
Office	Object	1,410	800(4,096)	10	A,W,D
Caltech	Object	1,123	800(4,096)	10	C

We utilize five public datasets: COIL20, USPS+MNIST, Office+Caltech10, which are benchmark datasets for the purpose of evaluation with a domain adaptation approach. Table II shows the details of the five datasets.

COIL20 (CO) contains 1,440 gray-scale images of 20 objects. The dataset is separated into two different subsets: COIL1 and COIL2 [17], which form two domain adaptation tasks: CO1→CO2, CO2→CO1. In the rest of this paper, we use A→B to denote the knowledge transfer from the source domain A to the target domain B. Office+Caltech consists of 2,533 images in 10 categories, forming four domains: AMAZON (A), WEBCAM (W), DSLR (D), and CALTECH (C). Twelve domain adaptation tasks can be constructed : A→C, A→D,$\cdots$, W →A. We use two types of features extracted from these datasets: SURF-BoW [31] and DeCAF6 [32]. USPS+MNIST consists of images sampled from handwritten digital datasets. MNIST has a training set of 60,000 examples and 10,000 test images of size 28×28. USPS contains 7,291 training images and 2,007 test images of size 16×16 (LeCun et al. 1998). They share 10 classes of digits. The datasets U and M is constructed by randomly sampling 1,800 images from USPS (U) and 2,000 images from MNIST (M). Then we have two tasks: U→M, M→U.

1) Baseline Methods: We compare our approach SRT with several baseline approaches for domain adaptation:

- 1NN and PCA + 1NN.
- TCA [22], which uses MMD [23] to match distributions.
- GFK [8], which connects two domains with geodesic curve on a Grassmann manifold.
- CORAL [10], which performs second-order subspace alignment.
- TJM [18], which jointly matches the feature representations and reweights the source data.
- JDA [17], which jointly mathches the marignal distributions and conditional distributions.
- SCA [24], which uses scatters to match domains.
- DMM [21], which learns the feature representations and reweights the source samples.

We use the parameters recommended by the original papers for all the baseline approaches. In our approach SRT, 1NN is chosen as the base classifier.

TABLE III
ACCURACY (%) ON COIL20 DATASETS.

Dataset	1NN	PCA	GFK	CORAL	TCA	TJM	JDA	SCA	SRT
CO1→CO2	83.6	84.7	87.6	85.6	88.5	90.0	89.3	89.1	**96.8**
CO2→CO1	82.8	84.0	87.9	86.9	86.3	91.8	88.5	90.5	**99.3**
Average	83.2	84.4	87.8	86.3	87.4	90.9	88.9	89.8	**98.1**

TABLE IV
ACCURACY (%) ON USPS+MNIST DATASETS.

Dataset	1NN	PCA	GFK	CORAL	TCA	TJM	JDA	SCA	SRT
U→M	44.7	45.0	46.5	30.5	51.2	52.3	**59.7**	48.0	58.7
M→U	65.9	66.2	61.2	49.2	56.3	63.3	67.3	65.1	**81.3**
Average	55.3	55.6	53.9	39.9	53.8	57.8	63.5	56.6	**70.0**

2) Implementation Details: Before reporting the evaluation results, it is necessary to explain how SRT hyper-parameters are tuned. The SRT approach has five hyper-parameters: the kernel K, the dimension k after dimension reduction, the number of iterations T, and the regularization parameters μ and λ. Apart from the hyper-parameters above, there are training parameters for AdamOptimizer.

In this paper, we set $k = 40, T = 10$ as the standard parameters. For the kernel function, we choose the Gaussian kernel

$$K_G(x, y) = \exp(-\frac{\|x - y\|^2}{2\sigma^2}), \tag{22}$$

where the kernel bandwidth σ is $median(\|a - b\|), \forall a \in \mathcal{X}_s, b \in \mathcal{X}_t$, as suggested by Gretton et al. [23]. We use AdamOptimizer with full batch and set the parameter $\epsilon = 10^{-8}$ and the exponential decay rates for the moment estimates $\beta_1 = 0.9$, $\beta_2 = 0.999$. Hence, the regularization parameters λ, μ and learning rate α are free parameters. It is impossible to tune the optimal parameters using cross validation because the labeled and unlabeled data are from different distributions. Therefore, we use grid-search to tune the free parameters. We search μ and λ from $\{1, 2, 5, 10\}$ and learning rate α from $\{0.001, 0.002, 0.003\}$. We select the activation function g_1 from the tanh and softplus functions. Parameter sensitivity analysis is provided for SRT, which will verify that SRT can achieve stable performance for a wide range of hyper-parameter settings. Table VII shows the details of the parameters we set in experiments.

The classification Accuracy [17] on the test data is

$$Accuracy = \frac{|x \in \mathcal{X}_t : f_t(x) = h(x)|}{|x : x \in \mathcal{X}_t|}, \tag{23}$$

where h is the predicted labeling function.

3) Experimental Results: The classification accuracy of 28 cross-domain tasks is demonstrated in Tables III-VI. The following facts can be observed from these tables. (1) The standard 1NN classifier performs very poorly on many of the target domains, indicating that reducing the distribution discrepancy is the key to the domain adaptation problem. (2) SRT outperforms other baseline approaches in most tasks and achieves a higher average accuracy. (3) The performance of the feature matching approaches (TCA, JDA and SCA) is generally worse than that of SRT. A major limitation of the feature matching approaches is that feature matching alone

TABLE V
ACCURACY (%) ON OFFICE+CALTECH10 DATASETS USING SURF FEATURES.

Dataset	1NN	PCA	GFK	CORAL	TCA	TJM	JDA	SCA	SRT
C→A	23.7	39.5	46.0	52.1	45.6	46.8	43.1	45.6	**50.2**
C→W	25.8	34.6	37.0	46.4	39.3	39.0	39.3	40.0	**49.8**
C→D	25.5	44.6	40.8	45.9	45.9	44.6	**49.0**	47.1	**49.0**
A→C	26.0	41.7	40.7	**45.1**	42.0	39.5	40.0	39.7	44.6
A→W	29.8	35.9	40.0	**44.4**	40.0	42.0	38.0	34.9	44.0
A→D	25.5	33.8	40.1	39.5	35.7	**45.2**	42.0	39.5	40.1
D→A	28.5	33.2	28.7	**37.7**	32.8	32.8	33.4	31.6	36.3
D→C	26.3	29.7	29.3	**33.8**	33.0	31.4	31.2	30.7	31.3
D→W	63.4	86.1	80.3	84.7	87.5	85.4	**89.2**	84.4	**89.2**
W→C	19.9	28.2	24.8	**33.7**	31.5	30.2	33.0	31.1	**33.7**
W→A	23.0	29.1	27.6	36.0	30.5	30.0	29.8	30.0	**40.1**
W→D	59.2	89.2	85.4	86.6	91.1	89.2	**92.4**	87.3	**92.4**
Average	31.4	43.6	43.1	48.8	46.2	46.3	46.8	45.2	**50.1**

TABLE VI
ACCURACY (%) ON OFFICE+CALTECH10 DATASETS USING DECAF6 FEATURES.

Dataset	1NN	PCA	GFK	CORAL	TCA	TJM	JDA	SCA	DMM	SRT
C→A	87.3	88.1	87.3	92.0	89.8	88.8	89.7	89.5	**92.4**	91.4
C→W	72.5	83.4	75.9	80.0	78.3	81.4	83.7	85.4	**87.5**	87.1
C→D	79.6	84.1	83.4	84.7	85.4	84.7	86.6	87.9	**90.4**	88.5
A→C	71.7	79.3	80.3	83.2	82.6	84.3	82.2	78.8	**84.8**	84.5
A→W	68.1	70.9	77.0	74.6	74.2	71.9	78.6	75.9	84.7	**88.8**
A→D	74.5	80.9	80.9	84.1	81.5	76.4	80.2	85.4	**92.4**	84.7
D→A	50.0	78.7	75.8	85.5	89.1	90.3	91.7	90.0	90.7	**93.1**
D→C	42.1	72.8	69.1	76.8	82.3	83.8	80.1	78.1	83.3	**85.1**
D→W	91.5	98.3	98.6	99.3	**99.7**	99.3	98.9	98.6	99.3	99.0
W→C	55.3	70.3	67.8	75.5	80.4	83.0	80.5	74.8	81.7	**83.6**
W→A	62.6	73.5	74.3	81.2	73.5	87.6	88.1	86.1	86.5	**88.9**
W→D	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	98.7	**100.0**
Average	71.1	81.7	80.9	84.7	85.6	86.0	86.7	85.9	89.4	**89.6**

TABLE VII
PARAMETERS IN EXPERIMENTS.

Dataset	λ	μ	α	function
COIL20	10	10	0.003	tanh
MNIST + USPS	5	1	0.002	tanh
Office+Caltech SURF	2	1	0.001	tanh
Office+Caltech DeCAF6	10	2	0.001	softplus

is not adequate for domain adaptation when the difference between two domains is large. SRT, TJM and DMM address the limitation by reweighing the source domain. SRT ensures that the distribution discrepancy is reduced because of the support of Theorem 1. (4) While TJM and DMM aim to reduce the domain discrepancy by matching the feature and reweighting instance, SRT performs feature matching using an indirect approach which encourages different labeling classes to be separated and causes data in the same class to become closer. (5) These results are obtained from a wide range of datasets, which means that SRT achieves good generalized performance in different domain adaptation scenarios.

B. Synthetic data

The synthetic source and target data are both generated from a mixture of three Gaussian distributions. Each Gaussian distribution represents one class. The global means, as well as the numbers of instances in different classes, are shifted between domains. The original data are 3-dimensional. We set the dimensionality of the spaces to 2 for all the approaches. Figure 2 shows that PCA performs better than the domain adaptation approaches (TCA, TJM, JDA). Although the discrepancy between domains is reduced for the TCA, TJM, and JDA, the instances in different clusters are disordered because

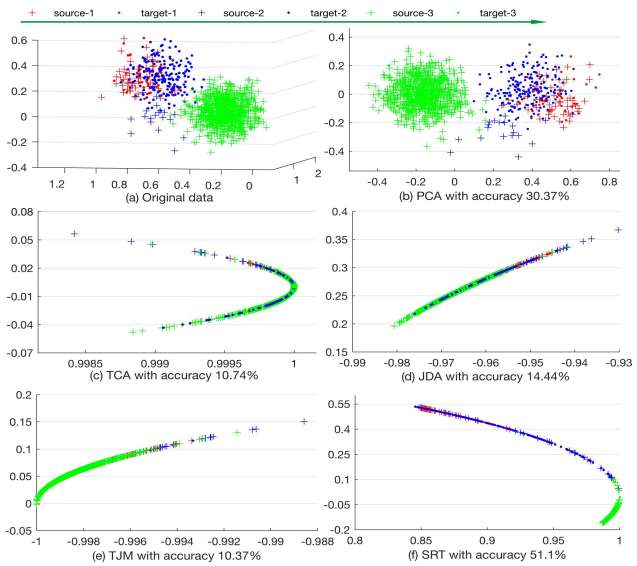

Fig. 2. Comparison of baseline domain adaptation approaches and the proposed SRT approach on synthetic data.

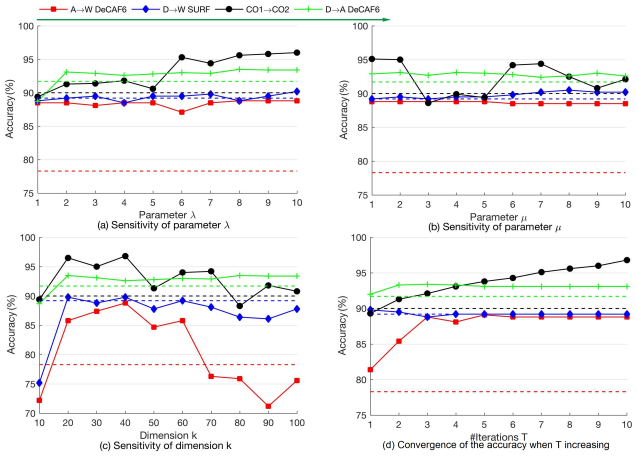

Fig. 3. Parameter sensitivity study and convergence analysis of the proposed SRT approach.

there may not be a latent feature space to simultaneously reduce domain difference and preserve the original information. However, after the application of SRT, different clusters are separated and most instances in class 3 converge into one cluster. The accuracy of SRT is 51.11%, and the accuracy is booted to more 20.74% than that of the best baseline PCA. The result confirms the effectiveness of SRT even though the difference between the source and target domains is large.

C. Parameter Sensitivity and Convergence analysis

We analyze the parameter sensitivity of SRT on different types of datasets to demonstrate that a wide range of parameter values can be chosen to obtain satisfactory performance. We evaluate three parameters: the space dimension k, and parameters λ, μ. We conduct experiments on the W→D (SURF), CO1→CO2, A→W (DeCAF6), and D→A (DeCAF6) tasks.

The results are shown in Figure 3. The solid line is the accuracy of SRT with different parameters and the dashed line denotes the results of the best baseline method (without DMM) on each dataset.

λ and μ are the regularization parameters. If λ is smaller, SRT encourages tighter clusters. If μ is smaller, the hypothesis space $\mathcal{H}$ will be larger. Figure 3 (a) shows that λ can be selected from $[2, 10]$, and Figure 3 (b) shows that μ can be selected from $[1, 10]$. For space dimension k, Figure 3 (c) illustrates that if we choose k from $[20, 60]$, we obtain better results than the best baseline method.

We analyze the convergence of the number of iterations T. The convergence of SRT on W→D (SURF), A→W (DeCAF6) and D→A (DeCAF6) in Figure 3 (d) shows the accuracy converges in 10 iterations. Although the accuracy of task CO1→CO2 does not converge in 10 iterations, the accuracy increases with the number of iterations. This may be because the learning rate α we set is so small that the accuracy does not reach its optimum value in 10 iterations.

VII. Conclusion

In this paper, a theorem is proven to show the relation between the intra-class variations and discrepancy between distributions of the source domain and the target domain. Based on this theorem, we propose a new unsupervised domain adaptation approach, Sphere Retracting Transformation (SRT). SRT matches domains by (1) reducing the intra-class variations for the source domain, and (2) weighting the source data using a class-specific weighting strategy. Experiments show that SRT outperforms several competitive approaches [33].

References

[1] J. Q. Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. MIT Press, 2009.

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

[3] B. Yang, A. J. Ma, and P. C. Yuen, "Domain-shared group-sparse dictionary learning for unsupervised domain adaptation," in *AAAI*, 2018, pp. 7453–7460.

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

[5] P. Koniusz, Y. Tas, and F. Porikli, "Domain adaptation by mixture of alignments of second- or higher-order scatter tensors," in *CVPR*, 2017, pp. 7139–7148.

[6] S. Herath, M. T. Harandi, and F. Porikli, "Learning an invariant Hilbert space for domain adaptation," in *CVPR*, 2017, pp. 3956–3965.

[7] X. Ding, B. Cai, T. Liu, and Q. Shi, "Domain adaptation via tree kernel based maximum mean discrepancy for user consumption intention identification," in *IJCAL*, 2018, pp. 4026–4032.

[8] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *CVPR*, 2012, pp. 2066–2073.

[9] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *ICCV*, 2013, pp. 2960–2967.

[10] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation." in *AAAI*, 2016, pp. 2058–2065.

[11] J. Zhang, W. Li, and P. Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *CVPR*, 2017, pp. 5150–5158.

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

[13] C. Cortes, Y. Mansour, and M. Mohri, "Learning bounds for importance weighting," in *NIPS*, 2010, pp. 442–450.

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

[15] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *CVPR*, 2017, pp. 945–954.

[16] M. Ishii and A. Sato, "Joint optimization of feature transform and instance weighting for domain adaptation," in *IJCNN*, 2017, pp. 3793–3799.

[17] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *ICCV*, 2013, pp. 2200–2207.

[18] ——, "Transfer joint matching for unsupervised domain adaptation," in *CVPR*, 2014, pp. 1410–1417.

[19] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[20] 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, 2015.

[21] Y. Cao, M. Long, and J. Wang, "Unsupervised domain adaptation with distribution matching machines," in *AAAI*, 2018, pp. 2795–2802.

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

[23] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, pp. 723–773, 2012.

[24] 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 & Machine Intelligence*, vol. 39, no. 7, pp. 1414–1430, 2017.

[25] M. Ghifary, W. B. Kleijn, and M. Zhang, "Domain adaptive neural networks for object recognition," in *PRICAI*, 2014, pp. 898–904.

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

[27] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. K. Sriperumbudur, "Optimal kernel choice for large-scale two-sample tests," in *NIPS*, 2012, pp. 1205–1213.

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

[29] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, "Domain adaptation under target and conditional shift," in , *ICML*, 2013, pp. 819–827.

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

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

[32] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014, pp. 647–655.

[33] B. Gholami, P. Sahu, O. Rudovic, K. Bousmalis, and V. Pavlovic, "Unsupervised multi-target domain adaptation: An information theoretic approach," *CoRR*, vol. abs/1810.11547, 2018. [Online]. Available: http://arxiv.org/abs/1810.11547