Pattern Recognition 137 (2023) 109271

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Riemannian representation learning for multi-source domain adaptation

Sentao Chen[a,*], Lin Zheng[a], Hanrui Wu[b]

[a] Department of Computer Science, Shantou University, China
[b] College of Information Science and Technology, Jinan University, China

ARTICLE INFO

Article history:
Received 16 December 2022
Accepted 17 December 2022
Available online 22 December 2022

Keywords:
Convex optimization
Hellinger distance
Multi-source domain adaptation
Representation learning
Riemannian manifold

ABSTRACT

Multi-Source Domain Adaptation (MSDA) aims at training a classification model that achieves small target error, by leveraging labeled data from multiple source domains and unlabeled data from a target domain. The source and target domains are described by related but different joint distributions, which lie on a Riemannian manifold named the statistical manifold. In this paper, we characterize the joint distribution difference by the Hellinger distance, which bears strong connection to the Riemannian metric defined on the statistical manifold. We show that the target error of a neural network classification model is upper bounded by the average source error of the model and the average Hellinger distance, i.e., the average of multiple Hellinger distances between the source and target joint distributions in the network representation space. Motivated by the error bound, we introduce Riemannian Representation Learning (RRL): An approach that trains the network model by minimizing (i) the average empirical Hellinger distance with respect to the representation function, and (ii) the average empirical source error with respect to the network model. Specifically, we derive the average empirical Hellinger distance by constructing and solving unconstrained convex optimization problems whose global optimal solutions are easy to find. With the network model trained, we expect it to achieve small error in the target domain. Our experimental results on several image datasets demonstrate that the proposed RRL approach is statistically better than the comparison methods.

1. Introduction

In traditional supervised learning, a fundamental assumption is that the training data and test data are independently drawn from the same joint distribution (domain) $P(\boldsymbol{x}, y)$, where $\boldsymbol{x}$ are the features and y is the class label. Building on this assumption, a classification model (e.g., Convolutional Neural Network, CNN) trained by minimizing the empirical error can obtain a small test error and perform well on the test data [1,2]. However, in many real-world applications (e.g., image recognition), the training and test data could be collected from several domains and are not identically distributed [3–6]. Under such circumstances, a classification model naively trained by minimizing the empirical error, often has an undesirable performance on the test data [7].

Multi-Source Domain Adaptation (MSDA) [7,8] is exactly concerned with such a non-identically-distributed multi-domain learning setting. To be precise, in MSDA one is given n $(n \geq 2)$ labeled datasets respectively drawn from n source joint distributions (source domains) $P^1(\boldsymbol{x}, y), \ldots, P^n(\boldsymbol{x}, y)$, and an unlabeled dataset from a target marginal distribution $P^t(\boldsymbol{x}) = \int P^t(\boldsymbol{x}, y)dy$, where $P^t(\boldsymbol{x}, y)$ is the target joint distribution (target domain). The source and target joint distributions $P^1(\boldsymbol{x}, y), \ldots, P^n(\boldsymbol{x}, y), P^t(\boldsymbol{x}, y)$ are related but different from each other, and the goal of MSDA is to train a classification model that achieves small target error and correctly predicts the target labels. Fig. 1 illustrates the problem of MSDA for image recognition, where the images are obtained from the Office-Home dataset [9].

To address the MSDA problem, previous works propose to align the marginal distributions under various alignment losses to reduce the marginal distribution difference. Specifically, these works perform the alignment of the source marginal distributions $P^1(\boldsymbol{x}), \ldots, P^n(\boldsymbol{x})$ to the target marginal distribution $P^t(\boldsymbol{x})$, i.e., the source-target marginal distribution alignment [7,10–14], or further the alignment within the n source marginal distributions, i.e., the source-source marginal distribution alignment [4,8]. More specifically, they align the marginal distributions under the adversarial loss [7,10,12–15], the Maximum Mean Discrepancy (MMD) [11,16], or the moment distance [4,8]. For instance, Multi-source Domain Adversarial Network (MDAN) [7] aligns n pairs of source

* Corresponding author.
E-mail address: sentaochenmail@gmail.com (S. Chen).

https://doi.org/10.1016/j.patcog.2022.109271
0031-3203/

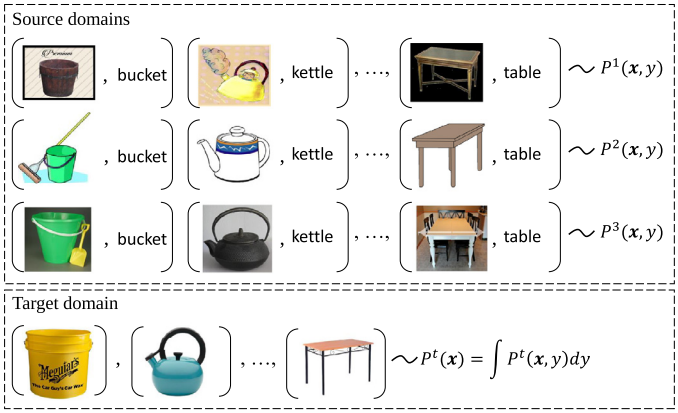


Fig. 1. Multi-Source Domain Adaptation (MSDA) for image recognition, where the three labeled datasets are drawn from source joint distributions $P^1(\boldsymbol{x}, y), P^2(\boldsymbol{x}, y), P^3(\boldsymbol{x}, y)$ and the unlabeled dataset are from target marginal distribution $P^t(\boldsymbol{x})$.

and target marginal distributions in the network representation space under the adversarial loss, and trains the downstream classifier by minimizing the empirical source error. Deep CockTail Network (DCTN) [10] aligns the source marginal distributions to target marginal distribution in an adversarial manner, learns multiple source classifiers, and combines their weighted predictions to classify the target data. In addition, Moment Matching for Multi-Source Domain Adaptation (M^3SDA) [8] minimizes the moment distance to perform both the source-target and source-source marginal distribution alignment in the network representation space, and trains n source classifiers whose outputs are averaged to make predictions for the target data. While these works have yielded promising experimental results, they do not address the joint distribution difference in MSDA, which is closely related to the target error of the neural network classification model. Moreover, their distribution alignment losses do not truly consider the geometry of the space of probability distributions. From information geometry, we know that probability distributions lie on a Riemannian manifold named the statistical manifold. In the series of works [17–19], it has been consistently demonstrated that exploiting the Riemannian metric of the statistical manifold to compare probability distributions is beneficial to the tasks at hand.

In this article, we propose to tackle the MSDA problem by aligning the source joint distributions $P^1(\boldsymbol{x}, y), \ldots, P^n(\boldsymbol{x}, y)$ to the target joint distribution $P^t(\boldsymbol{x}, y)$ to reduce the joint distribution difference, and exploit the Hellinger distance as the distribution alignment loss. As noted and proved by Baktashmotlagh et al. [17], the Hellinger distance bears strong connection to the Riemannian metric defined on the statistical manifold and therefore well reflects the geometrical distance between probability distributions. To be specific, we first show that the target error of a neural network classification model is upper bounded by the average source error of the model and the average Hellinger distance, i.e., the average of n Hellinger distances between the source and target joint distributions in the network representation space. Based on the error bound, we then introduce a Riemannian Representation Learning (RRL) approach, which learns the network model by learning (i) the representation function to align the source joint distributions to the target joint distribution under the average empirical Hellinger distance, and (ii) the downstream classifier to classify the target data. To be more specific, we derive the average empirical Hellinger distance by utilizing the property of a differential convex function and designing an exponential model, which leads to solving unconstrained convex optimization problems whose global optimal solutions are easy to find. Such derivation is different from the derivation in Baktashmotlagh et al. [17], since it avoids estimating the probability distributions, which is known as a difficult problem in statistical learning [20]. We train the neural network model (containing the representation function and the classifier) by the minibatch Stochastic Gradient Descent (SGD) algorithm. With the network model trained, we expect it to achieve small error in the target domain. In a nutshell, we make the following contributions in this article:

- We provide the error bound showing that the target error of a neural network classification model is upper bounded by the average source error of the model and the average Hellinger distance, i.e., the average of n Hellinger distances between the source and target joint distributions in the network representation space.
- We introduce the RRL approach that optimizes the parameters of the network to (i) align the source joint distributions to the target joint distribution in the representation space under the average empirical Hellinger distance, and (ii) learn the downstream classifier for target domain classification.
- We derive the average empirical Hellinger distance by constructing and solving unconstrained convex optimization problems.
- We conduct comprehensive experiments on several image datasets to demonstrate the superior adaptation performance of the proposed RRL approach.

2. Related work

We briefly review the relevant single-source and multi-source domain adaptation works.

2.1. Single-source domain adaptation

Single-Source Domain Adaptation (SSDA) [21] is closely related to MSDA and considers the adaptation problem from a single source joint distribution (domain) $P^s(\boldsymbol{x}, y)$ to a target joint distribution $P^t(\boldsymbol{x}, y)$. To tackle the SSDA problem, existing works propose to align the source and target marginal distributions, i.e., $P^s(\boldsymbol{x})$ and $P^t(\boldsymbol{x})$ [22–26], the source and target class-conditional distributions, i.e., $P^s(\boldsymbol{x}|y)$ and $P^t(\boldsymbol{x}|y)$ [27–29], or the source and target joint distributions [30–32], by minimizing various alignment losses, e.g., the adversarial loss [23,24,26], the MMD [22,28,29], or the L^2-distance [31]. Tzeng et al. [24] utilized the deep model to align the source and target marginal distributions by minimizing the adversarial loss, and concurrently trained a downstream probabilistic classifier. Chen et al. [25] leveraged a distribution adaptation function to align the source marginal distribution to the target marginal distribution under the Bregman divergence, and jointly learn the binary classification model. Cicek et al. [27] aligned the source and target class-conditional distributions in an adversarial manner, encouraged them to have disjoint support, and further employed the semi-supervised learning tools to improve the generalization ability of the deep model. Damodaran et al. [30] used a measure of discrepancy on joint deep representations based on optimal transport, learned representations aligned between the source and target domains, and simultaneously preserved the discriminative information used by the classifier. In the experiments, we compare our MSDA approach with some of the SSDA methods. To ensure that the SSDA methods for comparison can run on the MSDA setting, we combine all the source domains into a single domain for them.

2.2. Multi-source domain adaptation

While the MSDA problem can be naively simplified to the SSDA problem by combining all the source domains into a single do-

main, as noted in several works [5,10,33], such combination ignores the differences among domains and may result in a sub-optimal solution to the problem. Therefore, existing MSDA works develop methods to specifically handle the multi-source problem. Among them, a series of works propose to align the source and target marginal distributions $P^1(\boldsymbol{x}), \ldots, P^n(\boldsymbol{x}), P^t(\boldsymbol{x})$ in various manners, by minimizing the adversarial loss [3,7,10,12–15], the MMD [11,16], or the moment distance [4,8]. Zhao et al. [7] aligned multiple source marginal distributions to the target marginal distribution by adversarial training. Xu et al. [10] deployed multi-way adversarial learning and combined source-specific perplexity scores for target predictions. Peng et al. [8] aligned the first and second order moments of the source and target marginal distributions in the representation space. Zhao et al. [13] proposed the Multi-source Distilling Domain Adaptation (MDDA) network that not only considers the different distances among multiple source marginal distributions and the target marginal distribution, but also investigates the different similarities of the source samples to the target samples. Park et al. [3] proposed the Multi-source Information-regularized Adaptation Networks (MIAN) that does not rely on aligning multiple pairwise marginal distributions, but instead optimizes the mutual information between the latent representations and the domain labels to achieve the alignment of the source and target marginal distributions. Furthermore, Liu et al. [14] aligned the source marginal distributions to the target marginal distribution via adversarial training, performed the category-level alignment by minimizing the distance between the category prototypes and unlabeled target instances, and designed an instance weighting strategy for diverse source instances.

Our work goes beyond both the marginal distribution alignment and the alignment losses practiced in the above works. To be specific, motivated and guided by our target error bound (see Theorem 1 in the following section), we align multiple source joint distributions to the target joint distribution under the average empirical Hellinger distance, rather than the source marginal distributions and target marginal distribution under the above alignment losses. We note that the Hellinger distance is attractive from a geometrical perspective, since it strongly connects to the Riemannian metric defined on the statistical manifold and well reflects the geometrical distance between probability distributions [17].

Apart from the above works, there are also other important ones worth mentioning [33–36]. Venkat et al. [37] utilized implicit alignment without additional training objectives to perform adaptation. Zhou et al. [38] exploited the prototype to transfer the semantic category information from the source domains to the target domain. Li et al. [33] incorporated tensor singular value decomposition into the training process of the MSDA network. Xu et al. [6] used graphical models to realize cross-domain joint modeling and learnable domain combination. Moreover, Deng et al. [39] introduced the idea of viewing each sample as a fine domain and used dynamic neural networks with adaptive convolutional kernels to facilitate the learning of a common feature space. In Section 6, we experimentally compare our work with some of these works for completeness.

3. Statistical manifold and Hellinger distance

We briefly review some concepts of the statistical manifold and the Hellinger distance, which will be used in our approach.

Statistical manifolds are Riemannian manifolds whose elements are probability distributions. Loosely speaking, given a family of distributions $P(\boldsymbol{x}, y; \boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$, the space $\mathcal{M} = \{P(\boldsymbol{x}, y; \boldsymbol{\theta})\}$ forms a Riemannian manifold, and the Fisher–Rao Riemannian metric measures the length of curve between two elements on the manifold $\mathcal{M}$. See Fig. 2 for an illustration of the statistical manifold. Unfortunately, the Fisher–Rao metric can only be

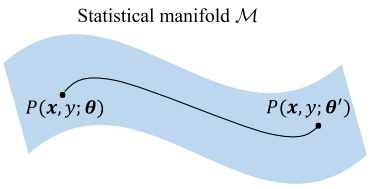


Fig. 2. Illustration of the statistical manifold $\mathcal{M}$. The Fisher–Rao metric is the Riemannian metric that measures the length of curve between two elements $P(\boldsymbol{x}, y; \boldsymbol{\theta})$ and $P(\boldsymbol{x}, y; \boldsymbol{\theta}')$ on the manifold, and the Hellinger distance bears strong connection to the Fisher–Rao Riemannian metric.

calculated for specific parametric distributions, such as Gaussians or mixture of Gaussians. In practical scenarios, it is impractical to compute this metric [17,18]. Therefore, several studies have opted for approximations of the Fisher–Rao metric. Among these approximations, an important one is the Hellinger distance, which is defined as

$$\begin{aligned}&\mathrm{H}(P(\boldsymbol{x}, y; \boldsymbol{\theta}), P(\boldsymbol{x}, y; \boldsymbol{\theta}'))\\ &= \int \left(\sqrt{P(\boldsymbol{x}, y; \boldsymbol{\theta})} - \sqrt{P(\boldsymbol{x}, y; \boldsymbol{\theta}')}\right)^2 d\boldsymbol{x}dy \qquad (1)\\ &= \int 2\left(1 - \sqrt{\frac{P(\boldsymbol{x}, y; \boldsymbol{\theta})}{P(\boldsymbol{x}, y; \boldsymbol{\theta}')}}\right) P(\boldsymbol{x}, y; \boldsymbol{\theta}')d\boldsymbol{x}dy. \qquad (2)\end{aligned}$$

From the definition, we know that the Hellinger distance is non-negative, and is upper bounded by the constant 2. Besides, it is symmetric, satisfies the triangle inequality, and equals to zero if and only if $P(\boldsymbol{x}, y; \boldsymbol{\theta}) = P(\boldsymbol{x}, y; \boldsymbol{\theta}')$. In the work of Baktashmotlagh et al. [17], the authors showed that the length of any given curve on the statistical manifold is the same under the Hellinger distance and under the Fisher–Rao metric up to scale. This nice property strongly connects the Hellinger distance to the Fisher–Rao Riemannian metric, which motivates and encourages us to explore it in this work.

4. Proposed approach

Let $\mathcal{X}$ be a feature space and $\mathcal{Y} = \{1, \ldots, c\}$ be a class label space. We identify a domain with a joint distribution $P(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$. The marginal distribution of a domain is defined as $P(\boldsymbol{x}) = \int P(\boldsymbol{x}, y)dy$. Let h be a classification model from a model space $\mathcal{H}$, and ℓ be a loss function. Additionally, we use $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m \sim P(\boldsymbol{x}, y)$ to denote an independent and identically distributed (i.i.d.) dataset from $P(\boldsymbol{x}, y)$. With these notations, the Multi-Source Domain Adaptation (MSDA) problem is formally defined as:

Definition 1 (MSDA)**.** Let $P^1(\boldsymbol{x}, y), \ldots, P^n(\boldsymbol{x}, y)$ be n ($n \geq 2$) source joint distributions and $P^t(\boldsymbol{x}, y)$ be a target joint distribution, which are related but different from each other. Given n labeled source datasets $\mathcal{D}^1 = \{(\boldsymbol{x}_i^1, y_i^1)\}_{i=1}^{m_1} \sim P^1(\boldsymbol{x}, y)$, …, $\mathcal{D}^n = \{(\boldsymbol{x}_i^n, y_i^n)\}_{i=1}^{m_n} \sim P^n(\boldsymbol{x}, y)$, and an unlabeled target dataset $\mathcal{D}^u = \{\boldsymbol{x}_i^t\}_{i=1}^{m_t} \sim P^t(\boldsymbol{x}) = \int P^t(\boldsymbol{x}, y)dy$, the goal of MSDA is to train a classification model $h \in \mathcal{H}$, such that the target error $\mathbb{E}_{P^t(\boldsymbol{x}, y)}[\ell(h(\boldsymbol{x}), y)]$ is small. Namely, the trained model should perform well in the target domain and correctly predict the target labels.

We model h as a neural network containing a representation function ϕ and a downstream classifier g, i.e., $h = g \circ \phi$, where ϕ maps from the input feature space to the representation space, and g maps from the representation space to the output space. In the following theorem, we show that under mild assumptions, the target error of the neural network classification model is controlled by its average source error and the average Hellinger distance.

Theorem 1. *Assume the loss $\ell \le M$ for some $M > 0$ and the Hellinger distance $\mathrm{H}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y)) \ge \epsilon$ for some $\epsilon > 0$ and $s \in \{1, \ldots, n\}$. Then, for any model $h \in \mathcal{H}$,*

$$\mathbb{E}_{P^t(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}), y)] \le \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})), y)] + \frac{4M}{\epsilon}\frac{1}{n}\sum_{s=1}^{n}\mathrm{H}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y)), \quad (3)$$

where the term $\frac{1}{n}\sum_{s=1}^{n}\mathrm{H}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y))$ is called the average Hellinger distance.

Proof. Please see Appendix A. □

Theorem 1 suggests that to train a network model that achieves small target error, we should try to minimize **(i)** the average source error with respect to (w.r.t.) the representation function ϕ and the classifier g, i.e., the first term in the right hand side of Eq. (3), and **(ii)** the average Hellinger distance w.r.t. the representation function ϕ, i.e., the second term in the right hand side of Eq. (3). Since the average source error and the average Hellinger distance are unknown in practice, we minimize their empirical estimates, which can be computed from the observed samples. For the average empirical source error, it can be straightforwardly computed as $\frac{1}{n}\sum_{s=1}^{n}\frac{1}{m_s}\sum_{i=1}^{m_s}\ell(g(\phi(\boldsymbol{x}_i^s)), y_i^s)$. For the average empirical Helinger distance, we elaborate on its derivation in the following subsection. After that subsection, we introduce our RRL approach that optimizes the representation function and the classifier to jointly minimize the two empirical terms.

4.1. Average empirical Hellinger distance

We show that the average empirical Hellinger distance between the source and target joint distributions can be derived from random samples that reflect the joint distributions. To this end, besides the n source datasets $\mathcal{D}^1 = \{(\boldsymbol{x}_i^1, y_i^1)\}_{i=1}^{m_1} \sim P^1(\boldsymbol{x}, y), \ldots, \mathcal{D}^n = \{(\boldsymbol{x}_i^n, y_i^n)\}_{i=1}^{m_n} \sim P^n(\boldsymbol{x}, y)$, we assume for the moment that a labeled target dataset $\mathcal{D}^t = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{m_t} \sim P^t(\boldsymbol{x}, y)$ is also available. Later in the next subsection, we explain how this set is obtained under the MSDA setting. As a critical part of our derivation in this subsection, the following proposition shows the property of a differential convex function.

Proposition 1. *The differential convex function $f(u) = 2(1 - \sqrt{u})$ has the following property:*

$$2(1 - \sqrt{u}) = \max_{v}[2(1 - \sqrt{v}) - \frac{1}{\sqrt{v}}(u - v)]. \quad (4)$$

Specifically, the maximal value on the right hand side is attained at $v = u$.

Proof. Please see Appendix B. □

To derive the average empirical Hellinger distance, we alternatively express the average Hellinger distance as

$$\frac{1}{n}\sum_{s=1}^{n}\mathrm{H}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y)) = \frac{1}{n}\sum_{s=1}^{n}\int 2\Big(1 - \sqrt{\frac{P^s(\phi(\boldsymbol{x}), y)}{P^t(\phi(\boldsymbol{x}), y)}}\Big)P^t(\phi(\boldsymbol{x}), y)d\phi(\boldsymbol{x})dy \quad (5)$$

$$= \frac{1}{n}\sum_{s=1}^{n}\int \max_{r^s}\Big[2\big(1 - \sqrt{r^s(\phi(\boldsymbol{x}), y)}\big) - \frac{1}{\sqrt{r^s(\phi(\boldsymbol{x}), y)}} \times \Big(\frac{P^s(\phi(\boldsymbol{x}), y)}{P^t(\phi(\boldsymbol{x}), y)} - r^s(\phi(\boldsymbol{x}), y)\Big)\Big]P^t(\phi(\boldsymbol{x}), y)d\phi(\boldsymbol{x})dy \quad (6)$$

$$\ge \frac{1}{n}\sum_{s=1}^{n}\max_{r^s}\int\Big[2\big(1 - \sqrt{r^s(\phi(\boldsymbol{x}), y)}\big) - \frac{1}{\sqrt{r^s(\phi(\boldsymbol{x}), y)}} \times \Big(\frac{P^s(\phi(\boldsymbol{x}), y)}{P^t(\phi(\boldsymbol{x}), y)} - r^s(\phi(\boldsymbol{x}), y)\Big)\Big]P^t(\phi(\boldsymbol{x}), y)d\phi(\boldsymbol{x})dy \quad (7)$$

$$= 2 - \frac{1}{n}\sum_{s=1}^{n}\min_{r^s}\Big(\int\frac{1}{\sqrt{r^s(\phi(\boldsymbol{x}), y)}}P^s(\phi(\boldsymbol{x}), y)d\phi(\boldsymbol{x})dy + \int\sqrt{r^s(\phi(\boldsymbol{x}), y)}P^t(\phi(\boldsymbol{x}), y)d\phi(\boldsymbol{x})dy\Big). \quad (8)$$

In Eq. (6), we leverage Eq. (4) in Proposition 1 and regard the ratio function $\frac{P^s(\phi(\boldsymbol{x}),y)}{P^t(\phi(\boldsymbol{x}),y)}$ as u and the function $r^s(\phi(\boldsymbol{x}), y)$ as v. In Eq. (7), it is straightforward to verify that the equality holds when the function $r^s(\phi(\boldsymbol{x}), y) = \frac{P^s(\phi(\boldsymbol{x}),y)}{P^t(\phi(\boldsymbol{x}),y)}$, and the corresponding maximal value is Eq. (5). Eq. (8) is obtained from simple mathematical calculations.

What makes the expression in Eq. (8) especially useful is that the joint distributions occur linearly in the integral. Therefore, we can replace the integrals in Eq. (8) by sample averages and approximate the average Hellinger distance as

$$\frac{1}{n}\sum_{s=1}^{n}\mathrm{H}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y)) \approx 2 - \frac{1}{n}\sum_{s=1}^{n}\min_{r^s}\Big(\frac{1}{m_s}\sum_{i=1}^{m_s}\frac{1}{\sqrt{r^s(\phi(\boldsymbol{x}_i^s), y_i^s)}} + \frac{1}{m_t}\sum_{i=1}^{m_t}\sqrt{r^s(\phi(\boldsymbol{x}_i^t), y_i^t)}\Big). \quad (9)$$

We design the function $r^s(\phi(\boldsymbol{x}), y)$ as the following exponential model:

$$r^s(\phi(\boldsymbol{x}), y; \boldsymbol{\theta}^s) = e^{2(\boldsymbol{\theta}^s)^\top\Phi(\phi(\boldsymbol{x}),y)}, \quad (10)$$

where $\boldsymbol{\theta}^s = (\theta_1^s, \ldots, \theta_{m_{st}}^s)^\top$ is the parameter vector to be learned and $\Phi(\phi(\boldsymbol{x}), y) = \big(k(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}_1))\delta(y, y_1), \ \ldots, k(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}_i))\delta(y, y_i), \ldots, k(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}_{m_{st}}))\delta(y, y_{m_{st}})\big)^\top$ is the kernel vector. In particular, $(\boldsymbol{x}_i, y_i) \in \mathcal{D}^s \cup \mathcal{D}^t = \{(\boldsymbol{x}_1^s, y_1^s), \ \ldots, (\boldsymbol{x}_{m_t}^t, y_{m_t}^t)\} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{m_{st}}, y_{m_{st}})\}$, where $s \in \{1, \ldots, n\}$ and $m_{st} = m_s + m_t$. The kernel function $k(\phi(\boldsymbol{x}), \phi(\boldsymbol{x}_i)) = e^{-\|\phi(\boldsymbol{x}) - \phi(\boldsymbol{x}_i)\|^2/\sigma}$ is the Gaussian kernel with kernel width σ (> 0)[1], and $\delta(y, y_i)$ is the delta kernel that evaluates 1 if $y = y_i$ and 0 otherwise. Note that, we design $r^s(\phi(\boldsymbol{x}), y)$ as the above exponential model due to the following considerations. (i) $r^s(\phi(\boldsymbol{x}), y)$ should be a positive function such that it has a square root (see Eq. (9)). (ii) With the concrete form of $r^s(\phi(\boldsymbol{x}), y)$, we want the minimization problems in Eq. (9) to be convex problems.

Plugging the exponential model into Eq. (9), we derive and obtain the average empirical Hellinger distance:

$$\frac{1}{n}\sum_{s=1}^{n}\widehat{\mathrm{H}}(P^s(\phi(\boldsymbol{x}), y), P^t(\phi(\boldsymbol{x}), y)) = 2 - \frac{1}{n}\sum_{s=1}^{n}\min_{\boldsymbol{\theta}^s}\Big(\frac{1}{m_s}\sum_{i=1}^{m_s}\frac{1}{\sqrt{r^s(\phi(\boldsymbol{x}_i^s), y_i^s; \boldsymbol{\theta}^s)}} + \frac{1}{m_t}\sum_{i=1}^{m_t}\sqrt{r^s(\phi(\boldsymbol{x}_i^t), y_i^t; \boldsymbol{\theta}^s)}\Big) \quad (11)$$

$$= 2 - \frac{1}{n}\sum_{s=1}^{n}\min_{\boldsymbol{\theta}^s}\Big(\frac{1}{m_s}\sum_{i=1}^{m_s}e^{-(\boldsymbol{\theta}^s)^\top\Phi(\phi(\boldsymbol{x}_i^s), y_i^s)} + \frac{1}{m_t}\sum_{i=1}^{m_t}e^{(\boldsymbol{\theta}^s)^\top\Phi(\phi(\boldsymbol{x}_i^t), y_i^t)}\Big) \quad (12)$$

[1] In the experiments, we set the Gaussian kernel width to the median pairwise squared distances on the unlabeled source and target data.

$$= 2 - \frac{1}{n}\sum_{s=1}^{n}\left(\frac{1}{m_s}\sum_{i=1}^{m_s} e^{-(\widehat{\boldsymbol{\theta}}^s)^\top \Phi(\phi(\boldsymbol{x}_i^s),y_i^s)} + \frac{1}{m_t}\sum_{i=1}^{m_t} e^{(\widehat{\boldsymbol{\theta}}^s)^\top \Phi(\phi(\boldsymbol{x}_i^t),y_i^t)}\right). \tag{13}$$

In Eq. (12), for a fixed index $s \in \{1,\ldots,n\}$ the minimization problem is an unconstrained convex problem. This can be easily verified by observing the objective function, which is the summation of the compositions of the affine function and the exponential function. According to the affine composition rule [40], the objective function is convex in its variable $\boldsymbol{\theta}^s$. In Eq. (13), $\widehat{\boldsymbol{\theta}}^s$ is the global optimal solution[2] returned by the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [41]. This is a popular and mature algorithm for solving convex problems and is widely available in many optimization packages.

Remark 1. We reveal an interesting relationship between the empirical Hellinger distance and the exponential loss minimization, which is an instance of Vapnik's Empirical Risk Minimization (ERM) framework [20]. According to Eq. (12), the empirical Hellinger distance is written as

$$\widehat{\mathrm{H}}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)) = 2 - \min_{\boldsymbol{\theta}^s}\left(\frac{1}{m_s}\sum_{i=1}^{m_s} e^{-(\boldsymbol{\theta}^s)^\top \Phi(\phi(\boldsymbol{x}_i^s),y_i^s)} + \frac{1}{m_t}\sum_{i=1}^{m_t} e^{(\boldsymbol{\theta}^s)^\top \Phi(\phi(\boldsymbol{x}_i^t),y_i^t)}\right). \tag{14}$$

To see the relationship, let us assign domain labels $l_i = +1$ to $(\phi(\boldsymbol{x}_i^s), y_i^s)$ for $i = 1,\ldots,m_s$ and $l_i = -1$ to $(\phi(\boldsymbol{x}_i^t), y_i^t)$ for $i = 1,\ldots,m_t$. If $m_s = m_t$, the above minimization problem for deriving the empirical Hellinger distance coincides with the exponential loss minimization with loss function $e^{-lf(\phi(\boldsymbol{x}),y;\boldsymbol{\theta}^s)}$ and binary classifier $f(\phi(\boldsymbol{x}),y;\boldsymbol{\theta}^s) = (\boldsymbol{\theta}^s)^\top \Phi(\phi(\boldsymbol{x}),y)$. If $m_s \neq m_t$, the problem corresponds to the weighted exponential loss minimization with weight $\frac{1}{m_s}$ for $\{(\phi(\boldsymbol{x}_i^s), y_i^s)\}_{i=1}^{m_s}$ and $\frac{1}{m_t}$ for $\{(\phi(\boldsymbol{x}_i^t), y_i^t)\}_{i=1}^{m_t}$.

4.2. Riemannian representation learning

Before proceeding to present the optimization problem of our approach, we first explain how the labeled target dataset $\mathcal{D}^t$ used in the previous subsection is obtained under the MSDA setting. To be specific, we utilize the commonly practiced pseudo labeling technique [5,16,42] to endow the unlabeled target dataset $\mathcal{D}^u = \{\boldsymbol{x}_i^t\}_{i=1}^{m_t}$ with pseudo labels and set $\mathcal{D}^t = \{(\boldsymbol{x}_i^t, y_i^t)\}_{i=1}^{m_t}$, where y_i^t is the pseudo label predicted by the network classification model trained on the labeled source datasets.

Our Riemannian Representation Learning (RRL) approach optimizes the representation function ϕ and the classifier g of the network model $h = g \circ \phi$ to jointly minimize the average empirical source error and the average empirical Hellinger distance. The optimization problem with objective function $F(g,\phi)$ is presented as follows

$$\min_{\phi,g} F(g,\phi) = \frac{1}{n}\sum_{s=1}^{n}\frac{1}{m_s}\sum_{i=1}^{m_s}\ell(g(\phi(\boldsymbol{x}_i^s)),y_i^s) + \frac{\lambda}{n}\sum_{s=1}^{n}\widehat{\mathrm{H}}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)). \tag{15}$$

Here, ℓ is the cross-entropy loss and λ is a tradeoff parameter for balancing the two loss terms. Note that, the network model can be one of the popular Convolutional Neural Networks (CNNs), e.g., AlexNet [1], ResNet [2].

[2] To avoid overfitting in practice, a penalty term $\gamma\|\boldsymbol{\theta}^s\|^2$ is added to the objective function when running the optimization algorithm, where γ is a small positive value set to 10^{-3}.

We employ the minibatch Stochastic Gradient Descent (SGD) algorithm to solve problem (15). In every iteration of the algorithm, a minibatch consists of $n+1$ minibatches respectively sampled from the n labeled source datasets $\mathcal{D}^1,\ldots,\mathcal{D}^n$ and the pseudo labeled target dataset $\mathcal{D}^t$, and the objective function $F(g,\phi)$ is calculated using these minibatches. In the very beginning, since the source joint distributions are not aligned to the target joint distribution in the network representation space, the pseudo target labels predicted by the source trained network model may be quite different from the true target labels. As a result, in the MSDA setting, the average empirical Hellinger distance that relies on the pseudo target labels, may not well reflect the average Hellinger distance. To address this issue, we update the pseudo target labels after the network parameters are updated. With a better trained network, the source joint distributions are better aligned to the target joint distribution in the representation space, and consequently the updated pseudo target labels could better approximate the true target labels (see the empirical evidence in Section 6.5.3). For clarity, we provide the pseudo code for training our RRL approach in Algorithm 1.

Algorithm 1 Pseudo code for training our RRL approach.

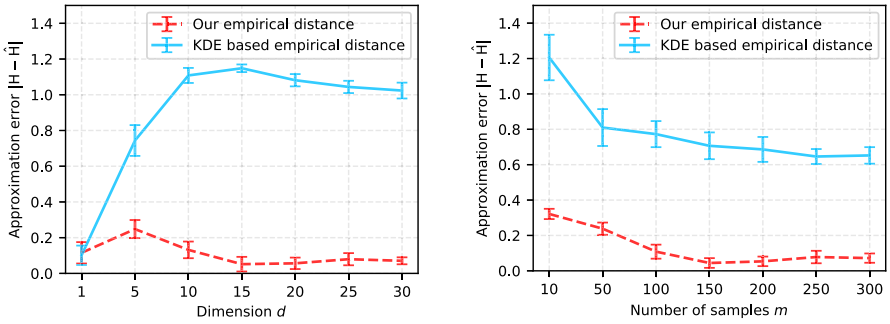
Input: Labeled source datasets $\mathcal{D}^1,\ldots,\mathcal{D}^n$, unlabeled target dataset $\mathcal{D}^u$.
Output: Trained network $\hat{h} = \hat{g} \circ \hat{\phi}$.

```
Train a network on the source datasets D^1, ..., D^n.
Use the network to label D^u and obtain pseudo labeled target dataset D^t.
while training does not end do
    for k in 1 : K do
        Sample minibatches D_k^1, ..., D_k^n, D_k^t from datasets D^1, ..., D^n, D^t.
        Calculate the objective function F(g, φ) in (15) using D_k^1, ..., D_k^n, D_k^t, where the second term of F(g, φ) is computed by applying the L-BFGS algorithm [41].
        Take a gradient step to update the parameters of the representation function φ and the classifier g.
    end for
    Use the network to update the pseudo labels in the target dataset D^t.
end while
```

5. Discussion

5.1. On the error bound

We discuss the difference between our target error bound in Theorem 1 and the ones proposed in prior MSDA works [4,5,7,8,12]. Most previous works analyze the target error by following the work of Ben-David et al. [43]. They concentrate on the binary classification problem, factorize the joint distribution $P(\boldsymbol{x},y)$ into the product of marginal distribution $P(\boldsymbol{x})$ and posterior distribution $P(y|\boldsymbol{x})$, i.e., $P(\boldsymbol{x},y) = P(\boldsymbol{x})P(y|\boldsymbol{x})$, and assume the posterior distribution is a deterministic function $f(\boldsymbol{x})$, i.e., $P(y = f(\boldsymbol{x})|\boldsymbol{x}) = 1$. Then, these works bound the target error by the $\mathcal{H}\Delta\mathcal{H}$-divergence [5,7], the discrepancy distance [12], or the moment distance [4,8] w.r.t. the source and target marginal distributions. However, many real-world classification problems are multi-class problems (e.g., face recognition, object recognition), and the posterior distribution may not necessarily be a deterministic function for most datasets [44]. In comparison, our analysis of the target error applies to the multi-class classification problem, works on the joint distributions, and bounds the target error by the Hellinger distance w.r.t. the source and target joint distributions. Importantly, unlike previous works



(a) Different dimensions (b) Different sample sizes

Fig. 3. Mean and standard deviation of approximation error $|H - \hat{H}|$ that measures the quality of an empirical Hellinger distance. (a) Approximation error with different dimensions. (b) Approximation error with different numbers of samples.

where there is a gap between theory and algorithm, our target error bound directly motivates and guides the design of our RRL approach.

5.2. On the Hellinger distance

We discuss the relationship and difference between our work and the work of Baktashmotlagh et al. [17], with a focus on the Hellinger distance. (i) Relationship. Our work relates to [17] since both works share the same motivation in employing the Hellinger distance. Namely, the Hellinger distance strongly connects to the Riemannian metric defined on the statistical manifold and can better reflect the geometrical distance between distributions, compared to other metrics like the MMD or the L^2-distance. (ii) Difference. Our derivation of the empirical Hellinger distance as the distribution alignment loss differs from the derivation in Baktashmotlagh et al. [17]. To be specific, Baktashmotlagh et al. [17] uses a two-step procedure to derive the empirical Hellinger distance, which consists of first estimating the distributions via Kernel Density Estimation (KDE) [45] and then computing the empirical distance based on the estimated distributions. However, this two-step procedure may result in a poor empirical distance, since the first step is conducted without regard to the second step and thus a small estimation error incurred in the first step can induce a big error in the second step. Besides, the KDE for distribution estimation is known to suffer from the curse of dimensionality and may not be reliable in high-dimensional spaces [45]. By contrast, our work directly derives the empirical Hellinger distance by solving the unconstrained convex optimization problem, i.e., Eq. (14), and circumvents the difficult problem of distribution estimation.

To show that our empirical Hellinger distance is better than the KDE based empirical Hellinger distance [17] in approximating the true distance, we conduct experiments to compare the two empirical distances under different dimensions and different numbers of samples. Particularly, we consider the Hellinger distance between two Gaussian distributions $P^s(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \mathbf{1}_d, \mathbf{I}_d)$ and $P^t(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \mathbf{0}_d, \mathbf{I}_d)$, where the mean vectors $\mathbf{1}_d$ and $\mathbf{0}_d$ are d-dimensional vectors of ones and zeros, and the covariance matrix $\mathbf{I}_d$ is the identity matrix. According to Pardo [46], this distance can be analytically computed as $H = H(P^s(\boldsymbol{x}), P^t(\boldsymbol{x})) = 2 - 2e^{-d/8}$. Therefore, we can measure the quality of an empirical Hellinger distance by computing its approximation error, which is defined as the absolute difference between the true distance H and the empirical distance $\hat{H}$, i.e., $|H - \hat{H}|$. We draw the same number of samples from the two Gaussians to compute the empirical Hellinger distance, and repeat the sampling procedure 10 times to calculate the mean and standard deviation of the approximation error. Fig. 3(a) and (b) plot the approximation error curves of our empirical distance and the KDE based empirical distance [17], by varying the dimension d and the number of samples m, respectively. We observe from Fig. 3(a), and (b) that the approximation error of our empirical distance is much smaller than the error of the KDE based empirical distance under different dimensions and different numbers of samples. This confirms that our empirical Hellinger distance is superior to the KDE based empirical Hellinger distance.

6. Experiments

We evaluate our RRL approach on several popular image datasets, and contrast its performance against the SSDA and MSDA methods. The Pytorch implementation (source code) of our approach is available at the link https://github.com/sentaochen/Riemannian-Representation-Learning. In the following subsections, we start by describing the datasets, then introduce the experimental setup, present the experimental results, perform the statistical test, and finally finish by conducting the experimental analysis.

6.1. Datasets

ImageCLEF [3] contains images from 3 domains: Caltech-256 (**Ca**) with 600 images, ImageNet ILSVRC 2012 (**Im**) with 600 images, and Pascal VOC 2012 (**Pa**) with 600 images. See Fig. 4(a) for the example images. The task is classification with 12 classes.

Office [47] includes object images taken from 3 domains: Amazon (**Am**) with 2817 images, DSLR (**Ds**) with 498 images, and Webcam (**We**) with 795 images. See Fig. 4(b) for the example images. The task is classification with 31 classes.

Office-Home [9] contains images of everyday objects organized into 4 domains: Art (**Ar**) with 2421 images, Clipart (**Cl**) with 4379 images, Product (**Pr**) with 4428 images, and RealWorld (**Re**) with 4357 images. See Fig. 4(c) for the example images. The task is classification with 65 classes.

Office-Caltech [48] consists of images from 4 domains: Amazon (**Am**) with 958 images, Caltech (**Ca**) with 1123 images, DSLR (**Ds**) with 157 images, and Webcam (**We**) with 295 images. See Fig. 4(b) for the example images. The task is classification with 10 classes.

DomainNet [8] includes images from 6 domains: Clipart (**Cl**), Infograph (**In**), Painting (**Pa**), Quickdraw (**Qu**), Real (**Re**), and Sketch (**Sk**). It has around 0.6 million images and a large number of classes. See Fig. 4(d) for the example images. The task is classification with 345 classes.

[3] https://www.imageclef.org/2014/adaptation.

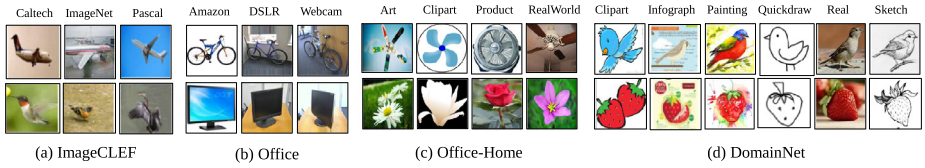


Fig. 4. Example images from datasets ImageCLEF, Office, Office-Home, and DomainNet.

Table 1
Configuration of network models for the datasets.

Dataset	ImageCLEF	Office	Office-Home	Office-Caltech	DomainNet
Model	ResNet50	ResNet50	ResNet50	ResNet101	ResNet101

6.2. Experimental setup

We compare our approach with the following 3 types of methods. **(i)** The Baseline method. It combines all the source domains into a single domain and trains the neural network model on the resulting source domain via cross-entropy loss minimization. **(ii)** The SSDA methods. They contain Domain Adaptation Network (DAN) [22], Domain-Adversarial Neural Network (DANN) [23], f-Domain-Adversarial Learning (f-DAL) [49]. To ensure that the SSDA methods can run on the MSDA setting, all the source domains are combined into a single domain. **(iii)** The MSDA methods. They include Multi-source Domain Adversarial Network (MDAN) [7], Moment Matching for Multi-Source Domain Adaptation (M^3SDA) [8], Multi-source Distilling Domain Adaptation (MDDA) [13], Domain AggRegation Network (DARN) [12], Curriculum Manager for Source Selection (CMSS) [35], Multi-source Information-regularized Adaptation Networks (MIAN) [3], and Tensor Singular Value Decomposition Net (T-SVDNet) [33].

We construct the MSDA task as follows. For a certain dataset (e.g., ImageCLEF), one of its domains is selected as the target domain and the remaining ones are selected as the source domains. For convenience, we denote such a task as "→ Ca" where Ca indicates the target domain. The performance of a method is measured by its classification accuracy (%) on the target domain. On each task, we repeat the experiment 5 times with different random seeds to report the mean and standard deviation of the classification accuracy.

We implement the neural network classification model as the CNN model, and present in Table 1 an overview of the configuration of network models for the datasets. To be specific, on datasets ImageCLEF, Office, and Office-Home, the network models are the ResNet50 model [2] following prior works [3,16]. On datasets Office-Caltech and DomainNet, the models are ResNet101 [2] following prior works [8,33]. On each dataset, the final layer of the network is reconstructed to have the same number of outputs as the number of classes in that dataset (12 for ImageCLEF, 31 for Office, 65 for Office-Home, 10 for Office-Caltech, and 345 for DomainNet).

We train the networks by the minibatch SGD algorithm, where the network parameters are pretrained on the ImageNet dataset. Since the classifier layer is trained from scratch, its learning rate is set to 10 times the learning rate of the representation function, i.e., 0.001 for the representation function and 0.01 for the classifier. As aforementioned, in every iteration of the algorithm, a minibatch consists of $n+1$ minibatches of the same size, which are respectively sampled from the n labeled source datasets and the pseudo labeled target dataset. For the tradeoff parameter λ, following the strategy in Ganin et al. [23], we gradually change it from 0 to 1 using the formula $\lambda_p = \frac{2}{1+e^{-10p}} - 1$, where p is the training progress linearly changing from 0 to 1. According to Ganin et al. [23], this strategy can suppress noisy signals at the early stages of training.

6.3. Experimental results

We report in Table 2 the classification results on ImageCLEF, in Table 3 the results on Office, in Table 4 the results on Office-Home, in Table 5 the results on Office-Caltech, and in Table 6 the results on DomainNet. Since our experimental settings coincide with prior works, we therefore directly quote the results of some comparison methods from Park and Lee [3], Li et al. [33], Yang et al. [35]. Moreover, for the comparison methods whose results on a certain dataset are not available, we use their released source codes, follow their hyperparameter tuning protocols, and produce their results on that dataset. With the complete classification results on each dataset, we can better compare our approach with others.

Table 2
Classification accuracy (%) of different methods with ResNet50 model on dataset ImageCLEF. In each column, the best result is highlighted in **bold**, and the second best is underlined.

Method	→Ca	→Im	→Pa	Avg
Baseline	91.96 (0.24)	88.23 (0.10)	77.42 (0.13)	85.87
DAN [22]	93.15 (0.17)	92.76 (0.16)	77.47 (0.08)	87.79
DANN [23]	93.78 (0.11)	91.70 (0.26)	77.85 (0.32)	87.78
f-DAL [49]	95.24 (0.21)	93.71 (0.20)	78.05 (0.18)	89.00
MDAN [7]	93.61 (0.35)	91.45 (0.32)	77.27 (0.24)	87.44
M^3SDA [8]	94.25 (0.20)	91.90 (0.31)	77.72 (0.10)	87.96
MDDA [13]	93.63 (0.30)	91.72 (0.15)	77.44 (0.27)	87.60
DARN [12]	93.85 (0.42)	90.63 (0.24)	77.02 (0.16)	87.17
CMSS [35]	95.78 (0.31)	92.34 (0.42)	77.95 (0.28)	88.69
MIAN [3]	95.14 (0.28)	91.45 (0.17)	77.56 (0.40)	88.05
T-SVDNet [33]	95.46 (0.30)	93.67 (0.22)	78.87 (0.35)	89.33
RRL (ours)	**97.63 (0.23)**	**96.86 (0.15)**	**79.98 (0.30)**	**91.50**

Table 3
Classification accuracy (%) of different methods with ResNet50 model on dataset Office. In each column, the best result is highlighted in **bold**, and the second best is underlined.

Method	→Am	→Ds	→We	Avg
Baseline	66.92 (0.45)	99.52 (0.16)	96.22 (0.29)	87.56
DAN [22]	67.69 (0.56)	99.60 (0.26)	96.68 (0.44)	87.99
DANN [23]	67.97 (0.38)	99.71 (0.37)	96.64 (0.26)	88.10
f-DAL [49]	73.40 (0.28)	99.46 (0.10)	98.25 (0.38)	90.37
MDAN [7]	66.12 (0.32)	99.68 (0.16)	97.86 (0.15)	87.88
M^3SDA [8]	69.41 (0.82)	99.64 (0.19)	99.30 (0.31)	89.45
MDDA [13]	67.93 (0.23)	99.60 (0.22)	97.32 (0.35)	88.28
DARN [12]	66.36 (0.43)	99.84 (0.08)	98.67 (0.19)	88.29
CMSS [35]	72.26 (0.27)	99.66 (0.39)	97.53 (0.47)	89.82
MIAN [3]	76.17 (0.24)	99.22 (0.35)	98.39 (0.76)	91.26
T-SVDNet [33]	74.07 (0.38)	99.42 (0.61)	**99.63 (0.18)**	91.04
RRL (ours)	**77.45 (0.26)**	**99.92 (0.10)**	99.42 (0.38)	**92.26**

Table 4
Classification accuracy (%) of different methods with ResNet50 model on dataset Office-Home. In each column, the best result is highlighted in **bold**, and the second best is underlined.

Method	→Ar	→Cl	→Pr	→Re	Avg
Baseline	67.40 (0.39)	52.38 (0.38)	77.95 (0.28)	80.70 (0.17)	69.61
DAN [22]	69.62 (0.87)	56.50 (1.36)	79.66 (0.65)	83.16 (0.40)	72.23
DANN [23]	69.19 (0.58)	57.08 (0.19)	79.18 (0.70)	82.63 (0.40)	72.02
f-DAL [49]	72.30 (0.47)	64.26 (0.39)	83.15 (0.41)	82.81 (0.30)	75.63
MDAN [7]	69.57 (0.36)	55.22 (0.46)	79.71 (0.29)	81.48 (0.19)	71.49
M^3SDA [8]	66.22 (0.52)	58.55 (0.62)	79.45 (0.52)	81.35 (0.19)	71.39
MDDA [13]	68.79 (0.39)	57.08 (0.51)	78.47 (0.33)	80.68 (0.21)	71.25
DARN [12]	68.93 (0.44)	55.64 (0.61)	79.72 (0.27)	82.45 (0.31)	71.68
CMSS [35]	71.94 (0.64)	58.64 (0.65)	80.14 (0.59)	82.01 (0.37)	73.18
MIAN [3]	69.88 (0.35)	64.20 (0.68)	80.87 (0.37)	81.49 (0.24)	74.11
T-SVDNet [33]	71.94 (0.30)	65.06 (0.59)	82.59 (0.35)	81.79 (0.03)	75.34
RRL (ours)	**75.02 (0.18)**	**68.93 (0.42)**	**85.61 (0.19)**	**84.05 (0.36)**	**78.40**

Table 5
Classification accuracy (%) of different methods with ResNet101 model on dataset Office-Caltech. We cite the classification accuracy of some comparison methods from [35], and set the standard deviations to 0.00, since these values are not available in the reference. In each column, the best result is highlighted in **bold**, and the second best is underlined.

Method	→Am	→Ca	→Ds	→We	Avg
Baseline	86.10 (0.00)	87.80 (0.00)	98.30 (0.00)	99.00 (0.00)	92.80
DAN [22]	94.80 (0.00)	89.70 (0.00)	98.20 (0.00)	99.30 (0.00)	95.50
DANN [23]	94.80 (0.00)	89.70 (0.00)	98.20 (0.00)	99.30 (0.00)	95.50
f-DAL [49]	95.40 (0.18)	95.05 (0.09)	**100.00 (0.00)**	99.45 (0.22)	97.48
MDAN [7]	95.40 (0.00)	91.80 (0.00)	98.60 (0.00)	98.90 (0.00)	96.10
M^3SDA [8]	94.50 (0.00)	92.20 (0.00)	99.20 (0.00)	99.50 (0.00)	96.40
MDDA [13]	95.46 (0.24)	91.64 (0.27)	98.97 (0.12)	99.42 (0.18)	96.37
DARN [12]	95.65 (0.42)	91.77 (0.33)	99.35 (0.11)	99.20 (0.22)	96.50
CMSS [35]	96.00 (0.00)	93.70 (0.00)	99.30 (0.00)	99.60 (0.00)	97.20
MIAN [3]	96.05 (0.07)	94.58 (0.13)	99.02 (0.05)	99.32 (0.17)	97.24
T-SVDNet [33]	**96.75 (0.34)**	93.87 (0.19)	**100.00 (0.00)**	99.50 (0.25)	97.53
RRL (ours)	96.32 (0.28)	**96.18 (0.15)**	**100.00 (0.00)**	**99.71 (0.15)**	**98.05**

Table 6
Classification accuracy (%) of different methods with ResNet101 model on dataset DomainNet. In each column, the best result is highlighted in **bold**, and the second best is underlined.

Method	→Cl	→In	→Pa	→Qu	→Re	→Sk	Avg
Baseline	47.60 (0.52)	13.10 (0.41)	38.10 (0.45)	13.30 (0.39)	51.90 (0.85)	33.70 (0.54)	32.95
DAN [22]	60.23 (0.29)	26.52 (0.48)	51.14 (0.75)	8.22 (0.47)	61.01 (0.56)	50.23 (0.84)	42.89
DANN [23]	60.60 (0.42)	25.80 (0.34)	50.40 (0.51)	7.70 (0.68)	62.00 (0.66)	51.70 (0.19)	43.00
f-DAL [49]	64.20 (0.31)	24.50 (0.53)	53.10 (0.70)	14.60 (0.62)	61.20 (0.67)	53.20 (0.51)	45.13
MDAN [7]	60.30 (0.41)	25.00 (0.43)	50.30 (0.36)	8.20 (1.92)	61.50 (0.46)	51.30 (0.58)	42.80
M^3SDA [8]	58.60 (0.53)	26.00 (0.89)	52.30 (0.55)	6.30 (0.58)	62.70 (0.51)	49.50 (0.76)	42.60
MDDA [13]	59.40 (0.60)	23.80 (0.80)	53.20 (0.60)	12.50 (0.60)	61.80 (0.50)	48.60 (0.80)	43.20
DARN [12]	59.24 (0.32)	24.78 (0.56)	51.25 (0.66)	8.91 (0.72)	61.88 (0.63)	51.55 (0.38)	42.94
CMSS [35]	64.20 (0.18)	28.00 (0.20)	53.60 (0.39)	16.00 (0.12)	63.40 (0.21)	53.80 (0.35)	46.50
MIAN [3]	60.57 (0.54)	24.45 (0.46)	55.25 (0.58)	14.88 (0.64)	62.78 (0.52)	50.76 (0.48)	44.78
T-SVDNet [33]	66.10 (0.40)	25.00 (0.80)	54.30 (0.70)	16.50 (0.90)	65.40 (0.50)	54.60 (0.60)	47.00
RRL (ours)	**69.54 (0.59)**	**28.35 (0.53)**	**59.21 (0.41)**	**29.48 (0.44)**	**69.44 (0.64)**	**62.33 (0.57)**	**53.06**

From Tables 2 to 6, we observe that on majority of the tasks, our RRL approach unambiguously outperforms the Baseline, SSDA, and MSDA methods. We also observe that some comparison methods, including f-DAL, MIAN, and T-SVDNet, have delivered promising classification results. However, these methods (i) do not address the crucial joint distribution difference problem in MSDA, and (ii) are not aware of the geometrical distance between probability distributions. Hence, on the challenging datasets Office-Home and DomainNet with very different domains (joint distributions), their results are significantly below the results of our RRL approach. Overall, the experimental results from Tables 2 to 6 confirm that for addressing the MSDA problem, our approach that aligns multiple source joint distributions to the target joint distribution under the average empirical Hellinger distance, is more advantageous than the comparison methods that exploit other strategies to tackle the problem (e.g., marginal distribution alignment under various losses [3,7,8,12], source selection [35], tensor singular value decomposition [33]).

6.4. Statistical test

To be strict in a statistical sense, we conduct statistical tests to check whether our RRL approach is significantly better than the comparison methods. We conduct the Wilcoxon signed-ranks test [50] based on the classification results from Tables 2 to 6. The test uses a statistic T to compare the performance of two methods

Table 7
Values of statistic T for the Wilcoxon signed-ranks tests of the comparison methods versus RRL.

Method	Baseline	DAN [22]	DANN [23]	f-DAL [49]	MDAN [7]	$^{M^3}$SDA [8]
T value	0.00	0.00	0.00	0.50	0.00	0.00
Method	MDDA [13]	DARN [12]	CMSS [35]	MIAN [3]	T-SVDNet [33]	–
T value	0.00	0.00	0.00	0.00	7.00	–

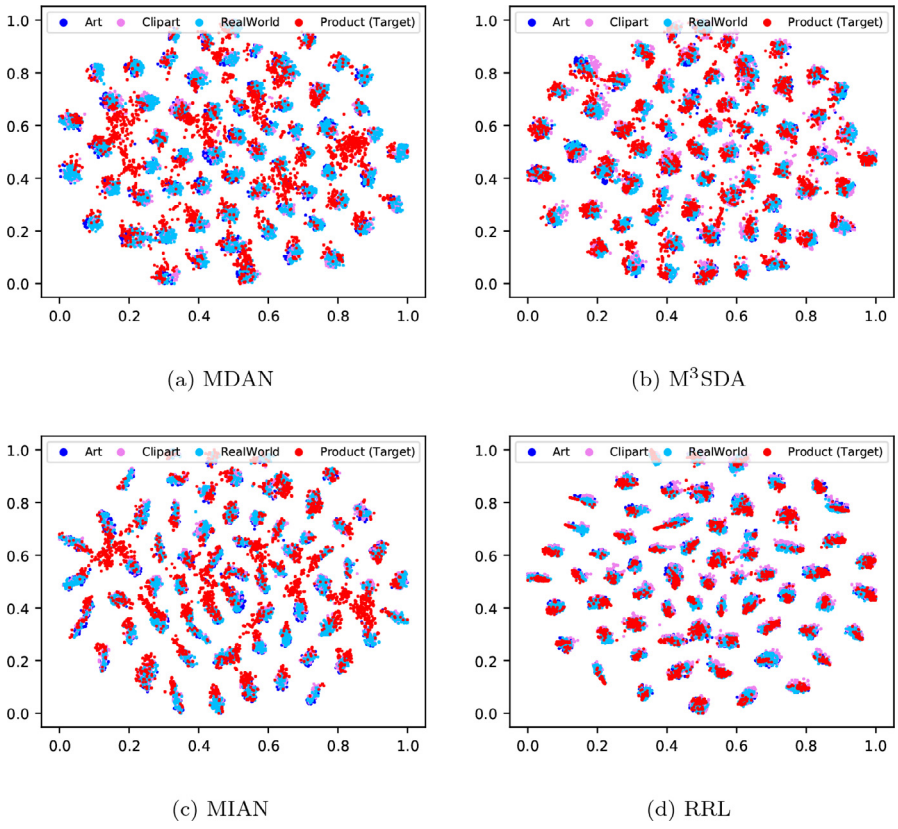


Fig. 5. T-SNE visualization of source and target data in the representation spaces of MDAN, $^{M^3}$SDA, MIAN, and RRL with ResNet50 model. The source domains are Art, Clipart, and RealWorld, and the target domain is Product.

over N tasks. Specifically, in each task the classification accuracy is adopted as the performance measure of the methods. We fix RRL as a control method, and conduct 11 pairs of tests: Baseline versus RRL, ..., T-SVDNet versus RRL. The detailed description of the test procedure is presented in Appendix C, and the resulting T values are reported in Table 7. We observe from Table 7 that the T values for the 11 pairs of tests are all below the value 52, which is the critical value of Wilcoxon's test with the confidence level $\alpha = 0.05$ and the number of tasks $N = 20$. According to Demšar [50], this indicates that the RRL approach is statistically better than the comparison methods.

6.5. Experimental analysis

6.5.1. Feature visualization

We exploit the t-SNE visualization tool [51] and visualize in Fig. 5(a)–(d) the representations of task "→ Pr" (Office-Home) generated by MDAN, $^{M^3}$SDA, MIAN, and RRL. Comparing Fig. 5(d) against Fig. 5(a)–(c), we observe that our RRL approach better aligns the source and target data in the network representation space than its competitors MDAN, $^{M^3}$SDA, and MIAN. These comparison results suggest that joint distribution alignment under the Hellinger distance is a powerful approach to MSDA.

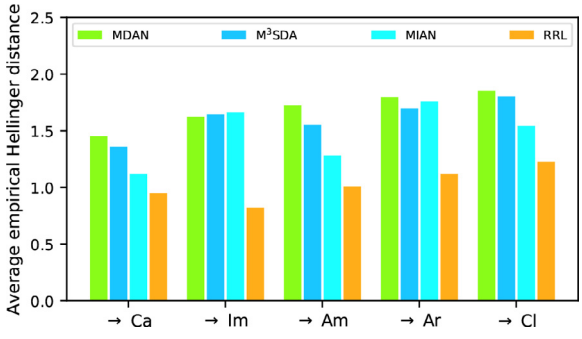
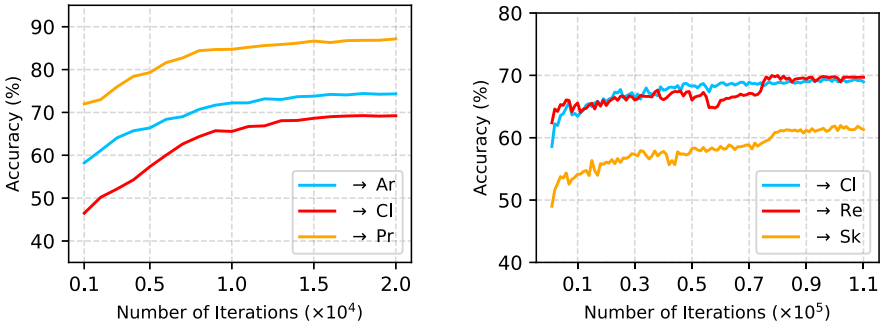


Fig. 6. The average empirical Hellinger distances on 5 tasks "→ Ca" and "→ Im" from ImageCLEF, "→ Am" from Office, and "→ Ar" and "→ Cl" from Office-Home.

6.5.2. Average empirical Hellinger distance

We compute the average empirical Hellinger distance via Eq. (13) using the representations from MDAN, M^{3}SDA, MIAN, and RRL, and plot in Fig. 6 the resulting values on 5 tasks from datasets ImageCLEF, Office, and Office-Home. Clearly, on the 5 tasks, the distances computed from our RRL representations are smaller than those from the MDAN, M^{3}SDA, or MIAN representations. With smaller average empirical Hellinger distances, our RRL approach



(a) Office-Home (b) DomainNet

Fig. 7. Variation curves of classification accuracy of the RRL approach. (a) Results on tasks from Office-Home. (b) Results on tasks from DomainNet.

Table 8
Ablation study results (%) of the RRL approach on dataset Office-Home.

Method	→Ar	→Cl	→Pr	→Re	Avg
Baseline	67.40 (0.39)	52.38 (0.38)	77.95 (0.28)	80.70 (0.17)	69.61
RRL-marginal	69.57 (0.43)	64.31 (0.27)	81.18 (0.55)	81.89 (0.36)	74.24
RRL-L^2	71.72 (0.56)	63.98 (0.27)	83.07 (0.32)	82.78 (0.45)	75.39
RRL	**75.02 (0.18)**	**68.93 (0.42)**	**85.61 (0.19)**	**84.05 (0.36)**	**78.40**

thus leads to better classification results in the target domain (again see Theorem 1 which shows that the target error is controlled by the average Hellinger distance).

6.5.3. Pseudo labeling

We verify that iteratively updating the pseudo target labels in our RRL approach will refine the labels and make them better approximate the true target labels. To this end, we monitor the variation of target classification accuracy on tasks from datasets Office-Home and DomainNet. The variation curves are plotted in Fig. 7(a) and (b). For all the tasks in the 2 figures, the target classification accuracy tends to grow higher and higher as the iteration proceeds, indicating that the pseudo target labels are refined and better approximate the true labels.

6.5.4. Ablation study

We conduct ablation study to investigate the contributions of the joint distribution alignment and the Hellinger distance in our RRL approach. Particularly, we design 2 RRL variants: (i) RRL-marginal with the marginal distribution alignment instead of the joint distribution alignment and (ii) RRL-L^2 with the L^2-distance instead of the Hellinger distance. We run the 2 variants on tasks from the Office-Home dataset and report the classification results in Table 8. We can observe from Table 8 that while RRL-marginal and RRL-L^2 improve the performance over the Baseline method, they can not perform as well as the complete RRL approach with both the joint distribution alignment and the Hellinger distance. This indicates that the 2 ingredients are important and indispensable to our approach.

7. Conclusion

In this paper, we consider the Hellinger distance for the MSDA problem, a metric that is aware of the structure of the space of probability distributions. We derive a target error bound and exploit the error bound to guide the design of our MSDA approach, Riemannian Representation Learning (RRL), which trains the network model (containing the representation function and the classifier) to minimize both the average empirical source error and the average empirical Hellinger distance. Particularly, we derive the average empirical Hellinger distance by constructing and solving unconstrained convex optimization problems. In our experiments, we find that the proposed RRL approach is statistically better than the comparison methods. In the future, we intend to extend the Riemannian representation learning idea in this work to tackle the problem of unsupervised representation learning.

Declaration of Competing Interest

Authors declare that they have no conflict of interest.

Data availability

Data will be made available on request.

Acknowledgments

We thank Professor Lei Wang from University of Wollongong for insightful comments and suggestions that helped to improve the paper. This work was supported in part by the National Natural Science Foundation of China under Grants 62106137 and 61902231, and in part by Shantou University under Grant NTF21035.

Appendix A. Proof of Theorem 1

Proof. We have

$$\mathbb{E}_{P^t(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}),y)] = \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}),y)] + \mathbb{E}_{P^t(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}),y)] - \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}),y)] \tag{A.1}$$

$$\leq \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] + \left|\mathbb{E}_{P^t(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] - \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)]\right| \tag{A.2}$$

$$= \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] + \frac{1}{n}\Big|\sum_{s=1}^{n}\Big(\mathbb{E}_{P^s(\phi(\boldsymbol{x}),y)}[\ell(g(\phi(\boldsymbol{x})),y)] - \mathbb{E}_{P^t(\phi(\boldsymbol{x}),y)}[\ell(g(\phi(\boldsymbol{x})),y)]\Big)\Big| \tag{A.3}$$

$$\leq \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] + \frac{1}{n}\sum_{s=1}^{n}\Big|\mathbb{E}_{P^s(\phi(\boldsymbol{x}),y)}[\ell(g(\phi(\boldsymbol{x})),y)] - \mathbb{E}_{P^t(\phi(\boldsymbol{x}),y)}[\ell(g(\phi(\boldsymbol{x})),y)]\Big| \tag{A.4}$$

$$\leq \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] + \frac{M}{n}\sum_{s=1}^{n}\int\Big|P^s(\phi(\boldsymbol{x}),y) - P^t(\phi(\boldsymbol{x}),y)\Big|d\phi(\boldsymbol{x})dy. \tag{A.5}$$

Eq. (A.4) is due to the property of integral and Eq. (A.5) uses the assumption that ℓ is upper bounded by M. Furthermore, we have

$$\int\Big|P^s(\phi(\boldsymbol{x}),y) - P^t(\phi(\boldsymbol{x}),y)\Big|d\phi(\boldsymbol{x})dy = \sqrt{\Big(\int\Big|P^s(\phi(\boldsymbol{x}),y) - P^t(\phi(\boldsymbol{x}),y)\Big|d\phi(\boldsymbol{x})dy\Big)^2} \tag{A.6}$$

$$\leq \Big[\int\Big(\sqrt{P^s(\phi(\boldsymbol{x}),y)} - \sqrt{P^t(\phi(\boldsymbol{x}),y)}\Big)^2 d\phi(\boldsymbol{x})dy\Big]^{\frac{1}{2}} \times \Big[\int\Big(\sqrt{P^s(\phi(\boldsymbol{x}),y)} + \sqrt{P^t(\phi(\boldsymbol{x}),y)}\Big)^2 d\phi(\boldsymbol{x})dy\Big]^{\frac{1}{2}} \tag{A.7}$$

$$= \sqrt{\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y))} \times \Big[\int\Big(\sqrt{P^s(\phi(\boldsymbol{x}),y)} + \sqrt{P^t(\phi(\boldsymbol{x}),y)}\Big)^2 d\phi(\boldsymbol{x})dy\Big]^{\frac{1}{2}} \tag{A.8}$$

$$= \sqrt{\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y))} \times \Big[1+1+2\int\sqrt{P^s(\phi(\boldsymbol{x}),y)P^t(\phi(\boldsymbol{x}),y)}d\phi(\boldsymbol{x})dy\Big]^{\frac{1}{2}} \tag{A.9}$$

$$\leq 2\sqrt{\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y))} \tag{A.10}$$

$$\leq \frac{4}{\epsilon}\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)). \tag{A.11}$$

Eq. (A.7) is a result of the Cauchy–Schwarz inequality. Eq. (A.10) holds since $\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)) \geq 0$. Eq. (A.11) makes use of the assumption that $\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)) \geq \epsilon$ ($\epsilon > 0$) and the fact that $\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y))$ is upper bounded by 2, leading to

$$\frac{2}{\epsilon}\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)) \geq 2 \geq \sqrt{\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y))}. \tag{A.12}$$

Combining Eqs. (A.5) and (A.11), we arrive at the result

$$\mathbb{E}_{P^t(\boldsymbol{x},y)}[\ell(h(\boldsymbol{x}),y)] \leq \frac{1}{n}\sum_{s=1}^{n}\mathbb{E}_{P^s(\boldsymbol{x},y)}[\ell(g(\phi(\boldsymbol{x})),y)] + \frac{4M}{\epsilon}\frac{1}{n}\sum_{s=1}^{n}\mathrm{H}(P^s(\phi(\boldsymbol{x}),y), P^t(\phi(\boldsymbol{x}),y)). \tag{A.13}$$

□

Appendix B. Proof of Proposition 1

Proof. For a differential convex function f, the inequality $f(u) \geq f(v) + f'(v)(u-v)$ holds for any u and v. Therefore, for $f(u) = 2(1-\sqrt{u})$ being a differential convex function, we have $2(1-\sqrt{u}) \geq [2(1-\sqrt{v}) - \frac{1}{\sqrt{v}}(u-v)]$. Consequently, $2(1-\sqrt{u}) = \max_v[2(1-\sqrt{v}) - \frac{1}{\sqrt{v}}(u-v)]$, where the maximal value is attained at $v = u$. □

Appendix C. Details of the statistical test

We describe the procedure of the Wilcoxon signed-ranks test [50]. The test compares the performance of two methods over N tasks. Specifically, in each task the classification accuracy is adopted as the performance measure of the methods. To run the test, we rank the differences in performance of two methods for each task out of N tasks. The differences are ranked according to their absolute values. The smallest absolute value gets the rank of 1, the second smallest gets the rank of 2, and so on. In case of equality, average ranks are assigned. The statistic of the Wilcoxon signed-ranks test is computed as

$$T(a,b) = \min\{R^+(a,b), R^-(a,b)\}, \tag{A.1}$$

where $R^+(a,b)$ is the sum of ranks for the tasks on which method b outperforms method a and $R^-(a,b)$ is the sum of ranks for the opposite. They are defined as follows:

$$R^+(a,b) = \sum_{\mathrm{diff}_i>0}\mathrm{rank}(\mathrm{diff}_i) + \frac{1}{2}\sum_{\mathrm{diff}_i=0}\mathrm{rank}(\mathrm{diff}_i), \tag{A.2}$$

$$R^-(a,b) = \sum_{\mathrm{diff}_i<0}\mathrm{rank}(\mathrm{diff}_i) + \frac{1}{2}\sum_{\mathrm{diff}_i=0}\mathrm{rank}(\mathrm{diff}_i), \tag{A.3}$$

where diff_i is the difference between the accuracy of two methods on the ith task out of N tasks, and $\mathrm{rank}(\mathrm{diff}_i)$ is the rank of $|\mathrm{diff}_i|$. We let b be our RRL approach, and let a be its comparison method. Based on formulas (C.1)–(C.3), we can compute the statistic $T(a,b)$.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

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

[3] G.Y. Park, S.W. Lee, Information-theoretic regularization for multi-source domain adaptation, in: IEEE International Conference on Computer Vision, 2021, pp. 9214–9223.

[4] Y. Zuo, H. Yao, C. Xu, Attention-based multi-source domain adaptation, IEEE Trans. Image Process. 30 (2021) 3793–3803.

[5] C.-X. Ren, Y.-H. Liu, X.-W. Zhang, K.-K. Huang, Multi-source unsupervised domain adaptation via pseudo target domain, IEEE Trans. Image Process. 31 (2022) 2122–2135.

[6] M. Xu, H. Wang, B. Ni, Graphical modeling for multi-source domain adaptation, IEEE Trans. Pattern Anal. Mach. Intell. (2022) 1–14.

[7] H. Zhao, S. Zhang, G. Wu, J.M.F. Moura, J.P. Costeira, G.J. Gordon, Adversarial multiple source domain adaptation, Advances in Neural Information Processing Systems, vol. 31, 2018.

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

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

[10] R. Xu, Z. Chen, W. Zuo, J. Yan, L. Lin, Deep cocktail network: multi-source unsupervised domain adaptation with category shift, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3964–3973.

[11] Y. Zhu, F. Zhuang, D. Wang, Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources, in: AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 5989–5996.

[12] J. Wen, R. Greiner, D. Schuurmans, Domain aggregation networks for multi-source domain adaptation, in: International Conference on Machine Learning, vol. 119, 2020, pp. 10214–10224.

[13] S. Zhao, G. Wang, S. Zhang, Y. Gu, Y. Li, Z. Song, P. Xu, R. Hu, H. Chai, K. Keutzer, Multi-source distilling domain adaptation, in: AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 12975–12983.
[14] Y.-H. Liu, C.-X. Ren, A two-way alignment approach for unsupervised multi–source domain adaptation, Pattern Recognit. 124 (2022) 108430.
[15] Y. Li, m. Murias, g. Dawson, D.E. Carlson, Extracting relationships by multi--domain matching, Advances in Neural Information Processing Systems, volume 31, 2018.
[16] K. Li, J. Lu, H. Zuo, G. Zhang, Multi-source contribution learning for domain adaptation, IEEE Trans. Neural Netw. Learn. Syst. (2021) 1–15.
[17] M. Baktashmotlagh, M.T. Harandi, B.C. Lovell, M. Salzmann, Domain adaptation on the statistical manifold, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2481–2488.
[18] M. Harandi, M. Salzmann, M. Baktashmotlagh, Beyond Gauss: image-set matching on the Riemannianmanifold of PDFs, in: IEEE International Conference on Computer Vision, 2015, pp. 4112–4120.
[19] M. Baktashmotlagh, M. Harandi, M. Salzmann, Distribution-matching embedding for visual domain adaptation, J. Mach. Learn. Res. 17 (108) (2016) 1–30.
[20] V.N. Vapnik, Statistical Learning Theory, Wiley-Interscience, 1998.
[21] W.M. Kouw, M. Loog, A review of domain adaptation without target labels, IEEE Trans. Pattern Anal. Mach. Intell. 43 (3) (2021) 766–785.
[22] M. Long, Y. Cao, J. Wang, M. Jordan, Learning transferable features with deep adaptation networks, in: International Conference on Machine Learning, 2015, pp. 97–105.
[23] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March, V. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (59) (2016) 1–35.
[24] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7167–7176.
[25] S. Chen, L. Han, X. Liu, Z. He, X. Yang, Subspace distribution adaptation frameworks for domain adaptation, IEEE Trans. Neural Netw. Learn. Syst. 31 (12) (2020) 5204–5218.
[26] S. Wang, L. Zhang, P. Wang, M. Wang, X. Zhang, Bp-triplet net for unsupervised domain adaptation: aBayesian perspective, Pattern Recognit. 133 (2023) 108993.
[27] S. Cicek, S. Soatto, Unsupervised domain adaptation via regularized conditional alignment, in: IEEE International Conference on Computer Vision, 2019, pp. 1416–1425.
[28] Y. Zhu, F. Zhuang, J. Wang, G. Ke, J. Chen, J. Bian, H. Xiong, Q. He, Deep subdomain adaptation network for image classification, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2021) 1713–1722.
[29] P. Ge, C.-X. Ren, X.-L. Xu, H. Yan, Unsupervised domain adaptation via deep conditional adaptation network, Pattern Recognit. 134 (2023) 109088.
[30] B. Bhushan Damodaran, B. Kellenberger, R. Flamary, D. Tuia, N. Courty, DeepJDOT: deep joint distribution optimal transport for unsupervised domain adaptation, in: European Conference on Computer Vision, 2018, pp. 447–463.
[31] S. Chen, M. Harandi, X. Jin, X. Yang, Domain adaptation by joint distribution invariant projections, IEEE Trans. Image Process. 29 (2020) 8264–8277.
[32] S. Chen, Z. Hong, M. Harandi, X. Yang, Domain neural adaptation, IEEE Trans. Neural Netw. Learn. Syst. (2022) 1–12.
[33] R. Li, X. Jia, J. He, S. Chen, Q. Hu, T-SVDNet: exploring high-order prototypical correlations for multi-source domain adaptation, in: IEEE International Conference on Computer Vision, 2021, pp. 9971–9980.
[34] H. Wang, M. Xu, B. Ni, W. Zhang, Learning to combine: knowledge aggregation for multi-source domain adaptation, in: European Conference on Computer Vision, 2020, pp. 727–744.
[35] L. Yang, Y. Balaji, S.-N. Lim, A. Shrivastava, Curriculum manager for source selection in multi-source domain adaptation, in: European Conference on Computer Vision, 2020, pp. 608–624.
[36] Z. Luo, X. Zhang, S. Lu, S. Yi, Domain consistency regularization for unsupervised multi-source domain adaptive classification, Pattern Recognit. 132 (2022) 108955.
[37] N. Venkat, J.N. Kundu, D. Singh, A. Revanur, et al., Your classifier can secretly suffice multi-source domain adaptation, Adv. Neural Inf. Process. Syst. 33 (2020) 4647–4659.
[38] L. Zhou, M. Ye, D. Zhang, C. Zhu, L. Ji, Prototype-based multisource domain adaptation, IEEE Trans. Neural Netw. Learn. Syst. (2021) 1–13.
[39] Z. Deng, K. Zhou, D. Li, J. He, Y.-Z. Song, T. Xiang, Dynamic instance domain adaptation, IEEE Trans. Image Process. 31 (2022) 4585–4597.
[40] S. Boyd, S.P. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[41] J. Nocedal, S.J. Wright, Numerical Optimization, Springer, 1999.
[42] M. Long, J. Wang, G. Ding, J. Sun, P.S. Yu, Transfer feature learning with joint distribution adaptation, in: IEEE International Conference on Computer Vision, 2013, pp. 2200–2207.
[43] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J.W. Vaughan, A theory of learning from different domains, Mach. Learn. 79 (1) (2010) 151–175.
[44] A.T. Nguyen, T. Tran, Y. Gal, P.H. Torr, A.G. Baydin, KL guided domain adaptation, in: International Conference on Learning Representations, 2022, pp. 1–12.
[45] L. Wasserman, All of Statistics: A Concise Course in Statistical Inference, vol. 26, Springer, 2004.
[46] L. Pardo, Statistical Inference Based on Divergence Measures, Chapman and Hall/CRC, 2018.
[47] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: European Conference on Computer Vision, 2010, pp. 213–226.
[48] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2066–2073.
[49] D. Acuna, G. Zhang, M.T. Law, S. Fidler, f-domain adversarial learning: theory and algorithms, in: International Conference on Machine Learning, vol. 139, 2021, pp. 66–75.
[50] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[51] L. van der Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.

Sentao Chen received the Ph.D. degree in Software Engineering from South China University of Technology, Guangzhou, China, in 2020. He is currently a Lecturer with the Department of Computer Science, Shantou University, Shantou, China. His research interests include statistical machine learning, domain adaptation, and domain generalization

Lin Zheng received the Ph.D. degree in Computer Science from Wuhan University, Wuhan, China, in 2017. From 2017 to 2018, he was a post-doctoral research fellow with the Department of Computer Science, Hong Kong Baptist University. He is currently a Lecturer with the Department of Computer Science, Shantou University. His research interests include deep learning and recommender systems.

Hanrui Wu is currently an Associate Professor with the Department of Computer Science, Jinan University, Guangzhou, China. Before that, he was a Postdoctoral Research Fellow with the Department of Mathematics, The University of Hong Kong, from 2020 to 2021. He received the B.S. and Ph.D. degrees in the School of Software Engineering from South China University of Technology, China, in 2013 and 2020, respectively. His research interests include transfer learning, zero-shot learning, hypergraph learning, recommendation systems, and brain-computer interactions.