Learning an Invariant Hilbert Space for Domain Adaptation

Samitha Herath[1,2], Mehrtash Harandi[1,2] and Fatih Porikli[1]
[1]Australian National University, [2]DATA61-CSIRO
Canberra, Australia
{samitha.herath, mehrtash.harandi}@data61.csiro.au, fatih.porikli@anu.edu.au

Abstract

This paper introduces a learning scheme to construct a Hilbert space (i.e., a vector space along its inner product) to address both unsupervised and semi-supervised domain adaptation problems. This is achieved by learning projections from each domain to a latent space along the Mahalanobis metric of the latent space to simultaneously minimizing a notion of domain variance while maximizing a measure of discriminatory power. In particular, we make use of the Riemannian optimization techniques to match statistical properties (e.g., first and second order statistics) between samples projected into the latent space from different domains. Upon availability of class labels, we further deem samples sharing the same label to form more compact clusters while pulling away samples coming from different classes.We extensively evaluate and contrast our proposal against state-of-the-art methods for the task of visual domain adaptation using both handcrafted and deep-net features. Our experiments show that even with a simple nearest neighbor classifier, the proposed method can outperform several state-of-the-art methods benefiting from more involved classification schemes.

1. Introduction

This paper presents a learning algorithm to address both *unsupervised* [21, 16, 49] and *semi-supervised* [27, 14, 29] domain adaptation problems. Our goal here is to learn a latent space in which domain disparities are minimized. We show such a space can be learned by first matching the statistical properties of the projected domains (*e.g.*, covariance matrices), and then adapting the Mahalanobis metric of the latent space to the labeled data, *i.e.*, minimizing the distances between pairs sharing the same class label while pulling away samples with different class labels. We develop a geometrical solution to jointly learn projections onto the latent space and the Mahalanobis metric there by making use of the concepts of Riemannian geometry.

Thanks to deep learning, we are witnessing rapid growth in classification accuracy of the imaging techniques if substantial amount of labeled data is provided [35, 48, 25, 26]. However, harnessing the attained knowledge into a new application with limited labeled data (or even without having labels) is far beyond clear [33, 37, 19, 8, 51]. To make things even more complicated, due to the inherit *bias* of datasets [50, 47], straightforward use of large amount of auxiliary data does not necessarily assure improved performances. For example, the ImageNet [43] data is hardly useful for an application designed to classify images captured by a mobile phone camera. Domain Adaptation (DA) is the science of reducing such undesired effects in transferring knowledge from the available auxiliary resources to a new problem.

The most natural solution to the problem of DA is by identifying the structure of a common space that minimizes a notion of domain mismatch. Once such a space is obtained, one can design a classifier in it, hoping that the classifier will perform equally well across the domains as the domain mismatched is minimized. Towards this end, several studies assume that either **1.** a subspace of the target[1] domain is the right space to perform DA and learn how the source domain should be mapped onto it [45, 29], or **2.** subspaces obtained from both source and target domains are equally important for classification, hence trying to either learn their evolution [22, 21] or similarity measure [46, 52, 14].

Objectively speaking, a common practice in many solutions including the aforementioned methods, is to simplify the learning problem by separating the two elements of it. That is, the algorithm starts by fixing a space (*e.g.*, source subspace in [16, 29]), and learns how to transfer the knowledge from domains accordingly. A curious mind may ask why should we resort to a predefined and fixed space in the first place.

In this paper, we propose a learning scheme that avoids such a separation. That is, we do not assume that a space or a transformation, apriori is known and fixed for DA. In

[1]In DA terminology target domain refers to the data directly related to the task. Source domain data is used as the auxiliary data for knowledge transferring.

essence, we propose to learn the structure of a Hilbert space (*i.e.*, its metric) along the transformations required to map the domains onto it jointly. This is achieved through the following contributions,

(i) We propose to learn the structure of a latent space, along its associated mappings from the source and target domains to address both problems of unsupervised and semi-supervised DA.

(ii) Towards this end, we propose to maximize a notion of discriminatory power in the latent space. At the same time, we seek the latent space to minimize a notion of statistical mismatch between the source and target domains (see Fig. 1 for a conceptual diagram).

(iii) Given the complexity of the resulting problem, we provide a rigorous mathematical modeling of the problem. In particular, we make use of the Riemannian geometry and optimization techniques on matrix manifolds to solve our learning problem[2].

(iv) We extensively evaluate and contrast our solution against several baseline and state-of-the-art methods in addressing both unsupervised and semi-supervised DA problems.

2. Proposed Method

In this work, we are interested in learning an **I**nvariant **L**atent **S**pace (ILS) to reduce the discrepancy between domains. We first define our notations. Bold capital letters denote matrices (*e.g.*, $\boldsymbol{X}$) and bold lower-case letters denote column vectors (*e.g.*, $\boldsymbol{x}$). $\mathbf{I}_n$ is the $n \times n$ identity matrix. $\mathcal{S}_{++}^n$ and $\mathrm{St}(n,p)$ denote the SPD and Stiefel manifolds, respectively, and will be formally defined later. We show the source and target domains by $\mathcal{X}_s \subset \mathbb{R}^s$ and $\mathcal{X}_t \subset \mathbb{R}^t$. The training samples from the source and target domains are shown by $\{\boldsymbol{x}_i^s, y_i^s\}_{i=1}^{n_s}$ and $\{\boldsymbol{x}_i^t\}_{i=1}^{n_t}$, respectively. For now, we assume only source data is labeled. Later, we discuss how the proposed learning framework can benefit form the labeled target data.

Our idea in learning an ILS is to determine the transformations $\mathbb{R}^{s\times p} \ni \boldsymbol{W}_s : \mathcal{X}_s \to \mathcal{H}$ and $\mathbb{R}^{t\times p} \ni \boldsymbol{W}_t : \mathcal{X}_t \to \mathcal{H}$ from the source and target domains to a latent p-dimensional space $\mathcal{H} \subset \mathbb{R}^p$. We furthermore want to equip the latent space with a Mahalanobis metric, $\boldsymbol{M} \in \mathcal{S}_{++}^p$, to reduce the discrepancy between projected source and target samples (see Fig. 1 for a conceptual diagram). To learn $\boldsymbol{W}_s$, $\boldsymbol{W}_t$ and $\boldsymbol{M}$ we propose to minimize a cost function in the form

$$\mathcal{L} = \mathcal{L}_d + \lambda \mathcal{L}_u \ . \tag{1}$$

In Eq. 1, $\mathcal{L}_d$ is a measure of dissimilarity between labeled samples. The term $\mathcal{L}_u$ quantifies a notion of statistical difference between the source and target samples in the latent space. As such, minimizing $\mathcal{L}$ leads to learning a latent space where not only the dissimilarity between labeled samples is reduced but also the domains are matched from a statistical point of view. The combination weight λ is envisaged to balance the two terms. The subscripts "d" and "u" in Eq. 1 stand for "Discriminative" and "Unsupervised". The reason behind such naming will become clear shortly. Below we detail out the form and properties of $\mathcal{L}_d$ and $\mathcal{L}_u$.

2.1. Discriminative Loss

The purpose of having $\mathcal{L}_d$ in Eq. 1 is to equip the latent space $\mathcal{H}$ with a metric to **1.** minimize dissimilarities between samples coming from the same class and **2.** maximizing the dissimilarities between samples from different classes.

Let $\boldsymbol{Z} = \{\boldsymbol{z}_j\}_{j=1}^n$ be the set of labeled samples in $\mathcal{H}$. In unsupervised domain adaptation $\boldsymbol{z}_j = \boldsymbol{W}_s^T \boldsymbol{x}_j^s$ and $n = n_s$. In the case of semi-supervised domain adaptation,

$$\boldsymbol{Z} = \left\{\boldsymbol{W}_s^T \boldsymbol{x}_j^s\right\}_{j=1}^{n_s} \bigcup \left\{\boldsymbol{W}_t^T \boldsymbol{x}_j^t\right\}_{j=1}^{n_{tl}},$$

where we assume n_{tl} labeled target samples are provided (out of available n_t samples). From the labeled samples in $\mathcal{H}$, we create N_p pairs in the form $(\boldsymbol{z}_{1,k}, \boldsymbol{z}_{2,k})$, $k = 1, 2, \cdots, N_p$ along their associated label $y_k \in \{-1, 1\}$. Here, $y_k = 1$ iff label of $\boldsymbol{z}_{1,k}$ is similar to that of $\boldsymbol{z}_{2,k}$ and -1 otherwise. That is the pair $(\boldsymbol{z}_{1,k}, \boldsymbol{z}_{2,k})$ is similar if $y_k = 1$ and dissimilar otherwise.

To learn the metric $\boldsymbol{M}$, we deem the distances between the similar pairs to be small while simultaneously making the distances between the dissimilar pairs large. In particular, we define $\mathcal{L}_d$ as,

$$\mathcal{L}_d = \frac{1}{N_p} \sum_{k=1}^{N_p} \ell_\beta\big(\boldsymbol{M}, y_k, \boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k}, 1\big) + r(\boldsymbol{M}), \tag{2}$$

with

$$\ell_\beta\big(\boldsymbol{M}, y, \boldsymbol{x}, u\big) = \frac{1}{\beta} \log\Big(1 + \exp\big(\beta y(\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x} - u)\big)\Big). \tag{3}$$

Here, ℓ_β is the generalized logistic function tailored with large margin structure (see Fig. 2) having a margin of u[3]. First note that the quadratic term in Eq. 3 (*i.e.*, $\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x}$) measures the Mahalanobis distance between $\boldsymbol{z}_{1,k}$ and $\boldsymbol{z}_{2,k}$ if used according to Eq.2. Also note that $\ell_\beta\big(\cdot, \cdot, \boldsymbol{x}, \cdot\big) = \ell_\beta\big(\cdot, \cdot, -\boldsymbol{x}, \cdot\big)$, hence how samples are order in the pairs is not important.

To better understand the behavior of the function ℓ_β, assume the function is fed with a similar pair, *i.e.* $y_k = 1$. For

[2]Our implementation is available on https://sherath@bitbucket.org/sherath/ils.git.

[3]For now we keep the margin at $u = 1$ and later will use this to explain the soft-margin extension.

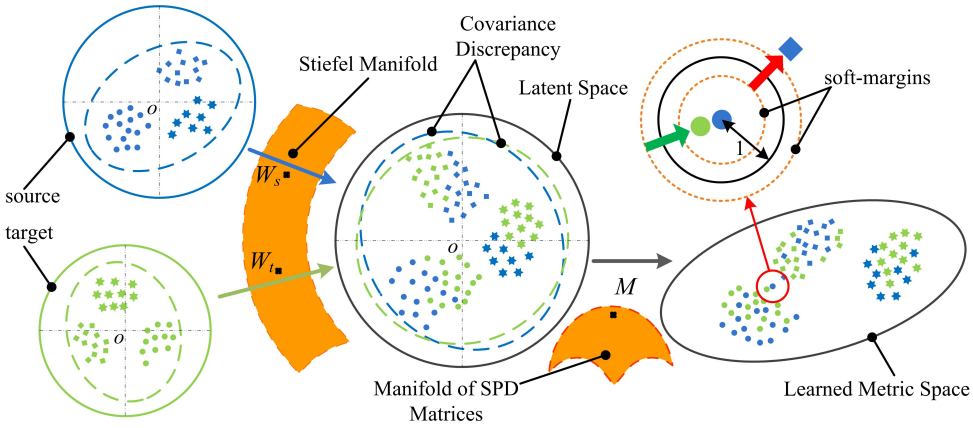

Figure 1. **A conceptual diagram of our proposal.** The marker shape represents the instance labels and color represents their original domains. Both source and target domains are mapped to a latent space using the transformations $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$. The metric, $\boldsymbol{M}$ defined in the latent space is learned to maximize the discriminative power of samples in it. Indicated by dashed ellipsoids are the domain distributions. The statistical loss of our cost function aims to reduce such discrepancies within the latent space. Our learning scheme identifies the transformations $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$ and the metric $\boldsymbol{M}$ jointly. This figure is best viewed in color.

the sake of discussion, also assume $\beta = 1$. In this case, ℓ_β is decreased if the distance between $\boldsymbol{z}_{1,k}$ and $\boldsymbol{z}_{2,k}$ is reduced. For a dissimilar pair (*i.e.*, $y_k = -1$), the opposite should happen to have a smaller objective. That is, the Mahalanobis distance between the samples of a pair should be increased.

The function $\ell_\beta\big(\cdot,\cdot,\boldsymbol{x},\cdot\big)$ can be understood as a smooth and differentiable form of the hinge-loss function. In fact, $\ell_\beta\big(\cdot,\cdot,\boldsymbol{x},\cdot\big)$ asymptotically reaches the hinge-loss function if $\beta \to \infty$. The smooth behavior of $\ell_\beta\big(\cdot,\cdot,\boldsymbol{x},\cdot\big)$ is not only welcomed in the optimization scheme but also avoids samples in the latent space to collapse into a single point.

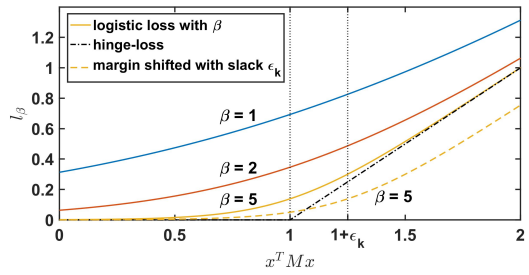

Figure 2. The behavior of the proposed ℓ_β(3) with $u = 1$ for various values of parameter β. Here, the horizontal axis is the value of the Mahalanobis distance and the function is plotted for $y = +1$. When $\beta \to \infty$, the function approaches the hinge-loss. An example of the soft-margin case (see Eq. 6), is also plotted for $\beta = 5$ curve. The figure is best seen in color.

Along the general practice in metric learning, we regularize the metric $\boldsymbol{M}$ by $r(\boldsymbol{M})$. The divergences derived from the $\log\det(\cdot)$ function are familiar faces for regularizing Mahalanobis metrics in the litrature [13, 45].

Among possible choices, we make use of the Stein divergence [11] in this work. Hence,

$$r(\boldsymbol{M}) = \frac{1}{p}\delta_s(\boldsymbol{M}, \boldsymbol{I_p}). \tag{4}$$

Where,

$$\delta_s(\boldsymbol{P},\boldsymbol{Q}) = \log\det\Big(\frac{\boldsymbol{P}+\boldsymbol{Q}}{2}\Big) - \frac{1}{2}\log\det\big(\boldsymbol{PQ}\big), \tag{5}$$

for $\boldsymbol{P},\boldsymbol{Q} \in \mathcal{S}_{++}$. Our motivation in using the Stein divergence stems from its unique properties. Among others, Stein divergence is symmetric, invariant to affine transformation and closely related to geodesic distances on the SPD manifold [11, 24, 9].

Soft Margin Extension

For large values of β, the cost in Eq. 2 seeks the distances of similar pairs to be less than 1 while simultaneously it deems the distances of dissimilar pairs to exceed 1. This hard-margin in the design of $\ell_\beta\big(\cdot,\cdot,\boldsymbol{x},\cdot\big)$ is not desirable. For example, with a large number of pairs, it is often the case to have outliers. Forcing outliers to fit into the hard margins can result in overfitting. As such, we propose a soft-margin extension of Eq. 3. The soft-margins are implemented by associating a non-negative slack variable ϵ_k to a pair according to

$$\mathcal{L}_d = \frac{1}{N_p}\sum_{k=1}^{N_p}\ell_\beta\big(\boldsymbol{M}, y_k, \boldsymbol{z}_{1,k} - \boldsymbol{z}_{2,k}, 1 + y_k\epsilon_k\big) + r(\boldsymbol{M}) + \frac{1}{N_p}\sqrt{\sum \epsilon_k^2}, \tag{6}$$

where a regularizer on the slack variables is also envisaged.

2.2. Matching Statistical Properties

A form of incompatibility between domains is due to their statistical discrepancies. Matching the first order

statistics of two domains for the purpose of adaptation is studied in [40, 2, 29][4]. In our framework, matching domain averages can be achieved readily. In particular, let $\bar{\boldsymbol{x}}_i^s = \boldsymbol{x}_i^s - \boldsymbol{m}_s$ and $\bar{\boldsymbol{x}}_j^t = \boldsymbol{x}_j^t - \boldsymbol{m}_t$ be the centered source and target samples with $\boldsymbol{m}_s$ and $\boldsymbol{m}_t$ being the mean of corresponding domains. It follows easily that the domain means in the latent space are zero[5] and hence matching is achieved.

To go beyond first order statistics, we propose to match the second order statistics (*i.e.*, covariance matrices) as well. The covariance of a domain reflects the relationships between its features. Hence, matching covariances of source and target domains in effect improves the cross feature relationships. We capture the mismatch between source and target covariances in the latent space using the $\mathcal{L}_u$ loss in Eq. 1. Given the fact that covariance matrices are points on the SPD manifold, we make use of the Stein divergence to measure their differences. This leads us to define $\mathcal{L}_u$ as

$$\mathcal{L}_u = \frac{1}{p}\delta_s(\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s, \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t), \tag{7}$$

with $\boldsymbol{\Sigma}_s \in \mathcal{S}_{++}^s$ and $\boldsymbol{\Sigma}_t \in \mathcal{S}_{++}^t$ being the covariance matrices of the source and target domains, respectively. We emphasize that matching the statistical properties as discussed above is an unsupervised technique, enabling us to address unsupervised DA.

2.3. Classification Protocol

Upon learning $\boldsymbol{W}_s$, $\boldsymbol{W}_t$, $\boldsymbol{M}$, training samples from the source and target (if available) domains are mapped to the latent space using $\boldsymbol{W}_s\boldsymbol{M}^{\frac{1}{2}}$ and $\boldsymbol{W}_t\boldsymbol{M}^{\frac{1}{2}}$, respectively. For a query from the target domain $\boldsymbol{x}_q^t$, $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{W}_t^T\boldsymbol{x}_q^t$ is its latent space representation which is subsequently classified by a nearest neighbor classifier.

3. Optimization

The objective of our algorithm is to learn the transformation parameters ($\boldsymbol{W}_s$ and $\boldsymbol{W}_t$), the metric $\boldsymbol{M}$ and slack variables $\epsilon_1, \epsilon_2, ...\epsilon_{N_p}$ (see Eq. 6 and Eq. 7). Inline with the general practice of dimensionality reduction, we propose to have orthogonality constraints on $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$. That is $\boldsymbol{W}_s^T\boldsymbol{W}_s = \boldsymbol{W}_t^T\boldsymbol{W}_t = \mathbf{I}_p$. We include an experiment elaborating how orthogonality constraint improves the discriminatory power of the proposed framework in the supplementary material.

[4] The use of Maximum Mean Discrepancy (MMD) [5] for domain adaptation is a well-practiced idea in the literature (see for example [40, 2, 29]). Empirically, determining MMD boils down to computing the distance between domain averages when domain samples are lifted to a reproducing kernel Hilbert space. Some studies claim matching the first order statistics is a weaker form of domain adaptation through MMD. We do not support this claim and hence do not see our solution as a domain adaptation method by minimizing the MMD.

[5] We note that $\sum \boldsymbol{W}_s^T\bar{\boldsymbol{x}}_i^s = \boldsymbol{W}_s^T\sum\bar{\boldsymbol{x}}_i^s = \mathbf{0}$. This holds for the target domain as well.

The problem depicted in Eq. 1 is indeed a non-convex and constrained optimization problem. One may resort to the method of Projected Gradient Descent (PGD) [7] to minimize Eq. 1. In PGD, optimization is proceed by projecting the gradient-descent updates onto the set of constraints. For example, in our case, we can first update $\boldsymbol{W}_s$ by ignoring the orthogonality constraint on $\boldsymbol{W}_s$ and then project the result onto the set of orthogonal matrices using eigen-decomposition. As such, optimization can be performed by alternatingly updating $\boldsymbol{W}_s$, $\boldsymbol{W}_t$, the metric $\boldsymbol{M}$ and slack variables using PGD.

In PGD, to perform the projection, the set of constraints needs to be closed though in practice one can resort to open sets. For example, the set of SPD matrices is open though one can project a symmetric matrix onto this set using eigen-decomposition.

Empirically, PGD showed an erratic and numerically unstable behavior in addressing our problem. This can be attributed to the non-linear nature of Eq. 1, existence of open-set constraints in the problem or perhaps the combination of both. To alleviate the aforementioned difficulty, we propose a more principle approach to minimize Eq. 1 by making use of the Riemannian optimization technique. We take a short detour and briefly describe the Riemannian optimization methods below.

Optimization on Riemannian manifolds.

Consider a non-convex constrained problem in the form

$$\begin{aligned} &\text{minimize } f(\boldsymbol{x}) \\ &\text{s.t. } \quad \boldsymbol{x} \in \mathcal{M}\,, \end{aligned} \tag{8}$$

where $\mathcal{M}$ is a Riemannian manifold, *i.e.*, informally, a smooth surface that locally resembles a Euclidean space. Optimization techniques on Riemannian manifolds (*e.g.*, Conjugate Gradient) start with an initial solution $\boldsymbol{x}^{(0)} \in \mathcal{M}$, and iteratively improve the solution by following the geodesic identified by the gradient. For example, in the case of Riemannian Gradient Descent Method (RGDM), the updating rule reads

$$\boldsymbol{x}^{(t+1)} = \tau_{\boldsymbol{x}^{(t)}}\big(-\alpha\ \text{grad}\ f(\boldsymbol{x}^{(t)})\big)\,, \tag{9}$$

with $\alpha > 0$ being the algorithm's step size. Here, $\tau_{\boldsymbol{x}}(\cdot) : T_{\boldsymbol{x}}\mathcal{M} \rightarrow \mathcal{M}$, is called the retraction[6] and moves the solution along the descent direction while assuring that the new solution is on the manifold $\mathcal{M}$, *i.e.*, it is within the constraint set. $T_{\boldsymbol{x}}\mathcal{M}$ is the tangent space of $\mathcal{M}$ at $\boldsymbol{x}$ and can be thought of as a vector space with its vectors being the gradients of all functions defined on $\mathcal{M}$.

[6] Strictly speaking and in contrast with the exponential map, a retraction only guarantees to pull a tangent vector on the geodesic locally, *i.e.*, close to the origin of the tangent space. Retractions, however, are typically easier to compute than the exponential map and have proven effective in Riemannian optimization [1].

Due to space limitation, we defer more details on Riemannian optimization techniques to the supplementary. As for now, it suffices to say that to perform optimization on the Riemannian manifolds, the form of Riemannian gradient, retraction and the gradient of the objective with respect to its parameters (shown by ∇) are required. The constraints in Eq.1 are orthogonality (transformations $\boldsymbol{W}_s$ and $\boldsymbol{W}_t$) and p.d. for metric $\boldsymbol{M}$. The geometry of these constraints are captured by the Stiefel [30, 23] and SPD [24, 10] manifolds, formally defined as

Definition 1 (The Stiefel Manifold) *The set of $(n \times p)$-dimensional matrices, $p \leq n$, with orthonormal columns endowed with the Frobenius inner product*[7] *forms a compact Riemannian manifold called the Stiefel manifold* $\mathrm{St}(p, n)$ *[1].*

$$\mathrm{St}(p, n) \triangleq \{\boldsymbol{W} \in \mathbb{R}^{n \times p} : \boldsymbol{W}^T \boldsymbol{W} = \mathbf{I}_p\} \,. \tag{10}$$

Definition 2 (The SPD Manifold) *The set of $(p \times p)$ dimensional real, SPD matrices endowed with the Affine Invariant Riemannian Metric (AIRM) [42] forms the SPD manifold $\mathcal{S}_{++}^p$.*

$$\mathcal{S}_{++}^p \triangleq \{\boldsymbol{M} \in \mathbb{R}^{p \times p} \colon \boldsymbol{v}^T \boldsymbol{M} \boldsymbol{v} > 0, \forall \boldsymbol{v} \in \mathbb{R}^p - \{\mathbf{0}_p\}\}. \tag{11}$$

Updating $\boldsymbol{W}_s$, $\boldsymbol{W}_t$ and $\boldsymbol{M}$ and slacks can be done alternatively using Riemannian optimization. As mentioned above, the ingredients for doing so are **1.** the Riemannian tools for the Stiefel and SPD manifolds along **2.** the form of gradients of the objective with respect to its parameters. To do complete justice, in Table. 1 we provide the Riemmanian metric, form of Riemannian gradient and retraction for the Stiefel and SPD manifolds. In Table. 2, the gradient of Eq. 1 with respect to $\boldsymbol{W}_s$, $\boldsymbol{W}_t$ and $\boldsymbol{M}$ and slacks is provided. The detail of derivations can be found in the supplementary material. A tiny note about the slacks worth mentioning. To preserve the non-negativity constraint on ϵ_k, we define $\epsilon_k = e^{v_k}$ and optimize on v_k instead. This in turn makes optimization for the slacks unconstrained.

Remark 1 *From a geometrical point of view, we can make use of the product topology of the parameter space to avoid alternative optimization. More specifically, the set*

$$\mathcal{M}_{prod.} = \mathrm{St}(p, s) \times \mathrm{St}(p, t) \times \mathcal{S}_{++}^p \times \mathbb{R}^{N_p}, \tag{12}$$

can be given the structure of a Riemannian manifold using the concept of product topology [1].

Remark 2 *In Fig. 3, we compare the convergence behavior of PGD, alternating Riemannian optimization and optimization using the product geometry. While optimization on $\mathcal{M}_{prod.}$ convergences faster, the alternating method results in a lower loss. This behavior resembles the difference between the stochastic gradient descent compared to its batch counterpart.*

[7] Note that the literature is divided between this choice and another form of Riemannian metric. See [15] for details.

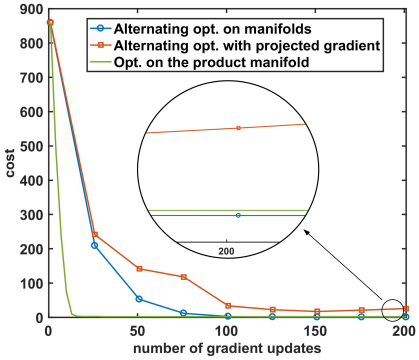

Figure 3. Optimizing Eq. 1 using PGD (red curve), Riemannian gradient descent using alternating approach (blue curve) and product topology (green curve). Optimization using the product topology converges faster but a lower cost can be attained using alternating Riemannian optimization.

Remark 3 *The complexity of the optimization depends on the number of labeled pairs. One can always resort to a stochastic solution [39, 44, 4] by sampling from the set of similar/dissimilar pairs if addressing a very large-scale problem. In our experiments, we did not face any difficulty optimizing with an i7 desktop machine with 32GB of memory.*

4. Related Work

The literature on domain adaptation spans a very broad range (see [41] for a recent survey). Our solution falls under the category of domain adaptation by subspace learning (DA-SL). As such, we confine our review only to methods under the umbrella of DA-SL.

One notable example of constructing a latent space is the work of Daumé III *et al*. [12]. In particular, the authors propose to use two fixed and predefined transformations to project source and target data to a common and higher-dimensional space. As a requirement, the method only accepts domains with the same dimensionality and hence cannot be directly used to adapt heterogeneous domains.

Goplan *et al*. observed that the geodesic connecting the source and target subspaces conveys useful information for DA and proposed the Sampling Geodesic Flow (SGF) method [22]. The Geodesic Flow Kernel (GFK) is an improvement over the SGF technique where instead of sampling a few points on the geodesic, the whole curve is used for domain adaptation [21]. In both methods, the domain subspaces are fixed and obtained by Principal Component Analysis (PCA) or Partial Least Square regression (PLS) [34]. In contrast to our solution, in SGF and GFK learning the domain subspaces is disjoint from the knowledge transfer algorithm. In our experiments, we will see

Table 1. Riemannian metric, gradient and retraction on $\mathrm{St}(p,n)$ and $\mathcal{S}_{++}^{p}$. Here, $\mathrm{uf}(\boldsymbol{A}) = \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1/2}$, which yields an orthogonal matrix, $\mathrm{sym}(\boldsymbol{A}) = \frac{1}{2}(\boldsymbol{A}+\boldsymbol{A}^T)$ and $\mathrm{expm}(\cdot)$ denotes the matrix exponential.

	$\mathrm{St}(p,n)$	$\mathcal{S}_{++}^{p}$
Matrix representation	$\boldsymbol{W} \in \mathbb{R}^{n\times p}$	$\boldsymbol{M} \in \mathbb{R}^{p\times p}$
Riemannian metric	$g_\nu(\xi,\varsigma) = \mathrm{Tr}(\xi^T\varsigma)$	$g_{\mathcal{S}}(\xi,\varsigma) = \mathrm{Tr}\left(\boldsymbol{M}^{-1}\xi\boldsymbol{M}^{-1}\varsigma\right)$
Riemannian gradient	$\nabla_{\boldsymbol{W}}(f) - \boldsymbol{W}\mathrm{sym}\Big(\boldsymbol{W}^T\nabla_{\boldsymbol{W}}(f)\Big)$	$\boldsymbol{M}\mathrm{sym}\Big(\nabla_{\boldsymbol{M}}(f)\Big)\boldsymbol{M}$
Retraction	$\mathrm{uf}(\boldsymbol{W}+\xi)$	$\boldsymbol{M}^{\frac{1}{2}}\,\mathrm{expm}(\boldsymbol{M}^{-\frac{1}{2}}\xi\boldsymbol{M}^{-\frac{1}{2}})\boldsymbol{M}^{\frac{1}{2}}$

Table 2. Gradients of soft-margin ℓ_β and $\mathcal{L}_u$ w.r.t. the model parameters and slack variables. Without less of generality we only consider a labeled similar ($y_k = +1$) pair $\boldsymbol{x}_i^s$ and $\boldsymbol{x}_j^t$. Here, $r = \exp\big(\beta\big((\boldsymbol{W}_s^T\boldsymbol{x}_i^s - \boldsymbol{W}_t^T\boldsymbol{x}_i^t)^T\boldsymbol{M}(\boldsymbol{W}_s^T\boldsymbol{x}_i^s - \boldsymbol{W}_t^T\boldsymbol{x}_i^t) - 1 - e^{v_k}\big)\big)$.

$\nabla_{\boldsymbol{W}_s}\ell_\beta$	$\frac{2}{N_p}(1+r^{-1})^{-1}\boldsymbol{x}_i^s(\boldsymbol{x}_i^{sT}\boldsymbol{W}_s - \boldsymbol{x}_j^{tT}\boldsymbol{W}_t)\boldsymbol{M}$
$\nabla_{\boldsymbol{M}}\ell_\beta$	$\frac{1}{N_p}(1+r^{-1})^{-1}(\boldsymbol{W}_s^T\boldsymbol{x}_i^s - \boldsymbol{W}_t^T\boldsymbol{x}_j^t)(\boldsymbol{x}_i^{sT}\boldsymbol{W}_s - \boldsymbol{x}_j^{tT}\boldsymbol{W}_t)$
$\nabla_{v_k}\ell_\beta$	$\frac{1}{N_p}e^{v_k}(1+r^{-1})^{-1}$
$\nabla_{\boldsymbol{W}_s}\mathcal{L}_u$	$\frac{1}{p}\boldsymbol{\Sigma}_s\boldsymbol{W}_s\big(2\big\{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s + \boldsymbol{W}_t^T\boldsymbol{\Sigma}_t\boldsymbol{W}_t\big\}^{-1} - \big\{\boldsymbol{W}_s^T\boldsymbol{\Sigma}_s\boldsymbol{W}_s\big\}^{-1}\big)$

that the subspaces determined by our method can even boost the performance of GFK, showing the importance of joint learning of domain subspaces and knowledge transferring. In [38] dictionary learning is used for interpolating the intermediate subspaces.

Domain adaptation by fixing the subspace/representation of one of the domains is a popular theme in many recent works, as it simplifies the learning scheme. Examples are the max-margin adaptation [27, 14], the metric/similarity learning of [45] and its kernel extension [36], the landmark approach of [29], the alignment technique of [16, 17], correlation matching of [49] and methods that use maximum mean discrepancy (MMD) [5] for DA [40, 2].

In contrast to the above methods, some studies opt to learn the domain representation along the knowledge transfer method jointly. Two representative works are the HeMap [46] and manifold alignment [52]. The HeMap learns two projections to minimize the instance discrepancies [46]. The problem is however formulated such that equal number of source and target instances is required to perform the training. The manifold alignment algorithm of [52] attempts to preserve the label structure in the latent space. However, it is essential for the algorithm to have access to labeled data in both source and target domains.

Our solution learns all transformations to the latent space. We do not resort at subspace representations learned disjointly to the DA framework. With this use of the latent space, our algorithm is not limited for applications where source and target data have similar dimensions or structure.

5. Experimental Evaluations

We run extensive experiments on both semi-supervised and unsupervised settings, spanning from the handcrafted features (SURF) to the current state-of-the-art deep-net features (VGG-Net). For comparisons, we use the implementations made available by the original authors. Our method is denoted by **ILS**.

5.1. Implementation Details

Since the number of dissimilar pairs is naturally larger than the number of similar pairs, we randomly sample from the different pairs to keep the sizes of these two sets equal. We initialize the projection matrices $\boldsymbol{W_s}$, $\boldsymbol{W_t}$ with PCA, following the transductive protocol [21, 16, 27, 29]. For the semi-supervised setting, we initialize $\boldsymbol{M}$ with the Mahalanobis metric learned on the similar pair covariances [31], and for the unsupervised setting, we initialize it with the identity matrix. For all our experiments we have $\lambda = 1$. We include an experiment showing our solution's robustness to λ in the supplementary material. We use the toolbox provided by [6] for our implementations.

Remark 4 *To have a simple way of determining β in Eq. 3, we propose a heuristic which is shown to be effective in our experiments. To this end, we propose to set β to the reciprocal of the standard deviation of the similar pair distances.*

5.2. Semi-supervised Setting

In our semi-supervised experiments, we follow the standard setup on the Office+Caltech10 dataset with the train/test splits provided by [28]. The Office+Caltech10 dataset contains images collected from 4 different sources and 10 object classes. The corresponding domains are **A**mazon, **W**ebcam, **D**SLR, and **C**altech. We use a subspace of dimension 20 for DA-SL algorithms. We employ SURF [3] for the handcrafted feature experiments. We extract VGG-Net features with the network model of [48] for the deep-net feature experiments[8]. We compare our performance with the following benchmarks:

1-NN-t and SVM-t : Basic Nearest Neighbor (1-NN) and linear SVM classifiers trained only on the target domain.

HFA [14] : This method employs latent space learning based on the max-margin framework. As in its original implementation, we use the RBF kernel SVM for its evaluation.

MMDT [27] : This method jointly learns a transformation between the source and target domains along a linear SVM for classification.

CDLS [29] : This is the cross-domain landmark search algorithm. We use the parameter setting ($\delta = 0.5$ in the notation of [29]) recommended by the authors.

[8] The same SURF and VGG-FC6 features are used for the unsupervised experiments as well.

Table 3 and Table 4 report the performances using the handcrafted SURF and the VGG-FC6 layer features, respectively. For the SURF features our solution achieves the best performance in 7 out 12 cases, and for the VGG-FC6 features, our solution tops in 9 sets. We notice the **1-NN-t** baseline performs the worst for both SURF and the VGG-FC6 features. Hence, it is clear that the used features do not favor the nearest neighbor classifier. We observe that **C**altech and **A**mazon domains contain the largest number of test instances. Although the performances of all tested methods decrease on these domains, particularly on **C**altech, our method achieves the top rank in almost all domain transformations.

5.3. Unsupervised Setting

In the unsupervised domain adaptation problem, only labeled data from the source domain is available [16, 21]. We perform two sets of experiments for this setting. (1) We evaluate the object recognition performance on the Office+Caltech10 dataset. Similar to the semi-supervised settings, we use the SURF and VGG-FC6 features. Our results demonstrate that the learned transformations by our method are superior domain representations. (2) We analyze our performance when the domain discrepancy is gradually increased. This experiment is performed on the PIE-Face dataset. We compare our method with the following benchmarks:

1-NN-s and SVM-s : Basic 1-NN and linear SVM classifiers trained only on the source domain.
GFK-PLS [21] : The geodesic flow kernel algorithm where partial least squares (PLS) implementation is used to initialize the source subspace. Results are evaluated on kernel-NNs.
SA [16] : This is the subspace alignment algorithm. Results are evaluated using 1-NN.
CORAL [49] : The correlation alignment algorithm that uses a linear SVM on the similarity matrix formed by correlation matching.

5.3.1 Office+Caltech10 (Unsupervised)

We follow the original protocol provided by [21] on Office+Caltech10 dataset. Note that several baselines, determine the best dimensionality per domain to achieve their maximum accuracies on SURF features. We observed that a dimensionality in the range [20,120] provides consistent results for our solution using SURF features. For VGG features we empirically found the dimensionality of 20 suits best for the compared DA-SL algorithms.

Table. 5 and Table. 6 present the unsupervised setting results using the SURF and VGG-FC6 features. For both feature types, our solution yields the best performance in 8 domain transformations out of 12.

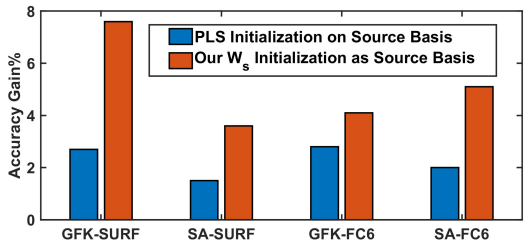

Figure 4. The accuracy gain on Office-Caltech dataset for GFK [21] and SA [16] when their initial PCA subspaces are replaced with PLS and our W_s transformation matrices.

Learned Transformations as Subspace Representations: We consider both GFK [21] and SA [16] as DA-SL algorithms. Both these methods make use of PCA subspaces to adapt the domains. Nevertheless, there is no strong reason to assume the PCA subspaces favorably capture the domain structure for transfer learning. Gong *et al.*, [21] show that their performance improves when employing PLS[9] to define the source subspace. However, this subspace learning is disjoint to their domain adaptation technique. We notice that, a more suitable initialization would be to use a subspace representation learned along with a domain adaptation framework. We empirically show this by using our learned source transformation matrix $\boldsymbol{W}_s$ as the source subspace initialization for [21] and [16].

Figure 4 compares the accuracy gains over PCA spaces by using PLS and our $\boldsymbol{W}_s$ initialization. It is clear that the highest classification accuracy gain is obtained by our $\boldsymbol{W}_s$ initialization. This proves that $\boldsymbol{W}_s$ is capable to learn a more favorable subspace representation for DA.

5.3.2 PIE-Multiview Faces

The PIE Multiview dataset includes face images of 67 individuals captured from different views, illumination conditions, and expressions. In this experiment, we use the views $C27$ (looking forward) as the source domain and $C09$ (looking down), and the views $C05$, $C37$, $C02$, $C25$ (looking towards left in an increasing angle, see Fig. 5) as target domains. We expect the face inclination angle to reflect the complexity of transfer learning. We normalize the images to $32\times$ 32 pixels and use the vectorized gray-scale images as features. Empirically, we observed that the GFK [21] and SA [16] reach better performances if the features are normalized to have unit ℓ_2 norm. We therefore use ℓ_2 normalized features in our evaluations. The dimensionality of the subspaces for all the subspace based methods (*i.e.*, [21, 16]) including ours is 100.

Table. 7 lists the classification accuracies with increasing angle of inclination. Our solution attains best scores for 4 views and the second best for the $C09$. With the increasing

[9]Despite using labeled data, this method falls under the unsupervised setting since it does not use the labeled target data.

Table 3. Semi-supervised domain adaptation results using SURF features on Office+Caltech10 [21] dataset with the evaluation setup of [27]. The best score (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-t	34.5	33.6	19.7	29.5	35.9	18.9	27.1	33.4	18.6	29.2	33.5	34.1
SVM-t	63.7	57.2	32.2	46.0	56.5	29.7	45.3	62.1	32.0	45.1	60.2	56.3
HFA [14]	57.4	55.1	31.0	**56.5**	56.5	29.0	42.9	60.5	30.9	43.8	58.1	55.6
MMDT [27]	64.6	56.7	36.4	47.7	67.0	32.2	46.9	74.1	34.1	49.4	63.8	56.5
CDLS [29]	**68.7**	**60.4**	35.3	51.8	60.7	33.5	50.7	68.5	34.9	50.9	**66.3**	**59.8**
ILS (1-NN)	59.7	49.8	**43.6**	54.3	**70.8**	**38.6**	**55.0**	**80.1**	**41.0**	**55.1**	62.9	56.2

Table 4. Semi-supervised domain adaptation results using VGG-FC6 features on Office+Caltech10 [21] dataset with the evaluation setup of [27]. The best (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-t	81.0	79.1	67.8	76.1	77.9	65.2	77.1	81.7	65.6	78.3	80.2	77.7
SVM-t	89.1	**88.2**	77.3	86.5	87.7	76.3	87.3	88.3	76.3	87.5	87.8	84.9
HFA [14]	87.9	87.1	75.5	85.1	87.3	74.4	85.9	86.9	74.8	86.2	86.0	**87.0**
MMDT [27]	82.5	77.1	78.7	84.7	85.1	73.6	83.6	86.1	71.8	85.9	82.8	77.9
CDLS [29]	**91.2**	86.9	78.1	87.4	88.5	78.2	88.1	90.7	77.9	88.0	89.7	86.3
ILS (1-NN)	90.7	87.7	**83.3**	**88.8**	**94.5**	**82.8**	**88.7**	**95.5**	**81.4**	**89.7**	**91.4**	86.9

Table 5. Unsupervised domain adaptation results using SURF features on Office+Caltech10 [21] dataset with the evaluation setup of [21].The best (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-s	23.1	22.3	20.0	14.7	31.3	12.0	23.0	51.7	19.9	21.0	19.0	23.6
SVM-s	25.6	33.4	35.9	30.4	67.7	23.4	34.6	70.2	31.2	43.8	30.5	40.3
GFK-PLS [21]	35.7	35.1	37.9	35.5	71.2	29.3	36.2	79.1	32.7	40.4	35.8	41.1
SA [16]	38.6	37.6	35.3	37.4	80.3	32.3	38.0	83.6	32.4	39.0	36.8	39.6
CORAL [49]	38.7	38.3	**40.3**	37.8	**84.9**	**34.6**	38.1	**85.9**	34.2	47.2	39.2	40.7
ILS (1-NN)	**40.6**	**41.0**	37.1	**38.6**	72.4	32.6	**38.9**	79.1	**36.9**	**48.6**	**42.0**	**44.1**

Table 6. Unsupervised domain adaptation results using VGG-FC6 features on Office+Caltech10 [21] dataset with the evaluation setup of [21].The best (in bold blue), the second best (in blue).

	A→W	A→D	A→C	W→A	W→D	W→C	D→A	D→W	D→C	C→A	C→W	C→D
1-NN-s	60.9	52.3	70.1	62.4	83.9	57.5	57.0	86.7	48.0	81.9	65.9	55.6
SVM-s	63.1	51.7	74.2	69.8	89.4	64.7	58.7	91.8	55.5	86.7	74.8	61.5
GFK-PLS [21]	74.1	63.5	77.7	77.9	**92.9**	71.3	69.9	92.4	64.0	86.2	76.5	66.5
SA [16]	76.0	64.9	77.1	76.6	90.4	70.7	69.0	90.5	62.3	83.9	76.0	66.2
CORAL [49]	74.8	67.1	**79.0**	81.2	92.6	75.2	75.8	**94.6**	64.7	**89.4**	77.6	67.6
ILS (1-NN)	**82.4**	**72.5**	78.9	**85.9**	87.4	**77.0**	**79.2**	94.2	**66.5**	87.6	**84.4**	**73.0**

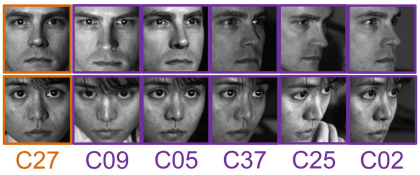

Figure 5. Two instances of the PIE-Multiview face data. Here, the view from $C27$ is used as the source domain. Remaining views are considered to be the target for each transformation.

Table 7. PIE-Multiview results. The variation of performance w.r.t. face orientations when frontal face images are considered as the source domain.

camera pose→	C09	C05	C37	C25	C02
1-NN-s	92.5	55.7	28.5	14.8	11.0
SVM-s	87.8	65.0	35.8	15.7	16.7
GFK-PLS [21]	92.5	74.0	32.1	14.1	12.3
SA [16]	**97.9**	85.9	47.9	16.6	13.9
CORAL [49]	91.4	74.8	35.3	13.4	13.2
ILS (1-NN)	96.6	**88.3**	**72.9**	**28.4**	**34.8**

camera angle, the feature structure changes up to a certain extent. In other words, the features become heterogeneous. However, our algorithm boosts the accuracies even under such challenging conditions.

Conclusion

In this paper, we proposed a solution for both semi-supervised and unsupervised Domain Adaptation (DA) problems. Our solution learns a latent space in which domain discrepancies are minimized. We showed that such a latent space can be obtained by **1.** minimizing a notion of discriminatory power over the available labeled data while simultaneously **2.** matching statistical properties across the domains. To determine the latent space, we modeled the learning problem as a minimization problem on Riemannian manifolds and solved it using optimization techniques on matrix manifolds.

Empirically, we showed that the proposed method outperformed state-of-the-art DA solutions in semi-supervised and unsupervised settings. With the proposed framework we see possibilities of extending our solution to large scale datasets with stochastic optimization techniques, multiple source DA and for domain generalization [20, 18]. In terms of algorithmic extensions we look forward to use dictionary learning [32] and higher order statistics matching.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 4, 5

[2] M. Baktashmotlagh, M. Harandi, and M. Salzmann. Distribution-matching embedding for visual domain adaptation. *Journal of Machine Learning Research*, 17(108):1–30, 2016. 4, 6

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006. 6

[4] S. Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013. 5

[5] K. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schoelkopf, and A. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22:e49–e57, 2006. 4, 6

[6] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15:1455–1459, 2014. 6

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. 4

[8] Q. Chen, J. Huang, R. Feris, L. M. Brown, J. Dong, and S. Yan. Deep domain adaptation for describing people based on fine-grained clothing attributes. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5315–5324, 2015. 1

[9] A. Cherian, V. Morellas, and N. Papanikolopoulos. Bayesian nonparametric clustering for positive definite matrices. *IEEE transactions on pattern analysis and machine intelligence*, 38(5):862–874, 2016. 3

[10] A. Cherian and S. Sra. Positive definite matrices: data representation and applications to computer vision. *Algorithmic Advances in Riemannian Geometry and Applications: For Machine Learning, Computer Vision, Statistics, and Optimization*, page 93, 2016. 5

[11] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos. Jensen-Bregman logdet divergence with application to efficient similarity search for covariance matrices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2161–2174, Sept 2013. 3

[12] H. Daumé III, A. Kumar, and A. Saha. Frustratingly easy semi-supervised domain adaptation. In *Workshop on Domain Adaptation for NLP*, 2010. 5

[13] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, Corvalis, Oregon, USA, 2007. 3

[14] L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 711–718, Edinburgh, Scotland, June 2012. Omnipress. 1, 6, 8

[15] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998. 5

[16] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 2960–2967, 2013. 1, 6, 7, 8

[17] B. Fernando, T. Tommasi, and T. Tuytelaars. Joint cross-domain classification and subspace learning for unsupervised adaptation. *Pattern Recognition Letters*, 65:60 – 66, 2015. 6

[18] C. Gan, T. Yang, and B. Gong. Learning attributes equals multi-source domain generalization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 87–97, 2016. 8

[19] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 1180–1189, 2015. 1

[20] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang. Scatter component analysis: A unified framework for domain adaptation and domain generalization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2016. 8

[21] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073, 2012. 1, 5, 6, 7, 8

[22] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 999–1006, 2011. 1, 5

[23] M. Harandi and B. Fernando. Generalized backpropagation, étude de cas: Orthogonality. *CoRR*, abs/1611.05927, 2016. 5

[24] M. Harandi, M. Salzmann, and R. Hartley. Dimensionality reduction on SPD manifolds: The emergence of geometry-aware methods. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2017. 3, 5

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[26] S. Herath, M. Harandi, and F. Porikli. Going deeper into action recognition: A survey. *Image and Vision Computing*, 60:4 – 21, 2017. Regularization Techniques for High-Dimensional Data Analysis. 1

[27] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *Int. Journal of Computer Vision*, 109(1):28–41, 2014. 1, 6, 8

[28] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell. Efficient learning of domain-invariant image representations. In *International Conference on Learning Representations*, 2013. 6

[29] Y.-H. Hubert Tsai, Y.-R. Yeh, and Y.-C. Frank Wang. Learning cross-domain landmarks for heterogeneous domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1, 4, 6, 8

[30] I. M. James. *The topology of Stiefel manifolds*, volume 24. Cambridge University Press, 1976. 5

[31] M. Koestinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence

constraints. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 6

[32] P. Koniusz and A. Cherian. Sparse coding for third-order super-symmetric tensor descriptors with application to texture recognition. *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 5395, 2016. 8

[33] P. Koniusz, Y. Tas, and F. Porikli. Domain adaptation by mixture of alignments of second-or higher-order scatter tensors. *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[34] A. Krishnan, L. J. Williams, A. R. McIntosh, and H. Abdi. Partial least squares (pls) methods for neuroimaging: a tutorial and review. *Neuroimage*, 56(2):455–475, 2011. 5

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105. Curran Associates, Inc., 2012. 1

[36] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1785–1792, June 2011. 6

[37] M. Long, J. Wang, Y. Cao, J. Sun, and P. S. Yu. Deep learning of transferable representation for scalable domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2027–2040, Aug 2016. 1

[38] J. Ni, Q. Qiu, and R. Chellappa. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 692–699, 2013. 6

[39] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 5

[40] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011. 4, 6

[41] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, May 2015. 5

[42] X. Pennec, P. Fillard, and N. Ayache. A Riemannian framework for tensor computing. *Int. Journal of Computer Vision*, 66(1):41–66, 2006. 5

[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 1

[44] C. D. Sa, C. Re, and K. Olukotun. Global convergence of stochastic gradient descent for some non-convex matrix problems. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 2332–2341, 2015. 5

[45] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. *Adapting Visual Category Models to New Domains*, pages 213–226. 2010. 1, 3, 6

[46] X. Shi, Q. Liu, W. Fan, S. Y. Philip, and R. Zhu. Transfer learning on heterogenous feature spaces via spectral transformation. In *2010 IEEE international conference on data mining*, pages 1049–1054, 2010. 1, 6

[47] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227 – 244, 2000. 1

[48] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 6

[49] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. 1, 6, 7, 8

[50] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1528, 2011. 1

[51] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 1

[52] C. Wang and S. Mahadevan. Heterogeneous domain adaptation using manifold alignment. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1541–1546, 2011. 1, 6