Spherical Space Domain Adaptation with Robust Pseudo-label Loss

Xiang Gu, Jian Sun (✉) and Zongben Xu
Xi'an Jiaotong University, Xi'an, 710049, China
xianggu@stu.xjtu.edu.cn, {jiansun, zbxu}@xjtu.edu.cn

Abstract

Adversarial domain adaptation (DA) has been an effective approach for learning domain-invariant features by adversarial training. In this paper, we propose a novel adversarial DA approach completely defined in spherical feature space, in which we define spherical classifier for label prediction and spherical domain discriminator for discriminating domain labels. To utilize pseudo-label robustly, we develop a robust pseudo-label loss in the spherical feature space, which weights the importance of estimated labels of target data by posterior probability of correct labeling, modeled by Gaussian-uniform mixture model in spherical feature space. Extensive experiments show that our method achieves state-of-the-art results, and also confirm effectiveness of spherical classifier, spherical discriminator and spherical robust pseudo-label loss.

1. Introduction

Deep learning approach has achieved great success in visual recognition [18, 22, 50]. Unfortunately, these performance improvements rely on massive labeled training data, and data labeling is expensive and time consuming. Domain adaption [37] alleviates the dependency on large scale labeled training dataset by transferring knowledge from relevant source domain with rich labeled data. Distribution discrepancy between source and target domains is a major obstacle in adapting predictive models across domains.

Domain adaptation mainly attempts to reduce domain shift between source and target domains [10, 37]. Previous shallow domain adaptation methods either learn invariant feature representation or estimate instance importance of source domain data [16, 20, 36] for learning predictive model for target domain. Recently, deep learning approach has been a dominant approach in domain adaptation [13, 30, 53]. These methods take advantage of deep network for learning domain invariant features by aligning distributions [21, 31, 33, 38, 56, 58, 62]. Adversarial domain adaptation [13, 14, 31, 33, 57, 58, 62] matches feature distributions of source and target domains by domain discriminator for distinguishing source and target domains, and learns feature extractor to fool the discriminator by adversarial training. Pseudo-labels of target domain, *i.e*., the estimated labels of target domain data, have shown to be useful for domain adaptation [4, 5, 47, 58, 61]. Since pseudo-labels unavoidably contain noises, how to select correctly labeled data is crucial when using pseudo-labels to guide domain adaptation task.

Though they have shown promising performance in real applications, current domain adaption methods still face great challenges, including the design of effective invariant feature space, and utilization of the pseudo-labels in a more robust way, *etc*. In this work, we tackle these two challenges in a unified model by proposing a spherical space domain adaptation method. Our method performs DA completely in spherical space by defining spherical classifier and discriminator and defining a robust pseudo-label loss in spherical feature space based on Gaussian-uniform mixture model. The proposed techniques can be embedded into other DA methods as orthogonal tools. This proposed domain adaption approach is dubbed as *robust spherical domain adaption* (RSDA). Our novelties are summarized as follows.

Firstly, since spherical (L2 normalized) features have shown improved performance in recognition and domain adaptation [29, 42, 46, 55, 59], we further extend this idea and design a novel spherical space DA approach with all DA operations defined in spherical feature space, fully taking advantages of the intrinsic structures of spherical space. To achieve that goal, we propose spherical discriminator and spherical classifier for adversarial DA performed in the spherical feature space. Both spherical discriminator and classifier are constructed based on spherical perceptron layers and spherical logistic regression layer defined in Sect. 5.

Secondly, we propose a novel robust pseudo-label loss in spherical feature space for utilizing target pseudo-labels more robustly. We measure the correctness of pseudo-label of target data based on feature distance to corresponding class center in spherical feature space. We treat the wrongly labeled data as outliers, then model the conditional probability of outlier / inlier based on Gaussian-uniform mixture model, which is defined in Sect. 4. Experiments will justify

the effectiveness of the proposed robust loss.

Based on these above two techniques, we design a novel training loss for domain adaption in spherical feature space, which is alternately optimized in a principled way. Our method is built upon two baselines, DANN [14] and MSTN [58]. Comprehensive experiments on standard datasets of Office-31, ImageCLEF-DA, Office-Home and VisDA-2017 show that the proposed techniques are effective. In experiments, RSDA achieves state-of-the-art results on all datasets.

2. Related Works

Adversarial domain adaptation. Adversarial domain adaptation [13, 51, 52] integrates adversarial learning and domain adaptation in a two-player game, in which domain discriminator and feature extractor are adversarially trained to learn domain invariant features. Methods in [19, 28] reduce domain shift in raw pixel space by translating source domain image to the style of target domain. Methods in [40, 63, 31] propose to align conditional distributions in feature space to ensure correct matching. Wang *et al*. [57] and Kurmi *et al*. [23] apply attention to adversarial DA. Saito *et al*. [48] and Zhang *et al*. [62] utilize task-specific classifiers as a discriminator to learn invariant features. Recently, other types of adversarial learning, *e.g*., minimax entropy [46], drop to adapt [26], adversarial regularization and adversarial sample based methods [6, 11, 27] have been proposed. We next summarize closely related works to ours.

Pseudo-label based methods. Recent domain adaptation methods often use pseudo-labels of target domain to learn semantic features. Methods in [5, 57, 58] utilize pseudo-labels to estimate target class centers which are used to match source class centers. CPUA [35] employs classification scores as features for adversarial learning. CAN [21] utilizes target pseudo-labels to estimate contrastive domain discrepancy. iCAN [61] designs a robust pseudo-label loss by selecting data based on predicted classification score. In this work, we propose to measure correctness of pseudo-labels of target data by Gaussian-uniform mixture model. It is modeled based on the feature distance to class center in the spherical feature space by a robust model. As shown in experiments, our method can robustly utilize pseudo-labels and achieves performance improvement.

Normalization based methods. Our approach also relates to Kang *et al*. [21], Saito *et al*. [46], Roy *et al*. [43] and Xu *et al*. [59]. Kang *et al*. [21] obtain pseudo-labels via k-means clustering using cosine dissimilarity. Roy *et al*. [43] whitens source and target features to a common spherical distribution, which is a generalization of Batch Normalization. Xu *et al*. [59] progressively adapts the feature norms of the two domains to a large range of values. Saito *et al*. [46] exploits cosine similarity-based classifier for semi-supervised domain adaptation. Different to these methods, our approach projects features onto sphere, meanwhile constructs spherical network on sphere, and also designs a robust pseudo-label loss on sphere, deducing a novel domain adaptation method completely defined in spherical space.

3. Method

3.1. Overview

Given labeled dataset $\{x_i^s, y_i^s\}_{i=1}^{N_s}$ in source domain and unlabeled dataset $\left\{x_j^t\right\}_{j=1}^{N_t}$ in target domain, our goal is to learn a classifier from labeled source data to transfer to the unlabeled target data. This can be achieved by learning a domain invariant feature extractor F by introducing a discriminator D on top of F to distinguish source and target domains, and F is learned to fool discriminator D [13].

Figure 1 illustrates our robust spherical domain adaptation method. It has the following distinct characteristics. We learn domain invariant features by adversarial training completely in spherical feature space. Using a backbone CNN (*e.g*., ResNet [18]) as feature extractor F, the features are normalized to map onto a sphere. Our classifier C and discriminator D are all accordingly defined in the spherical feature space, which consist of spherical perceptron layers and spherical logistic regression layer. We also propose a robust pseudo-label loss in spherical feature space to fully utilize pseudo-labels of target domain data in a robust way based on Gaussian-uniform mixture model.

Spherical feature embedding retains the power of feature learning because it only reduces feature dimension by one but makes domain adaptation easier since differences in norms are eliminated. Thus, our method performing DA in spherical space may better solve DA problem.

3.2. Spherical Adversarial Training Loss

Spherical adversarial training loss is defined as

$$\mathcal{L} = \mathcal{L}_{bas}(F, C, D) + \mathcal{L}_{rob}(F, C, \phi) + \gamma \mathcal{L}_{ent}(F), \quad (1)$$

which is combination of basic loss, robust pseudo-label loss and conditional entropy loss, and all of these losses are defined in the spherical feature space. By minimizing the total loss, we enforce to learn classifier in source domain and align features across domains by $\mathcal{L}_{bas}$, utilize pseudo-labels of target domain in a robust way by $\mathcal{L}_{rob}$, and reduce prediction uncertainty by $\mathcal{L}_{ent}$. We next introduce these losses.

Basic loss. This is the basic adversarial domain adaption loss. Taking DANN [14] and MSTN [58] as baseline methods, this basic loss is composed of cross entropy loss $\mathcal{L}_{src}$ in source domain with ground-truth labels, an adversarial training loss $\mathcal{L}_{adv}$, and a semantic matching loss

$$\mathcal{L}_{bas}(F, C, D) = \mathcal{L}_{src}(F, C) + \lambda \mathcal{L}_{adv}(F, D) + \lambda' \mathcal{L}_{sm}(F), \quad (2)$$

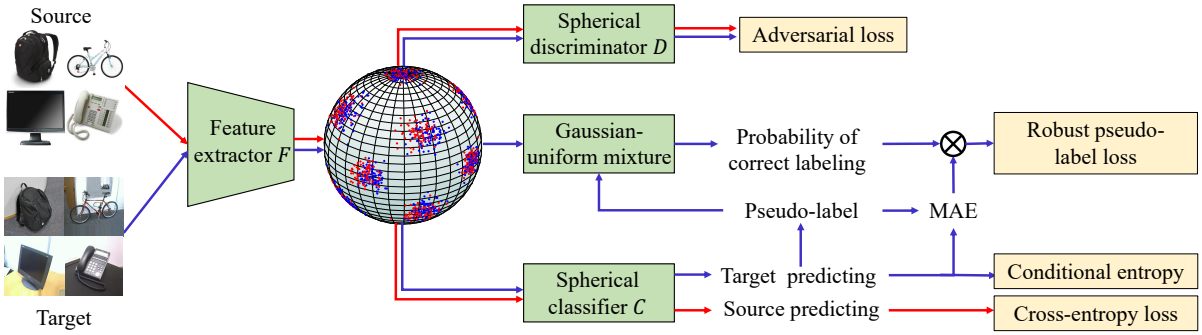


Figure 1. Architecture of our Robust Spherical Domain Adaptation (RSDA) method. Red and blue arrows indicate computational flows for source domain and target domain respectively. The feature extractor F is a deep convolutional network, the extracted features are embedded onto a sphere, the spherical classifier and discriminator (in Fig. 3) are constructed to predict class labels and domain labels respectively. The target pseudo-labels along with features are fed into a Gaussian-uniform mixture model to estimate the posterior probability of correct labeling, which is used to weight the pseudo-label loss for robustness.

where F, C, D are respectively the spherical feature extractor, spherical classifier and spherical discriminator. The spherical feature is just the l_2-normalized feature extracted by backbone feature extraction network. The spherical classifier C and discriminator D are networks defined in spherical feature space as discussed in Sect. 5. Semantic matching loss is defined as $\mathcal{L}_{sm} = \sum_{k=1}^{K} \text{dist}(\mathcal{C}_k^s, \mathcal{C}_k^t)$ based on MSTN [58], where $\mathcal{C}_k^s, \mathcal{C}_k^t$ are centroids for k-th class in spherical space, as in Appendix A, $\text{dist}(u, v) = 1 - \frac{u^T v}{||u|| ||v||}$ is cosine distance. When $\lambda' = 0$ and 1, $\mathcal{L}_{bas}$ is respectively the spherical version of loss in DANN and MSTN.

Conditional entropy loss. We also consider a conditional entropy loss [17, 32, 49, 60, 63]

$$\mathcal{L}_{ent}(F) = \frac{1}{N_t} \sum_{j=1}^{N_t} H\left(C(F(x_j^t))\right), \tag{3}$$

where $H(\cdot)$ is the entropy of a distribution. Minimizing entropy encourages the learned features being away from the classification boundary, and reduces the uncertainty of the predicted classification probability. Conditional entropy minimization is also seen as implicit pseudo-label constraint as discussed in [25]. Following [60], we only use conditional entropy to update F.

In following sections, we will introduce our robust pseudo-label loss $\mathcal{L}_{rob}$ in spherical space and spherical neural network for defining classifier C and discriminator D.

4. Robust Pseudo-label Loss in Spherical Space

Since data in target domain are unlabeled, their pseudo-labels estimated by classifier C could be helpful to learn discriminative features for both source and target domains. However, these pseudo-labels are not accurate, we therefore propose a novel robust loss in spherical feature space to utilize these pseudo-labels. The pseudo-label $\tilde{y}_j^t$ of x_j^t in target domain is $\tilde{y}_j^t = \arg\max_k [C(F(x_i^s))]_k$, where $[\cdot]_k$ denotes the k-th element. To model the fidelity of pseudo-label, we introduce a random variable $z_j \in \{0, 1\}$ for each target sample with pseudo-label, *i.e.*, $\left(x_j^t, \tilde{y}_j^t\right)$, indicating whether the data is correctly or wrongly labeled by values of 1 and 0 respectively. If probability of correct labeling is denoted as $P_\phi\left(z_j = 1 | x_j^t, \tilde{y}_j^t\right)$, where ϕ denotes parameters, then our robust loss is defined as

$$\mathcal{L}_{rob}(F, C, \phi) = \frac{1}{N_0} \sum_{j=1}^{N_t} w_\phi(x_j^t) \mathcal{J}\left(C(F(x_j^t)), \tilde{y}_j^t\right), \tag{4}$$

where $N_0 = \sum_{j=1}^{N_t} w_\phi(x_j^t)$, $\mathcal{J}(\cdot, \cdot)$ is taken as mean absolute error (MAE) [15]. $w_\phi(x_j^t)$ is defined based on the posterior probability of correct labeling

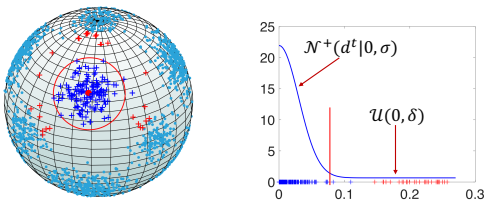
$$w_\phi(x_j^t) = \begin{cases} \gamma_j, & \text{if } \gamma_j \geq 0.5, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where $\gamma_j = P_\phi\left(z_j = 1 | x_j^t, \tilde{y}_j^t\right)$. In this way, we discard target data with probability of correct labeling less than 0.5. The probability $P_\phi\left(z_j = 1 | x_j^t, \tilde{y}_j^t\right)$ is modeled by feature distance of data to center of the class that it belongs to, using Gaussian-uniform mixture model in spherical space based on pseudo-labels, which will be given in Sect. 4.1.

4.1. Posterior Probability of Correct Labeling

We now compute posterior probability of correct labeling, *i.e.*, $P_\phi\left(z_j = 1 | x_j^t, \tilde{y}_j^t\right)$ for each target data indexed by j. As shown in Fig. 2(a), for data in target domain with pseudo-labels, we assume that data with larger feature distance to class center, *e.g.*, the red points on sphere, have larger possibility of being wrongly labeled.

Given spherical feature f_j^t for j-th target data, its distance to corresponding spherical class center $\mathcal{C}_{\tilde{y}_j^t}$ for class



(a) (b)

Figure 2. (a) The wrongly labeled target data (red) are away from the predicted class center, whereas the correctly labeled data (blue) cluster around the class center. (b) The distribution of distances of features to center modeled by Gaussian-uniform mixture model.

$\tilde{y}_j^t$ is computed by $d_j^t = \text{dist}(f_j^t, \mathcal{C}_{\tilde{y}_j^t})$, where $\text{dist}(\cdot,\cdot)$ is cosine distance. We model distribution of feature distance d_j^t for each class by Gaussian-uniform mixture model, a statistical distribution considering outliers [9, 24],

$$p(d_j^t|\tilde{y}_j^t) = \pi_{\tilde{y}_j^t}\mathcal{N}^+(d_j^t|0,\sigma_{\tilde{y}_j^t}) + (1-\pi_{\tilde{y}_j^t})\mathcal{U}(0,\delta_{\tilde{y}_j^t}), \quad (6)$$

where $\mathcal{N}^+(u|0,\sigma)$ is with density proportional to Gaussian distribution when $u \geq 0$, otherwise the density is zero. $\mathcal{U}(0,\delta_{\tilde{y}_j^t})$ is uniform distribution defined on $[0,\delta_{\tilde{y}_j^t}]$. The Gaussian component models the correctly labeled target data and uniform component models the wrongly labeled data, as shown in Fig. 2(b). With Eq. (6), the posterior probability of correct labeling for j-th target data is

$$P_\phi\left(z_j = 1|x_j^t, \tilde{y}_j^t\right) = \frac{\pi_{\tilde{y}_j^t}\mathcal{N}^+(d_j^t|0,\sigma_{\tilde{y}_j^t})}{\pi_{\tilde{y}_j^t}\mathcal{N}^+(d_j^t|0,\sigma_{\tilde{y}_j^t}) + (1-\pi_{\tilde{y}_j^t})\mathcal{U}(0,\delta_{\tilde{y}_j^t})}. \quad (7)$$

The parameters of Gaussian-uniform mixture models are $\phi = \{\pi_k, \sigma_k, \delta_k\}_{k=1}^K$ where K is number of classes. These parameters will be estimated in Sect. 6.

5. Spherical Neural Network

This section introduces details on how spherical classifier and discriminator are constructed based on spherical neural network (SNN). Note the term of SNN has also been used in spherical CNNs [7, 8] and geometric SNNs [3]. Different to them, our SNN is an extension of MLP from Euclidean space to spherical space. Before defining spherical neural network, we normalize feature with $f = r\frac{F(x)}{||F(x)||}$ to obtain features in spherical space $\mathbb{S}_r^{n-1} = \{f \in \mathbb{R}^n : ||f|| = r\}$. As shown in Fig. 3, our classifier (discriminator) is constructed by stacking M_C (M_D) spherical perceptron (SP) layers and a final spherical logistic regression (SLR) layer. We next introduce SP and SLR layers.

The SP layer is an extension of perceptron layer of MLP from Euclidean space to sphere. A perceptron layer of MLP consists of a linear transform and an activation function. Inspired by hyperbolic neural network [12], we will define spherical linear transform and spherical activation function.

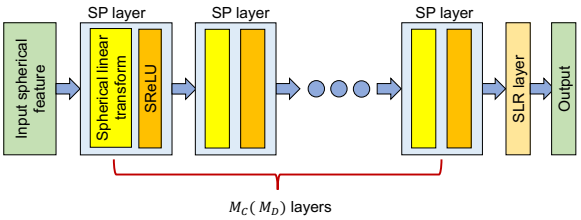


Figure 3. Structure of spherical neural network. It is constructed by stacking multiple spherical perceptron (SP) layers and a final spherical logistic regression (SLR) layer.

Spherical linear transform. The spherical linear transform consists of three components, *i.e.*, a spherical logarithmic map, a linear transform in tangent space and a spherical exponential map. When performing the spherical linear transform from one spherical space to another, we first project features in the former spherical space to its tangent space (*i.e.*, a hyperplane), then transform the projected features into tangent space of the later spherical space by the linear transform, finally project the transformed features into the later spherical space by the spherical exponential map. Mathematically, the spherical linear transform $g_s : \mathbb{S}_r^{n_1-1} \to \mathbb{S}_r^{n_2-1}$ is defined by

$$g_s(x) = \exp_{N_2}(g(\log_{N_1}(x))), \quad (8)$$

where $g : T_{N_1}\mathbb{S}_r^{n_1-1} \to T_{N_2}\mathbb{S}_r^{n_2-1}$ is a linear transform, $\exp_{N_2}$ and $\log_{N_1}$ are spherical exponential and logarithmic maps respectively, $N_i = (0,\cdots,0,r) \in \mathbb{R}^{n_i}$ is north pole of $\mathbb{S}_r^{n_i-1}$, $i = 1,2$. Due to space limit, expressions of $\exp_{N_2}$ and $\log_{N_1}$ are given in Appendix A. They can be implemented by simple mathematical operations.

Spherical activation function. It is easy to define nonlinear activation function in spherical space. We define spherical ReLU by

$$\text{SReLU}(x) = r\frac{\text{ReLU}(x)}{||\text{ReLU}(x)||}, \quad \forall x \in \mathbb{S}_r^{n-1}. \quad (9)$$

Spherical perceptron layer. With above spherical linear transform and spherical activation function, given input spherical feature $f_{in} \in \mathbb{S}_r^{n_1-1}$ of the SP layer, the output spherical feature $f_{out} \in \mathbb{S}_r^{n_2-1}$ is obtained by

$$f_{out} = \text{SReLU}(g_s(f_{in})). \quad (10)$$

Parameters of SP layer come only from linear transform g.

Spherical logistic regression layer. This layer is designed for predicting classification scores on sphere. A circle on sphere $\mathbb{S}_r^{n-1}$ corresponds to a hyperplane in $\mathbb{R}^n$. The circle can be expressed as $w^T z + b = 0$, where $z \in \mathbb{S}_r^{n-1}$, w is a unit normal vector, b is bias in $[-r, r]$. Similar to Euclidean logistic regression, we define SLR layer as

$$p(y = k|z) \propto \exp(w_k^T z + b_k), \quad k = 1,2,\cdots,K, \quad (11)$$

where $w_k \in \mathbb{R}^n, ||w_k|| = 1, b_k \in [-r, r]$. $w_k^T z + b_k = 0$ is the classification circle boundary on $\mathbb{S}_r^{n-1}$. The constraint that $b_k \in [-r, r]$ can be enforced by modeling $b_k = r \tanh(b'_k)$ where $b'_k \in \mathbb{R}$ is a parameter to be learned.

Structure of spherical classifier and discriminator. The number of layers and nodes of spherical classifier C and spherical discriminator D are the same as that of [14] . The spherical classifier C is composed of a SLR layer. The spherical discriminator D consists of two SP layers each with 1024 nodes and a final SLR layer.

Bound of spherical radius. To obtain a proper estimate of spherical radius r, we have the following bound

$$r \geq \frac{K-1}{K} \ln \frac{(K-1)P_w}{1-P_w}, \tag{12}$$

where P_w is a hyper-parameter indicating expected minimal classification probability of class center. The deduction of the bound is given in Appendix B.

6. Training Algorithm

In this section, we discuss on how to optimize networks F, C, D and estimate the parameters ϕ of Gaussian-uniform mixture models. To minimize total loss in Eq. (1), we alternately optimize networks and estimate parameters ϕ by fixing others as known. Initially, we train networks with basic loss Eq. (2) via training strategies as in [14, 58] to initialize F, C, D. Then we alternately run the following procedures.

Estimating ϕ with fixed F, C, D. Fixing F, C, D, we first update pseudo-label $\tilde{y}_j^t$ and calculate the distance d_j^t for all target data, then ϕ is estimated using EM algorithm as below. Let $\tilde{d}_j^t = (-1)^{m_j} d_j^t$, where m_j is sampled from Bernoulli distribution $B(1, 0.5)$, then ϕ can be estimated via the following EM algorithm

$$\begin{aligned}
\gamma_j^{(l+1)} &= \frac{\pi_{\tilde{y}_j^t}^{(l)} \mathcal{N}(\tilde{d}_j^t | 0, \sigma_{\tilde{y}_j^t}^{(l)})}{\pi_{\tilde{y}_j^t}^{(l)} \mathcal{N}(\tilde{d}_j^t | 0, \sigma_{\tilde{y}_j^t}^{(l)}) + (1 - \pi_{\tilde{y}_j^t}^{(l)}) \mathcal{U}(-\delta_{\tilde{y}_j^t}^{(l)}, \delta_{\tilde{y}_j^t}^{(l)})}, \\
\pi_k^{(l+1)} &= \frac{1}{\sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}}} \sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}} \gamma_j^{(l+1)}, \\
\sigma_k^{(l+1)} &= \frac{\sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}} \gamma_j^{(l+1)} (\tilde{d}_j^t)^2}{\sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}} \gamma_j^{(l+1)}}, \delta_k^{(l+1)} = \sqrt{3(q_2 - q_1^2)},
\end{aligned} \tag{13}$$

where

$$\begin{aligned}
q_1 &= \frac{1}{\sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}} \gamma_j^{(l+1)}} \sum_{j=1}^{N_t} \frac{1 - \gamma_j^{(l+1)}}{1 - \pi_k^{(l+1)}} I_{\{\tilde{y}_j^t = k\}} \tilde{d}_j^t, \\
q_2 &= \frac{1}{\sum_{j=1}^{N_t} I_{\{\tilde{y}_j^t = k\}} \gamma_j^{(l+1)}} \sum_{j=1}^{N_t} \frac{1 - \gamma_j^{(l+1)}}{1 - \pi_k^{(l+1)}} I_{\{\tilde{y}_j^t = k\}} (\tilde{d}_j^t)^2.
\end{aligned}$$

Deductions of Eq. (13) are given in Appendix B.

Optimizing F, C, D with fixed ϕ. Given current target pseudo-labels and estimated ϕ, training network F, C, D is a standard domain adaptation training problem, which can be performed via progressive adversarial training strategy as in [14] with objective function Eq. (1).

7. Theoretical Analysis

Theoretical analysis of our approach is based on the theory of domain adaptation [1, 2]

$$\varepsilon_T(h) \leq \varepsilon_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_S, P_T) + \lambda^*, \tag{14}$$

where h is in hypothesis space $\mathcal{H}$, $\varepsilon_S(h)$ and $\varepsilon_T(h)$ are expected risks of source and target domains respectively, $\lambda^* = \min_{h' \in \mathcal{H}} \varepsilon_S(h') + \varepsilon_T(h')$ is the combined error of ideal joint hypothesis, $d_{\mathcal{H}\Delta\mathcal{H}}(P_S, P_T)$ is the $\mathcal{H}\Delta\mathcal{H}$-divergence between source and target domains. For our approach, we further consider classification error for pseudo-labels in deduction of our upper bound, obtaining following lemma.

Lemma 1. *Let $h \in \mathcal{H}$ be a hypothesis, f_S and f_T be the true labeling function for source and target respectively, f'_T be the pseudo-labeling function for target domain, then*

$$\begin{aligned}
\varepsilon_T(h) \leq & \frac{1}{2}(\varepsilon_S(h) + \varepsilon_T(h, f'_T) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_S, P_T)) \\
& + \varepsilon_T(f'_T, f_T) + \frac{1}{2}\beta,
\end{aligned} \tag{15}$$

where $\varepsilon_T(h, h') = \mathbb{E}_{x \sim P_T}[h(x) \neq h'(x)]$, $\beta = \min_{h' \in \mathcal{H}} \{\varepsilon_S(h') + \varepsilon_T(h', f'_T)\}$ is a constant to h.

The proof is given in Appendix B. For our approach, the source error, *i.e.*, $\varepsilon_S(h)$ in Eq. (15), is imposed by source domain cross-entropy loss, the classification error for pseudo-labels, *i.e.*, $\varepsilon_T(h, f'_T)$, is conducted by robust pseudo-label loss, $d_{\mathcal{H}\Delta\mathcal{H}}(P_S, P_T)$ is minimized through adversarial training. Our Gaussian-uniform mixture model to select target correct pseudo-labels implicitly minimizes the disagreement between target pseudo-labels and true labels, *i.e.*, $\varepsilon_T(f'_T, f_T)$, and β is a constant w.r.t. h.

8. Experiments

We evaluate the proposed method on following domain adaptation datasets, comparing with many state-of-the-art domain adaptation methods. Code is available at https://github.com/XJTU-XGU/RSDA.

Datasets. We will evaluate on the Office-31, ImageCLEF-DA, Office-Home and VisDA-2017. **Office-31** dataset [45] contains 4,110 images of 31 categories shared by three distinct domains: *Amazon* (**A**), *Webcam* (**W**) and *Dslr* (**D**). **ImageCLEF-DA** dataset has been used by [33], containing three distinct domains: *Caltech-256* (**C**), *ImageNet ILSVRC 2012* (**I**) and *Pascal VOC 2012* (**P**), sharing 12 classes. **Office-Home** dataset [54] is well organized and more challenging than Office-31, which consists of 15,500 images in

Table 1. Accuracy (%) on Office-31 for unsupervised domain adaption (ResNet-50). * Reproduced by [4].

Method	A→W	W→A	A→D	D→A	D→W	W→D	Avg
ResNet-50 [18]	68.4±0.2	60.7±0.3	68.9±0.2	62.5±0.3	96.7±0.1	99.3±0.1	76.1
iCAN [61]	92.5	69.9	90.2	72.1	98.8	**100.0**	87.2
CDAN [31]	94.1±0.1	69.3±0.3	92.9±0.2	71.0±0.3	98.6±0.1	**100.0±.0**	87.7
SymNets [63]	90.8±0.1	72.5±0.5	93.9±0.5	74.6±0.6	98.8±0.3	**100.0±.0**	88.4
MDD [62]	94.5±0.3	72.2±0.1	93.5±0.2	74.6±0.3	98.4±0.1	**100.0±.0**	88.9
CAN [21]	94.5±0.3	77.0±0.3	95.0±0.3	**78.0±0.3**	99.1±0.2	99.8±0.2	90.6
SAFN+ENT [59]	90.1±0.8	70.2±0.3	90.7±0.5	73.0±0.2	98.6±0.2	99.8±0.0	87.1
CAT [11]	94.4±0.1	70.2±0.1	90.8±1.8	72.2±0.6	98.0±0.2	100.0±0.0	87.1
DANN [14]	82.0±0.4	67.4±0.5	79.7±0.4	68.2±0.4	96.9±0.2	99.1±0.1	82.2
DANN+S (ours)	93.2±0.8	71.0±0.5	87.5±0.2	70.3±0.8	98.0±0.2	**100.0±.0**	86.7
RSDA-DANN (ours)	95.3±0.3	76.0±0.6	95.2±0.2	75.5±0.6	**99.3±0.2**	**100.0±.0**	90.2
MSTN* [58]	91.3	65.6	90.4	72.7	98.9	**100.0**	86.5
MSTN+S (ours)	94.6±0.3	76.0±0.6	91.3±0.7	75.4±0.7	98.5±0.2	**100.0±.0**	89.3
RSDA-MSTN (ours)	**96.1±0.2**	**78.9±0.3**	**95.8±0.3**	77.4±0.8	**99.3±0.2**	**100.0±.0**	**91.1**

65 object classes, coming from four extremely different domains: *Artistic images* (**Ar**), *Clip Art* (**Cl**), *Product images* (**Pr**) and *Real-World images* (**Rw**). **VisDA-2017** [41] is a large scale dataset with two extremely distinct domains: **Synthetic** and **Real**, sharing 12 classes.

Implementation details. We implement our method based on PyTorch [39]. The feature extractor F is set to ResNet-50 [18] pre-trained on ImageNet dataset [44], excluding the last FC layer. P_w in Eq. (12) is set to 0.999, and the spherical radius r is set to the bound. When optimizing F, C and D, all network parameters are updated by stochastic gradient descent (SGD) with momentum of 0.9. The learning rates of C and D are 10 times of F. We also follow [63] to set $\gamma = \lambda$. Following [14], the learning rate η and the hyper-parameter λ are adjusted by $\eta = \frac{0.01}{(1+\alpha p)^\beta}$ and $\lambda = \frac{2}{1+\exp(-\tau p)} - 1$, where $\alpha = 10, \beta = 0.75, \tau = 10$ and p is the optimizing progress linearly changing from 0 to 1, which means λ and γ increase from 0 to 1. We perform the alternated iteration 10 times and in each time, SGD is performed 5000 steps when optimizing network parameters. When estimating ϕ, we enforce $\pi_k \leq 0.5$ to control that the rate of samples from Gaussian distribution should be not larger than 0.5 to further enhance robustness of model.

We implement our RSDA based on DANN [14] and MSTN [58] by setting $\lambda' = 0$ and $\lambda' = \lambda$ in Eq. (2) respectively. DANN can be considered as the most representative method of adversarial domain adaptation. MSTN is a recently proposed method widely taken as backbone of several methods [4, 5]. It is worth noting that RSDA can also be implemented based on other adversarial domain adaptation methods by embedding our spherical techniques, such as spherical network and robust pseudo-label loss to them.

8.1. Results

We report average classification accuracies with standard deviations for all adaptation tasks on benchmark datasets. The results on Office-31, ImageCLEF-DA, Office-Home and VisDA-2017 are reported in Tables 1, 2, 3 and 4 respectively. Results of other methods are either from their original papers if available, or quoted from [31]. In the tables, we denote performing in spherical feature space as "S", robust pseudo-label loss of Eq. (4) as "R" and conditional entropy of Eq. (3) as "E". DANN+S means performing DANN in spherical feature space, *i.e.*, the features are projected onto spherical space and the classifier and discriminator are built based on spherical networks. DANN+S+R means adding robust pseudo-label loss to DANN+S. MSTN+S, DANN+S+R+E, *etc.*, are similarly defined. RSDA-DANN (RSDA-MSTN) denoting RSDA based on DANN (MSTN) is equivalent to DANN+S+R+E (MSTN+S+R+E).

Comparison with baselines. In Table 1, RSDA-DANN and RSDA-MSTN improve accuracies of DANN and MSTN by 8.0% and 4.6% respectively on Office-31. In Table 2, on ImageCLEF-DA, RSDA-DANN and RSDA-MSTN improve their baselines DANN and MSTN by 5.1% and 2.3% in average respectively. Table 3 compares results on Office-Home, our RSDA-DANN and RSDA-MSTN improve their baselines DANN and MSTN by 12.2% and 5.2% respectively. In Table 4, RSDA-DANN improves the accuracy of DANN by 12.1% on VisDA-2017. These improvements imply effectiveness of our method of RSDA.

Comparison with state-of-the-art methods. This paragraph compares our method with state-of-the-art (SOTA) methods of CAN [21], SymNets [63] and MDD [62]. On Office-31 dataset, our method of RSDA-MSTN achieves SOTA classification accuracy (91.1%), outperforming CAN by 0.5%. On ImageCLEF-DA dataset, our RSDA-MSTN achieves SOTA classification accuracy (90.5%), outperforming SymNets by 0.6%, and RSDA-MSTN significantly outperforms SymNets on Office-31 and Office-Home by 2.7% and 3.3% respectively. On Office-Home dataset, the SOTA result (70.9%) achieved by RSDA-MSTN improves classification accuracy of MDD (68.1%) by 2.8%.

Table 2. Accuracy (%) on ImageCLEF-DA for unsupervised domain adaption (ResNet-50). * Reproduced by us with ResNet-50.

Method	I→P	P→I	I→C	C→I	C→P	P→C	Avg
ResNet-50 [18]	74.8±0.3	83.9±0.1	91.5±0.3	78.0±0.2	65.5±0.3	91.2±0.3	80.7
iCAN [61]	79.5	89.7	94.7	89.9	78.5	92.0	87.4
CDAN [31]	77.7±0.3	90.7±0.2	97.7±0.3	91.3±0.3	74.2±0.2	94.3±0.3	87.7
SymNets [63]	**80.2±0.3**	93.6±0.2	97.0±0.3	93.4±0.3	78.7±0.3	96.4±0.1	89.9
SAFN+ENT [59]	79.3±0.1	93.3±0.4	96.3±0.4	91.7±0.0	77.6±0.1	95.3±0.1	88.9
CAT [11]	77.2±0.2	91.6±0.3	95.5±0.3	91.3±0.3	75.3±0.6	93.6±0.5	87.3
DANN [14]	75.0±0.6	86.0±0.3	96.2±0.4	87.0±0.5	74.3±0.5	91.5±0.6	85.0
DANN+S (ours)	78.3±0.5	91.0±0.4	96.8±0.2	91.8±0.6	77.7±0.5	95.2±0.5	88.5
RSDA-DANN (ours)	79.2±0.4	93.0±0.2	**98.3±0.4**	93.6±0.4	78.5±0.3	**98.2±0.2**	90.1
MSTN* [58]	77.3±0.3	91.3±0.4	96.8±0.2	91.2±0.5	77.7±0.2	95.0±0.5	88.2
MSTN+S (ours)	78.5±0.5	93.8±0.2	97.0±0.2	93.3±0.6	**79.2±0.3**	95.3±0.2	89.5
RSDA-MSTN (ours)	79.8±0.2	**94.5±0.5**	98.0±0.4	**94.2±0.4**	**79.2±0.3**	97.3±0.3	**90.5**

Table 3. Accuracy (%) on Office-Home for unsupervised domain adaption (ResNet-50). *Reproduced by us.

Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
ResNet-50 [18]	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
CDAN [31]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
SymNets [63]	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
MDD [62]	**54.9**	73.7	77.8	60.0	71.4	71.8	61.2	**53.6**	78.1	72.5	**60.2**	82.3	68.1
DWT-MEC [43]	54.7	72.3	77.2	56.9	68.5	69.8	54.8	47.9	78.1	68.6	54.9	81.2	65.4
SAFN [59]	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
DANN [14]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
DANN+S (ours)	$45.5^{\pm0.1}$	$61.9^{\pm0.3}$	$72.2^{\pm0.5}$	$54.6^{\pm0.2}$	$59.2^{\pm0.4}$	$62.8^{\pm0.3}$	$52.0^{\pm0.2}$	$43.9^{\pm0.1}$	$71.8^{\pm0.4}$	$66.3^{\pm0.5}$	$51.5^{\pm0.2}$	$76.5^{\pm0.4}$	59.8
RSDA-DANN (ours)	$51.5^{\pm0.5}$	$76.8^{\pm0.8}$	$81.1^{\pm0.2}$	$\mathbf{67.1^{\pm0.4}}$	$72.1^{\pm0.2}$	$\mathbf{77.0^{\pm0.6}}$	$64.2^{\pm0.3}$	$51.1^{\pm0.5}$	$81.8^{\pm0.6}$	$74.9^{\pm0.2}$	$55.9^{\pm0.2}$	$84.5^{\pm0.7}$	69.8
MSTN* [58]	49.8	70.3	76.3	60.4	68.5	69.6	61.4	48.9	75.7	70.9	55.0	81.1	65.7
MSTN+S (ours)	$51.9^{\pm0.4}$	$72.3^{\pm0.9}$	$78.3^{\pm0.3}$	$63.7^{\pm0.6}$	$69.9^{\pm0.2}$	$73.5^{\pm0.5}$	$63.5^{\pm0.3}$	$52.1^{\pm0.5}$	$80.2^{\pm0.2}$	$73.6^{\pm0.8}$	$57.7^{\pm0.1}$	$82.7^{\pm0.3}$	68.3
RSDA-MSTN (ours)	$53.2^{\pm0.9}$	$\mathbf{77.7^{\pm1.0}}$	$\mathbf{81.3^{\pm0.3}}$	$66.4^{\pm0.6}$	$\mathbf{74.0^{\pm0.2}}$	$76.5^{\pm0.6}$	$\mathbf{67.9^{\pm0.1}}$	$53.0^{\pm0.1}$	$\mathbf{82.0^{\pm0.5}}$	$\mathbf{75.8^{\pm0.6}}$	$57.8^{\pm0.2}$	$\mathbf{85.4^{\pm0.3}}$	**70.9**

Table 4. Accuracy (%) on VisDA-2017 (ResNet-50).

Method	Synthetic → Real
CDAN [31]	70.0
MDD [62]	74.6
DANN[14]	63.7
DANN+S (ours)	67.6
RSDA-DANN (ours)	**75.8**

On VisDA-2017, RSDA-DANN achieves competitive result (75.8%), outperforming MDD (74.6%) by 1.2%.

Comparison with pseudo-label based methods. As discussed in related works, CAN [21] utilizes target pseudo-labels to estimate contrastive domain discrepancy. The improvement of our method indicates that our robust pseudo-label loss utilizes pseudo-labels more effectively. Compared with iCAN [61], which also defines pseudo-label loss by selecting data based on predicted classification score, our RSDA-MSTN improves its accuracy by 3.1% on ImageCLEF-DA and by 3.8% on Office-31, indicating that our Gaussian-uniform mixture based pseudo-label loss is more reliable to detect wrongly labeled data.

Comparison with normalization based methods. Compared with DWT-MEC [43], our RSDA-MSTN outperforms it by 5.4% on Office-Home. Compared with another normalization based method SAFN+ENT [59], our RSDA-MSTN outperforms it by 4.0%, 1.6%, 3.6% on Office-31, ImageCLEF-DA and Office-Home respectively. In methodology, as discussed in related works, our method performs DA completely in spherical feature space, in which spherical classifier and discriminator are utilized and a robust pseudo-label loss is defined, which is different from above normalization-based DA methods.

8.2. Ablation study

Can pseudo-label loss detect wrongly labeled data? To test whether the robust pseudo-label loss does help detect wrongly labeled target data, we calculate the ratio of correctly labeled samples w.r.t. probability of correct labeling based on Gaussian-uniform model for task **W**→**A** in Office-31 dataset, as shown in Fig. 4. The boxplot shows that probability of correct labeling based on Gaussian-uniform model is a good indicator for identifying truly correct labeling of target data and removing wrongly labeled data.

Effect of robust pseudo-label loss. To test the effectiveness of robust pseudo-label loss, we conduct ablation study for RSDA based on DANN on Office-31, ImageCLEF-DA and Office-Home. We test different combinations of "S", "R" and "E", the meanings of which are discussed in Sect. 8.1. Results in Table 5 show that DANN+S+R+E, *i.e.*, RSDA-DANN, significantly outperforms DANN+S+E by 2.8%, 1.4%, and 1.8% on three datasets respectively,

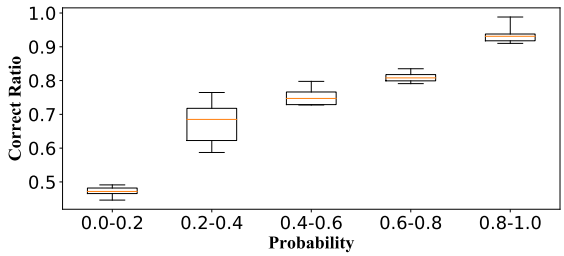


Figure 4. Ratio of truly correct labeling of samples w.r.t. probability of correct labeling based on Gaussian-uniform model for task **W**→**A** in Office-31 dataset. Variations come from updating pseudo labels 10 times during performing alternative optimization.

Table 5. Average accuracy (%) in ablation experiments for RSDA based on DANN. The meanings of notations "S, R, E" are discussed in Sect. 8.1.

Method	Office-31	ImageCLEF-DA	Office-Home
DANN	82.2	85.0	57.6
DANN+S	86.7	88.5	59.8
DANN+R	88.7	89.1	67.3
DANN+S+R	89.2	89.4	68.4
DANN+S+E	87.4	88.7	68.0
DANN+R+E	89.8	89.4	68.8
DANN+S+R+E (RSDA)	**90.2**	**90.1**	**69.8**

demonstrating effectiveness of robust pseudo-label loss. DANN+R, which defines robust pseudo-label loss in Euclidean space, performs significantly better than DANN, indicating that our robust pseudo-label loss also performs well in Euclidean space. But DANN+R degrades performance of DANN+S+R that defined in spherical space. Meanwhile, DANN+S+R+E performs better than DANN+S+R indicates that conditional entropy loss also contributes to performance gain. This shows that conditional entropy (CE) loss is complementary to our robust pseudo-loss that only utilizes a fraction of confident pseudo labels. The CE loss is gradually imposed by increasing γ from 0 to 1 during training, which helps to impose more pseudo-labels guidance when the training progresses.

Effect of adaptation in spherical feature space. In Tables 1, 2, 3 and 4, DANN+S, the meaning of which is discussed in Sect. 8.1, outperforms DANN by 4.5%, 3.5%, 2.2%, 3.9% on four datasets respectively, and MSTN+S outperforms MSTN by 2.8%, 1.7%, 2.6% on Office-31, ImageCLEF-DA and Office-Home respectively, confirming that performing DA in spherical feature space using spherical classifier and discriminator is much better than that in Euclidean space. Moreover, we show in Table 5 that DANN+S+R+E, which is defined in spherical space, outperforms DANN+R+E, indicating that defining robust pseudo-label loss and CE in spherical feature space is also better than defining that in Euclidean space.

Do spherical classifier and discriminator help? To justify usefulness of spherical classifier and discriminator, we conduct ablation studies on Office-31 based on DANN and report results in Table 6. DANN+S (Eucl) denotes normalizing learned features to the sphere but the classifier and discriminator still being Euclidean versions. DANN+S (w/o exp, log) denotes another way to define SP layer by simply normalizing feature after each non-linearity without utilizing spherical exponential and logarithmic maps in spherical networks. The results show DANN+S improves result of DANN+S (Eucl) by 2.0%, demonstrating that the spherical classifier and discriminator are more suitable for spherical features. DANN+S (w/o exp, log) degrades results of DANN+S, which is consistent with our idea that transforming features in spherical space with exponential and logarithmic maps is more reasonable for spheres.

Table 6. Ablation results (%) of spherical classifier and discriminator on Office-31.

Method	A→W	W→A	A→D	D→A	D→W	W→D	Avg
DANN	82.0	67.4	79.7	68.2	96.9	99.1	82.2
DANN+S (Eucl)	87.9	70.3	84.5	67.6	98.1	99.8	84.7
DANN+S (w/o exp, log)	90.8	71.7	87.5	71.3	95.2	100.0	86.1
DANN+S	93.2	71.0	87.5	70.3	98.0	100.0	86.7

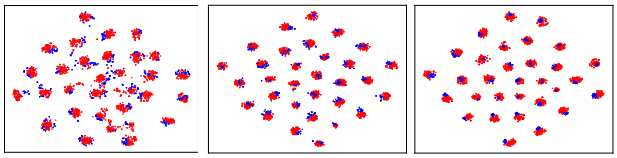


Figure 5. The t-SNE visualization of features learned by DANN (left), DANN+S (middle) and RSDA-DANN (right). Red: source domain **A**. Blue: target domain **W**.

Feature visualization. We visualize features by t-SNE [34] on task **A**→**W** (31 classes) in Fig. 5. It shows that source and target features are aligned better in spherical feature space by DANN+S than DANN in Euclidean feature space. The alignment is further improved by RSDA-DANN, demonstrating effectiveness of our approach.

9. Conclusion

This paper proposed a novel domain adaptation method completely defined in spherical feature space. We designed spherical classifier, discriminator and robust pseudo-label loss in spherical feature space to robustly use pseudo labels. Experiments show that the proposed spherical domain adaption method outperforms Euclidean counterparts, and achieves state-of-the-art results for visual recognition on benchmarks. In future work, we are interested in further analyzing spherical embedding and robust loss for domain adaptation or other applications with weak labels.

Acknowledgment. This work was supported by NSFC (11971373, 11690011, U1811461, 61721002) and National Key R&D Program 2018AAA0102201.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *ML*, 79(1-2):151–175, 2010. 5

[2] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NeurIPS*, 2007. 5

[3] Efrain Castillo-Muñiz and Eduardo Bayro-Corrochano. Geometric spherical networks for visual data processing. In *IJCNN*, 2012. 4

[4] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, 2019. 1, 6

[5] Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. Progressive feature alignment for unsupervised domain adaptation. In *CVPR*, 2019. 1, 2, 6

[6] Safa Cicek and Stefano Soatto. Unsupervised domain adaptation via regularized conditional alignment. In *ICCV*, 2019. 2

[7] Taco Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Convolutional networks for spherical signals. *arXiv preprint arXiv:1709.04893*, 2017. 4

[8] Taco S. Cohen, Mario Geiger, Jonas Khler, and Max Welling. Spherical CNNs. In *ICLR*, 2018. 4

[9] Pietro Coretto and Christian Hennig. Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust gaussian clustering. *JASA*, 111(516):1648–1659, 2016. 4

[10] Hal Daume III and Daniel Marcu. Domain adaptation for statistical classifiers. *JAIR*, 26(1):101–126, 2006. 1

[11] Zhijie Deng, Yucen Luo, and Jun Zhu. Cluster alignment with a teacher for unsupervised domain adaptation. In *ICCV*, 2019. 2, 6, 7

[12] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *arXiv:1805.09112*, 2018. 4

[13] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 1, 2

[14] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016. 1, 2, 5, 6, 7

[15] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *AAAI*, 2017. 3

[16] Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013. 1

[17] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NeurIPS*, 2005. 3

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 2, 6, 7

[19] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 2

[20] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J. Smola. Correcting sample selection bias by unlabeled data. In *NeurIPS*. 2007. 1

[21] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*, 2019. 1, 2, 6, 7

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1

[23] Vinod Kumar Kurmi, Shanu Kumar, and Vinay P. Namboodiri. Attending to discriminative certainty for domain adaptation. In *CVPR*, 2019. 2

[24] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. Deepgum: Learning deep robust regression with a gaussian-uniform mixture model. In *ECCV*, 2018. 4

[25] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, 2013. 3

[26] Seungmin Lee, Dongwan Kim, Namil Kim, and Seong-Gyun Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *ICCV*, 2019. 2

[27] Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *ICML*, 2019. 2

[28] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NeurIPS*, 2017. 2

[29] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1

[30] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 1

[31] Mingsheng Long, ZhangJie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *NeurIPS*, 2018. 1, 2, 6, 7

[32] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *NeurIPS*, 2016. 3

[33] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017. 1, 5

[34] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(Nov):2579–2605, 2008. 8

[35] Jeroen Manders, Elena Marchiori, and Twan van Laarhoven. Simple domain adaptation with class prediction uncertainty alignment. *arXiv:1804.04448*, 2018. 2

[36] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE TNN*, 22(2):199–210, 2010. 1

[37] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE TKDE*, 22(10):1345–1359, 2010. 1

[38] Yingwei Pan, Ting Yao, Yehao Li, Yu Wang, Chong-Wah Ngo, and Tao Mei. Transferrable prototypical networks for unsupervised domain adaptation. In *CVPR*, 2019. 1

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary Devito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS-Workshops*, 2017. 6

[40] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *AAAI*, 2018. 2

[41] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 6

[42] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010. 1

[43] Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Samuel Rota Bulo, Nicu Sebe, and Elisa Ricci. Unsupervised domain adaptation using feature-whitening and consensus loss. In *CVPR*, 2019. 2, 7

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 6

[45] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 5

[46] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *ICCV*, 2019. 1, 2

[47] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, 2017. 1

[48] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018. 2

[49] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-t approach to unsupervised domain adaptation. In *ICLR*, 2018. 3

[50] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 1

[51] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015. 2

[52] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 2

[53] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014. 1

[54] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 5

[55] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 1

[56] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *NeurIPS*. 2019. 1

[57] Ximei Wang, Liang Li, Weirui Ye, Mingsheng Long, and Jianmin Wang. Transferable attention for domain adaptation. In *AAAI*, 2019. 1, 2

[58] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In *ICML*, 2018. 1, 2, 3, 5, 6, 7

[59] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*, 2019. 1, 2, 6, 7

[60] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *CVPR*, 2018. 3

[61] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *CVPR*, 2018. 1, 2, 6, 7

[62] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *ICML*, 2019. 1, 2, 6, 7

[63] Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *CVPR*, 2019. 2, 3, 6, 7