Flowing on Riemannian Manifold: Domain Adaptation by Shifting Covariance

Zhen Cui, *Student Member, IEEE,* Wen Li, *Student Member, IEEE,* Dong Xu, *Senior Member, IEEE,* Shiguang Shan, *Member, IEEE,* Xilin Chen, *Senior Member, IEEE,* and Xuelong Li, *Fellow, IEEE*

***Abstract*—Domain adaptation has shown promising results in computer vision applications. In this paper, we propose a new unsupervised domain adaptation method called domain adaptation by shifting covariance (DASC) for object recognition without requiring any labeled samples from the target domain. By characterizing samples from each domain as one covariance matrix, the source and target domain are represented into two distinct points residing on a Riemannian manifold. Along the geodesic constructed from the two points, we then interpolate some intermediate points (i.e., covariance matrices), which are used to bridge the two domains. By utilizing the principal components of each covariance matrix, samples from each domain are further projected into intermediate feature spaces, which finally leads to domain-invariant features after the concatenation of these features from intermediate points. In the multiple source domain adaptation task, we also need to effectively integrate different types of features between each pair of source and target domains. We additionally propose an SVM based method to simultaneously learn the optimal target classifier as well as the optimal weights for different source domains. Extensive experiments demonstrate the effectiveness of our method for both single source and multiple source domain adaptation tasks.**

***Index Terms*—Domain adaptation, riemannian manifold, support vector machine.**

I. Introduction

IN MANY real world applications, the domain of interest (i.e., the target domain) contains very few or even no labeled samples because the collection of labeled samples is generally expensive and time consuming. The goal of domain adaptation is to learn robust target classifiers in this scenario by leveraging a large number of labeled training samples from other domains (i.e., auxiliary/source domains). Domain adaptation methods have been successfully used in many computer vision applications (please refer to Section II for more details).

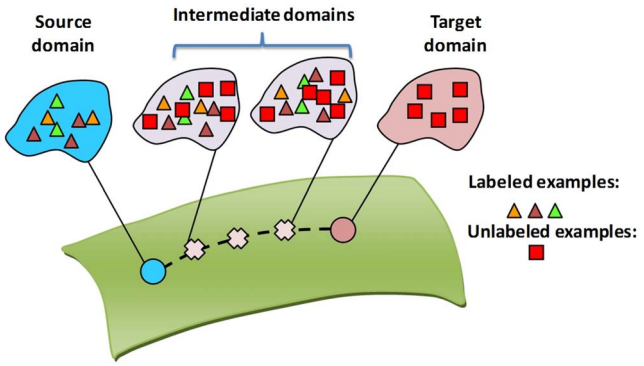


Fig. 1. Domain adaptation by using intermediate domains.

In domain adaptation, the samples from the source domain and the target domain often have different data distributions. To reduce the mismatch of data distributions, the technique which introduces intermediate domains (see Fig. 1) can be employed to extract domain invariant features to bridge the distribution mismatch between the source and target domain. Specifically, Gopalan *et al.* [1] proposed an unsupervised domain adaptation method [called sampling geodesic flow (SGF) in [2]] by representing each domain as one subspace spanned by principal components of samples. Then they constructed a few intermediate domains between the source and target domains by interpolating subspaces between the two subspaces. Gong *et al.* [2] further extended [1] and proposed a new kernel based approach called geodesic flow kernel (GFK) by integrating an infinite number of subspaces. However, modeling one domain as a subspace is not sufficient to represent the distribution difference between two domains, especially when the two subspaces intersect in a common subspace. We give two examples in Fig. 2, where we plot the samples from two domains (i.e., red points and blue triangles) in a 2-D intersected subspace of two domains. It can be observed that the distributions of the samples from two domains are different. However, existing subspace-based methods such as

Manuscript received May 14, 2013; revised November 23, 2013; accepted January 17, 2014. Date of publication March 11, 2014; date of current version November 13, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61125106, Grant 61202297, and Grant 61300138, in part by the Shaanxi Key Innovation Team of Science and Technology under Grant 2012KCT-04, in part by the Natural Science Foundation of Fujian Province under Grant 2013J01239, and in part by the Singapore MoE Tier 2 Project under Grant ARC42/13. This paper was recommended by Associate Editor D. Goldgof.

Z. Cui is with the College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China and also the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail:zhen.cui@vipl.ict.ac.cn).

W. Li and D. Xu are with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: wli1@e.ntu.edu.sg; dongxu@ntu.edu.sg).

S. Shan and X. Chen are with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: sgshan@ict.ac.cn; xlchen@ict.ac.cn).

X. Li is with the Center for Optical Imagery Analysis and Learning, State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (e-mail: xuelong_li@opt.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2014.2305701

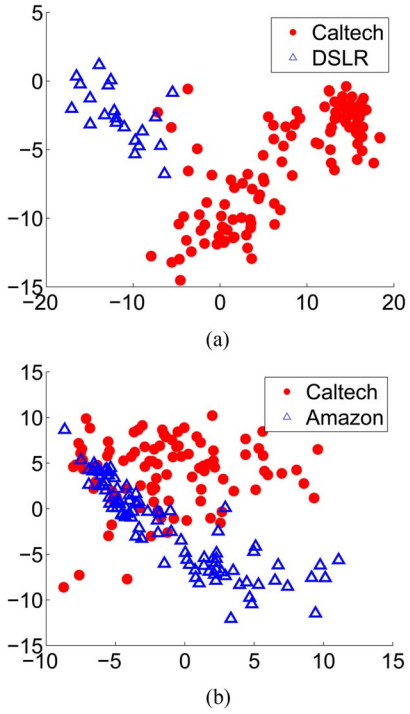


Fig. 2. Distributions of the samples from different domains in the 2-D intersected subspace. (a) Samples of touring-bike from Caltech and DSLR datasets. (b) Samples of video-projector from Caltech and Amazon datasets.

[1], [2] cannot be used to reduce the distribution mismatch in the intersected subspace, since the principal angles between the two subspaces of source and target domains are zero degrees in the intersected subspace (see Section III-A for more details).

In Section III, we propose a new unsupervised domain adaptation approach for object recognition, which can better reduce the domain distribution mismatch in all the feature dimensions without the aid of the subspaces as in [1] and [2]. According to [3], any symmetric positive-definite (SPD) matrix resides on a Riemannian manifold. Thus, we assume the covariance matrices of the samples from two domains can be represented as two distinct points on a Riemannian manifold. Then we propose a simple but effective approach to seek for a geodesic flow curve connecting these two points. Specifically, we first embed the two distinct points into the log-Euclidean space and interpolate a few isometric points in this space. By inversely mapping these interpolated points back into the original Riemannian manifold, we obtain a set of interpolated covariance matrices. Then, we learn the projection matrices from these covariance matrices by using PCA, with which the samples from both domains can be projected into a set of intermediate feature representations that bridge the two domains. Linear discriminant analysis (LDA) [4] can be further used to extract the discriminant features. Finally, we concatenate all the features to form domain-invariant features and employ the nearest neighbor (NN) or support vector machine (SVM) classifier for the single source domain adaptation task.

For multi-domain adaptation, the domain-invariant features from each pair of source and target domains are different, so the existing multiple source domain adaptation methods cannot be directly used to learn the optimal target classifier and select the most relevant source domains. In Section IV, we propose to simultaneously learn the optimal target classifier and the optimal weights for different source domains. In Section V, we conduct comprehensive experiments using the office [5] and the Caltech256 [6] datasets under both single source domain and multi-domain settings. The results clearly demonstrate the effectiveness of our DASC as well as our new SVM-based learning algorithm for domain adaptation tasks.

II. Related Work

Domain adaptation methods have been used for document classification [7], object recognition [1], [2], [5], [8]–[10], object localization [11], face recognition [12], indoor location [13], event recognition [14], [15], and video concept detection [7], [16].

In general, these methods can be categorized as feature (transform)-based approaches and classifier-based approaches. The feature (transform)-based approaches try to learn domain-invariant features for domain adaptation. Pan *et al.* [17] proposed to learn the feature mapping by using the maximum mean discrepancy embedding. Saenko *et al.* [5] proposed a metric learning method by enforcing the samples that are from the same class but different domains to be closer with each other, which was further generalized in [8] by learning an asymmetric nonlinear transformation. However, their methods utilized the labeled data from the target domain, and hence cannot be applied to unsupervised domain adaptation. Gopalan *et al.* [1] and Gong *et al.* [2] proposed two methods to reduce the domain distribution mismatch based on the Grassmann manifold assumption. Recently, Zhu proposed a probabilistic graphic model based method [18], and Shi *et al.* [10] proposed an information theory-based method for domain adaptation.

The classifier-based approaches [16], [19], [20] directly seek target classifiers (e.g., SVM-based classifiers) for domain adaptation. Yang *et al.* [19] developed Adaptive SVM (A-SVM) by leveraging the existing source classifiers. Duan *et al.* [16] proposed a new multiple kernel learning (MKL)-based approach for domain adaption by simultaneously learning the optimal linear weight coefficients of base kernels and the target classifier. Besides the two categories, other methods were also proposed, such as structure correspondence learning [21], sample reweighting [22], and feature replication [23].

Multiple source domain adaptation methods [7], [15], [24], [25] have also been studied. Schweikert *et al.* [24] proposed to match the means of different domains based on kernel mean matching (KMM) [22]. Duan *et al.* [7], [15] proposed two multi-domain adaptation methods by leveraging or selecting source domains. Chattopadhyay *et al.* [25] also proposed to weight different source domains based on a smoothness regularizer. Hoffman *et al.* [9] proposed a new clustering-based approach to partition a dataset into latent domains and extended [8] for multi-domain adaptation by learning multiple transformations. Moreover, some theoretical results can be found in [26]–[28], and [29]. Please refer to [30] for a comprehensive survey on transfer learning.

III. Proposed Method

In this paper, we denote a vector/matrix by a lowercase/uppercase letter in boldface. The transpose of a vector or matrix is denoted by the superscript $'$. Moreover, we use $\mathbf{O}_{m\times n}$ to represent an m-by-n matrix with all zeros and $\mathbf{I}_n$ to represent an n-by-n identity matrix, respectively. We also define $\mathbf{1}$ as the vector with all ones. The inequality $\mathbf{u} = [u_1, \ldots, u_n]' \geq 0$ means that $u_i \geq 0$ for $i = 1, \ldots, n$.

In the common setting of the domain adaptation (DA) problem, the source domain samples are labeled and the target domain contains no labeled samples or only a limited number of labeled samples, which are referred to as unsupervised DA and semi-supervised DA, respectively. In this paper, we focus on the unsupervised DA. However, the proposed approach can be easily applied to semi-supervised DA. Formally, let us denote $\mathbf{X}^s = [\mathbf{x}_1^s, \ldots, \mathbf{x}_{N_s}^s] \in \mathbb{R}^{D\times N_s}$ as the source domain data and denote $y_i^s \in \{1, 2, \ldots, K\}$ as the label of the ith sample $\mathbf{x}_i^s$, where D is the feature dimension, N_s is the total number of source samples and K is the total number of classes. Similarly, we denote $\mathbf{X}^t = [\mathbf{x}_1^t, \ldots, \mathbf{x}_{N_t}^t] \in \mathbb{R}^{D\times N_t}$ as the target domain data from the same K classes, where N_t is the total number of target samples.

A. Subspace Analysis and Motivation

For domain adaptation, the key issue is to reduce the mismatch on the data distributions of two domains. Since the intermediate domains can contain more shared information of two domains as shown in Fig. 1, projecting the examples into the intermediate domains makes their distributions better matched, and further leads to higher discriminability in these auxiliary domains.

SGF [1] and GFK [2] characterize each domain with a subspace spanned from one column-orthogonal matrix, and then interpolate the intermediate subspaces between two domains (or subspaces). Specifically, by performing PCA on the examples from each domain, the source and target domains can be regarded as two subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$, which span from their corresponding principal components $\mathbf{S}_1, \mathbf{S}_2 \in \mathbb{R}^{D\times d}$. Therefore, the intermediate domains become the interpolated points between $\mathcal{S}_1$ and $\mathcal{S}_2$. Furthermore, the interpolation process can be performed by gradually rotating principal angles of $\mathcal{S}_1$ and $\mathcal{S}_2$ until full overlap. Formally, the interpolated points are $\mathbf{S}(\tau) = \mathbf{T}\begin{bmatrix} \text{diag}\{\cos(\tau\theta_1), \ldots, \cos(\tau\theta_d)\} \\ \text{diag}\{\sin(\tau\theta_1), \ldots, \sin(\tau\theta_d)\} \end{bmatrix}$, where $0 \leq \tau \leq 1$, $\mathbf{T}$ is a transform matrix, diag is the diagonalization operation, and $\theta_1, \ldots, \theta_d$ are the principal angles between $\mathcal{S}_1$ and $\mathcal{S}_2$ [31]. The principal angles $\theta_1, \ldots, \theta_d \in [0, \pi/2]$ are defined as follows:

$$\cos(\theta_j) = \max_{\mathbf{u}_j\in\mathcal{S}_1}\max_{\mathbf{v}_j\in\mathcal{S}_2} \mathbf{u}_j'\mathbf{v}_j \tag{1}$$

$$\text{s.t. } \mathbf{u}_j'\mathbf{u}_j = \mathbf{v}_j'\mathbf{v}_j = 1, \mathbf{u}_j'\mathbf{u}_i = \mathbf{v}_j'\mathbf{v}_i = 0, i < j. \tag{2}$$

Intuitively, the first principal angle θ_1 represents the smallest angle between all pairs of basis vectors in two subspaces, the rest of the principal angles are similarly defined between the complementary subspaces of two subspaces with each subspace being spanned by the corresponding selected basis vectors.

However, when two subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$ intersect into a r-dim subspace $\hat{\mathcal{S}} = \mathcal{S}_1 \bigcap \mathcal{S}_2$, the first r principal angles $\theta_1, \ldots, \theta_r$ must be equal to zero-degree, where we only need to assign the r principal components of the common subspace $\hat{\mathcal{S}}$ to $\{\mathbf{u}_1 = \mathbf{v}_1, \mathbf{u}_2 = \mathbf{v}_2, \cdots, \mathbf{u}_r = \mathbf{v}_r\}$ according to (1) and (2). As a result, the first r columns of the intermediate point $\mathcal{S}(\tau)$, corresponding to the common subspace $\hat{\mathcal{S}}$, remain unchangeable when τ ranges from 0 to 1. In this case, after projecting these samples into the intermediate subspaces as used in SGF and GFK, the distributions of the samples from two domains in this intersected subspace are still unchanged. Empirically, given two domains, their intersection subspace is usually nonempty especially when d is larger than $D/2$, and meanwhile the distributions of samples from two domains are usually different in the intersection subspace. Two examples are shown in Fig. 2, where the samples are projected into the 2-D space by using the first two dimensions of PCA on the common subspace.

To address the above problem, we directly represent each domain by a covariance matrix instead of a subspace (i.e., a column-orthogonal matrix). Due to the robustness of the covariance matrix for characterizing the data distribution, we then use the intermediate covariance matrices between two domains to reduce the data distribution mismatch.

B. Domain Adaptation by Shifting Covariance

We denote the covariance matrices of the source domain and the target domain as $\mathbf{C}^s$ and $\mathbf{C}^t$, respectively, which are symmetric positive-definite (SPD) matrices.[1] Since D-by-D SPD matrices lie on a Riemannian manifold, so each domain corresponds to a point on this manifold. To bridge the source and target domains, we seek a geodesic path $g(t)|_{t=0}^1$ from $\mathbf{C}^s$ to $\mathbf{C}^t$ on the Riemannian manifold in which we expect $g(0) = \mathbf{C}^s$, $g(1) = \mathbf{C}^t$, and $g(t)$ is a covariance matrix flowing on the geodesic path with gradually changing from $\mathbf{C}^s$ to $\mathbf{C}^t$ while t increases from 0 to 1.

By flowing on the geodesic path $g(t)$, the covariance difference between two domains can be gradually reduced and finally it reaches the target covariance. Nevertheless, it is nontrivial to seek such a geodesic path on a Riemannian manifold. partial differential equations (PDEs) may be used, however, it is usually computationally expensive. Below we employ a much more efficient approach called Log-Euclidean metric which can achieve the same excellent theoretical properties on a Riemannian manifold [3].

C. Computation With Log-Euclidean Metric

Before introducing the Log-Euclidean metric, we first give the definition of matrix exponential $\exp(\cdot)$ and logarithm $\log(\cdot)$ of a covariance matrix $\mathbf{C}$ as follows:

$$\exp(\mathbf{C}) = \mathbf{U}\exp(\mathbf{\Lambda})\mathbf{U}', \quad \log(\mathbf{C}) = \mathbf{U}\log(\mathbf{\Lambda})\mathbf{U}'$$

where $\mathbf{C} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}'$ is from singular value decomposition (SVD). Then, we further define the logarithmic multiplication $\odot$ and

[1] We can always add a small positive value to the diagonal elements when a covariance matrix is not SPD.

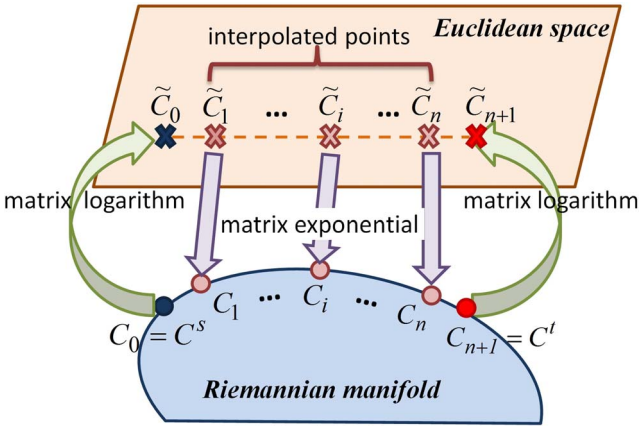


Fig. 3. Generating intermediate covariances. Based on the Log-Euclidean metric [3], we first transform the covariance matrices $\mathbf{C}_0, \mathbf{C}_{n+1}$ as $\tilde{\mathbf{C}}_0, \tilde{\mathbf{C}}_{n+1}$ from the Riemannian manifold to the Log-Euclidean space by the matrix logarithm operation, and then interpolate n isometric points $\tilde{\mathbf{C}}_i|_{i=1}^n$ along the line $(\tilde{\mathbf{C}}_0, \tilde{\mathbf{C}}_{n+1})$. Finally, the matrix exponential operation is applied on $\tilde{\mathbf{C}}_i$ to obtain n intermediate covariance matrices $\mathbf{C}_i, i = 1, \cdots, n$.

the logarithmic scalar multiplication $\otimes$ on the SPD matrices space as

$$\mathbf{C}_1 \odot \mathbf{C}_2 = \exp(\log(\mathbf{C}_1) + \log(\mathbf{C}2)),$$
$$\lambda \otimes \mathbf{C}_1 = \exp(\lambda \log(\mathbf{C}_1)) = \mathbf{C}_1^{\lambda}$$

where $\mathbf{C}_1$ and $\mathbf{C}_2$ are two SPD matrices and λ is a scalar. When the SPD matrix space is associated with the multiplication $\odot$, it is actually a lie group [3], which may induce a bi-invariant metric, i.e., Log-Euclidean metric in [3] (referred as LEM here). Formally, the distance of two SPD matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ based on LEM can be written as

$$\mathrm{DIST}(\mathbf{C}_1, \mathbf{C}_2) = \| \log(\mathbf{C}_1) - \log(\mathbf{C}_2) \|_F$$

where $\|\cdot\|_F$ is the matrix Frobenius norm. With LEM, the SPD matrix space is isomorphically and isometrically mapped to the Euclidean space of symmetric matrices [3], which means many calculations on Riemannian manifold can be simplified in the Euclidean space. Specifically, we can calculate the interpolated point on the geodesic path $g(t)$ using matrix exponential and logarithm as

$$\mathbf{C}(t) = \exp\left((1-t)\log(\mathbf{C}^s) + t\log(\mathbf{C}^t)\right). \tag{3}$$

Therefore, to seek the geodesic path from the source domain to the target domain, we first map the covariance matrices $\mathbf{C}^s$ and $\mathbf{C}^t$ into the Log-Euclidean space with the matrix logarithm. Given any $0 \leq t \leq 1$, we then obtain the interpolated point along the line between $\log(\mathbf{C}^s)$ and $\log(\mathbf{C}^t)$ in the Euclidean space. After that, we use matrix exponential to obtain the SPD matrix on Riemannian manifold. The procedure is illustrated in Fig. 3.

D. Performing Recognition with DASC

As shown in Fig. 3, we denote these interpolated covariance matrices as $\mathbf{C}_j|_{j=1}^n$, and also denote the covariance matrices of the source domain and the target domain as $\mathbf{C}_0$ and $\mathbf{C}_{n+1}$, respectively. These $n+2$ covariance matrices actually represent $n+2$ intermediate domains that bridge the source domain and the target domain.

To obtain the domain-invariant features, a common way is to project the original features from the source/target domain into those intermediate domains. To this end, we perform principal component analysis (PCA) [4] on each covariance matrix to obtain the projection matrix. Formally, we denote $\mathbf{P}_j|_{j=0}^{n+1} \in \mathbb{R}^{D\times d}$ as the projection matrices obtained from those $n+2$ covariance matrices using PCA. For any sample $\mathbf{x}_i$ in the source/target domain (i.e., $\mathbf{x}_i^s$ or $\mathbf{x}_i^t$), the projected feature by using $\mathbf{P}_j$ can be computed by $\hat{\mathbf{x}}_{ij} = \mathbf{P}_j'\mathbf{x}_i$. Then, we perform LDA [4] using the labeled samples in each intermediate domain to obtain the discriminative representation of each sample, denoted by $\mathbf{z}_{ij}$. The labeled data are from the source domain (resp., both the source and the target domain) for unsupervised DA (resp., semi-supervised DA). Finally, the $n+2$ features are concatenated to form the domain-invariant feature by $\mathbf{z}_i = [\mathbf{z}_{i0}', \ldots, \mathbf{z}_{i(n+1)}']'$. After obtaining the domain-invariant features for the samples in both domains, any type of classifier (e.g., SVM classifier or NN classifier) can be employed to perform the recognition task.

IV. Multiple Source Domain Adaptation

In this section, we first introduce the problem of multiple source DA, then we propose an SVM based method to learn the target classifier as well as the weights of source domains. Finally, we discuss the most related work with our proposed method.

A. Problem

When multiple source domains are available, multiple geodesic paths can be constructed on Remainnian manifold by connecting each source domain with the target domain. Formally, given M source domains, $\mathbf{X}^{s,1}, \mathbf{X}^{s,2}, \ldots, \mathbf{X}^{s,M}$, we can obtain M pairs of domain-invariant features after performing feature extraction with our DASC method, that is $(\mathbf{Z}^{s,1}, \mathbf{Z}^{t,1}), (\mathbf{Z}^{s,2}, \mathbf{Z}^{t,2}), \ldots, (\mathbf{Z}^{s,M}, \mathbf{Z}^{t,M})$ where $\mathbf{Z}^{s,m} = [\mathbf{z}_1^{s,m}, \ldots, \mathbf{z}_{N_m}^{s,m}]$ and $\mathbf{Z}^{t,m} = [\mathbf{z}_1^{t,m}, \ldots, \mathbf{z}_{N_t}^{t,m}]$ are the domain-invariant features of the mth source domain data $\mathbf{X}^{s,m}$ and the target domain data $\mathbf{X}^t$ generated from the mth geodesic path, N_m and N_t are the numbers of source and target domain samples. Note that the features from the same pair of source and target domains (e.g. $\mathbf{Z}^{s,m}$ and $\mathbf{Z}^{t,m}$) are homogenous, because they are obtained by using the same projection matrix. However, the features generated from different pathes (e.g. $\mathbf{Z}^{s,m}$ and $\mathbf{Z}^{s,\hat{m}}$ for any $m \neq \hat{m}$) are heterogeneous because different projection matrices are used. A straightforward way for multiple source domain adaptation is to independently learn one classifier for each pair of source and target domains and then fuse multiple classifiers in a late-fusion fashion. However, some source domains might be more relevant to the target domain, so the corresponding classifiers should be more important for the final classification. To this end, we propose to jointly learn multiple classifiers as well as the optimal weights of source domains.

B. Formulation

To simplify the notation, we denote $\{\mathbf{z}_i^m|_{i=1}^{N^m}\}$ as the samples using the mth type of domain-invariant features, where N^m is defined as $N^m = N_l^m + N_u$ with N_l^m being the number of labeled samples and N_u being the number of unlabeled samples. In the task of unsupervised DA, the labeled samples come from source domains and unlabeled samples are the target domain data.

Given each type of domain-invariant features, we propose to train one binary SVM classifier by using the labeled samples from the source domain as well as the unlabeled samples from the target domain. Formally, let us denote the classifier for the mth type of features as $f^m(\mathbf{z}_i^m) = \mathbf{w}_m'\phi_m(\mathbf{z}_i^m) + b_m (i = 1, \ldots, N^m)$, where $\phi_m(\mathbf{z}_i^m)$ is the nonlinear feature mapping function, $\mathbf{w}_m$ and b_m are the weight vector and the bias of the SVM classifier, respectively. When learning those M classifiers, we also propose to learn the weight coefficient γ_m for each classifier $f^m(\mathbf{z}_i^m)$. To this end, we formulate the learning task as follows:

$$\min_{\substack{\boldsymbol{\gamma},\mathbf{w}_m,b_m,\\ \mathbf{f}^m,\xi_i^m,\xi_i^{m*}}} \sum_{m=1}^{M}\left\{\frac{1}{2\gamma_m}\|\mathbf{w}_m\|^2 + C\sum_{i=1}^{N^m}(\xi_i^m + \xi_i^{m*}) + \frac{\theta}{2}(\|\mathbf{f}_l^m - \mathbf{y}_l^m\|^2 + \gamma_m\|\mathbf{f}_u^m - \mathbf{v}_u\|^2)\right\}, \tag{4}$$

$$\text{s.t. } \mathbf{w}_m'\phi_m(\mathbf{z}_i^m) + b_m - f_i^m \le \varepsilon + \xi_i^m, \xi_i^m \ge 0, \tag{5}$$

$$f_i^m - \mathbf{w}_m'\phi_m(\mathbf{z}_i^m) - b_m \le \varepsilon + \xi_i^{m*}, \xi_i^{m*} \ge 0, \tag{6}$$

$$1 \ge \boldsymbol{\gamma} \ge 0 \tag{7}$$

where these terms are described as follows:

1) $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_M]'$: the vector of weight coefficients for M classifiers;
2) $\mathbf{y}_l^m = [y_1^m, \ldots, y_{N_l^m}^m]'$: the label vector of labeled samples from the mth pair of source and target domains, with $y_i^m \in \{+1, -1\}$ for $i = 1, \ldots, N_l^m$;
3) $\mathbf{v}_u = [v_1, \ldots, v_{N_u}]'$: the virtual labels of unlabeled samples in the target domain;
4) $\mathbf{f}^m = [\mathbf{f}_l^{m'}, \mathbf{f}_u^{m'}]'$: the expected decision value vector for the samples when using the mth type of features, where $\mathbf{f}_l^m = [f_1^m, \ldots, f_{N_l^m}^m]'$ and $\mathbf{f}_u^m = [f_{N_l^m+1}^m, \ldots, f_{N_l^m+N_u}^m]'$ are the vectors for labeled and unlabeled samples, respectively;
5) ξ_i^m and ξ_i^{m*}: slack variables for ε-insensitive loss;
6) C and θ: two regularization parameters.

Specifically, the first term in (4) is to regularize the complexity of the learned classifiers, which is similar to MKL. The second term is the ε-insensitive loss which enforces the decision values of learned classifiers (i.e., $\mathbf{w}_m'\phi_m(\mathbf{z}_i^m) + b_m$) to be close to the expected decision values (i.e., f_i^m). In the third term, we constrain the expected decision values of labeled data $\mathbf{f}_l^m$ to as close to their labels $\mathbf{y}_l^m$ as possible, and we also introduce a data-dependent regularizer on the unlabeled data to constrain the expected decision values on the target domain unlabeled samples (i.e., $\mathbf{f}_u^m$) and learn the optimal weights for source domains (i.e. $\boldsymbol{\gamma}$).

1) *Data-dependent Regularizer:* The data-dependent regularizer in the third term of (4) not only enforces the expected decision values of unlabeled data from different classifiers to be consistent with the virtual labels, but also encourages the classifiers from relevant source domains to have higher weights. Specifically, if the mth source domain is more relevant to the target domain, then the expected decision values (i.e., $\mathbf{f}_u^m$) should be closer to the virtual labels of unlabeled data $\mathbf{v}_u$. In this case, it will encourage the corresponding γ_m to be larger. On the other hand, a larger γ_m will also enforce the corresponding $\mathbf{f}_u^m$ to be closer to $\mathbf{v}_u$.

2) *Virtual Labels:* To construct the virtual label vector $\mathbf{v}_u$, we first train an SVM classifier for each pair of domains separately, and use them to predict the decision values of the unlabeled samples. Given an unlabeled sample $\mathbf{x}$, let us denote these M decision values by $p_m|_{m=1}^M$, then the virtual label of $\mathbf{x}$ is calculated by using the weighted sum of these M decision values, which is defined as $\sum_{m=1}^M e^{-\frac{1}{\delta}d_m^2}p_m$ where d_m is the distance between $\mathbf{x}$ and the closest sample from the mth source domain, the parameter δ is used to decide the weights of local neighbors.

C. Optimization

We alternatingly optimize the weight coefficient vector $\boldsymbol{\gamma}$ and other variables related to the classifier.

1) *Fix $\boldsymbol{\gamma}$:* When fixing $\boldsymbol{\gamma}$, the optimization problem in (4) is separable with respect to m, $m = 1, \ldots, M$. We therefore optimize these M subproblems one by one. To simplify the presentation, we omit the subscript m below unless necessary.

For each subproblem, we introduce the Lagrangian multipliers α_i's and η_i's (resp. α_i^*'s and η_i^*'s) for the constraints in (5) (resp., (6)), and set the derivatives of the Lagrangian with respect to the primal variables ($\mathbf{f}^m, \mathbf{w}_m, b_m, \xi_i^m$ and ξ_i^{m*}) to zeros, respectively. Then, we obtain the following equations:

$$\mathbf{w}_m = \gamma_m\boldsymbol{\Phi}_m(\boldsymbol{\alpha}^* - \boldsymbol{\alpha}), \tag{8}$$

$$\mathbf{f}^m = \begin{pmatrix}\mathbf{y}_l^m\\ \mathbf{v}_u\end{pmatrix} + \begin{pmatrix}\mathbf{I}_{N_l^m} & \mathbf{O}_{N_l^m\times N_u}\\ \mathbf{O}_{N_u\times N_l^m} & \frac{1}{\gamma_m}\mathbf{I}_{N_u}\end{pmatrix}\frac{\boldsymbol{\alpha} - \boldsymbol{\alpha}^*}{\theta} \tag{9}$$

as well as the constraints: $\mathbf{1}'\boldsymbol{\alpha} = \mathbf{1}'\boldsymbol{\alpha}^*$, $0 \le \boldsymbol{\alpha}, \boldsymbol{\alpha}^* \le C$ where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{N^m}]'$, $\boldsymbol{\alpha}^* = [\alpha_1^*, \ldots, \alpha_{N^m}^*]'$, and $\boldsymbol{\Phi}_m = [\phi_m(\mathbf{z}_1^m), \ldots, \phi_m(\mathbf{z}_{N^m}^m)]$.

Substituting those equations and constraints back into the Lagrangian, we arrive at the following dual form:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} \frac{1}{2}(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*)'\tilde{\mathbf{K}}_m(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + \tilde{\mathbf{y}}^{m\prime}(\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + \varepsilon\mathbf{1}'(\boldsymbol{\alpha} + \boldsymbol{\alpha}^*) \tag{10}$$

$$\text{s.t.} \quad \mathbf{1}'\boldsymbol{\alpha} = \mathbf{1}'\boldsymbol{\alpha}^*, \ 0 \le \boldsymbol{\alpha}, \boldsymbol{\alpha}^* \le C \tag{11}$$

where $\tilde{\mathbf{y}}^m = \begin{pmatrix}\mathbf{y}_l^m\\ \mathbf{v}_u\end{pmatrix}$, $\tilde{\mathbf{K}}_m = \gamma_m\mathbf{K}_m + \frac{1}{\theta}\begin{pmatrix}\mathbf{I}_{N_l^m} & \mathbf{O}_{N_l^m\times N_u}\\ \mathbf{O}_{N_u\times N_l^m} & \frac{1}{\gamma_m}\mathbf{I}_{N_u}\end{pmatrix}$ and $\mathbf{K}_m = \boldsymbol{\Phi}_m'\boldsymbol{\Phi}_m$ is the kernel matrix using the mth type of features. Actually, the above dual formulation is in analogous to the support vector regression (SVR) formulation, and can be readily solved with LIBSVM [32].

2) *Fix $\mathbf{f}^m, \mathbf{w}_m$ and b_m:* When each classifier is known, $\gamma_m|_{m=1}^M$ can be optimized as

$$\min_{\boldsymbol{\gamma}} \frac{1}{2}\sum_{m=1}^{M}\left(\frac{1}{\gamma_m}\|\mathbf{w}_m\|^2 + \theta\gamma_m\|\mathbf{f}_u^m - \mathbf{v}_u\|^2\right) \tag{12}$$

$$\text{s.t. } 1 \ge \boldsymbol{\gamma} \ge 0.$$

Algorithm 1: Multiple Source Domain Adaptation

Input: Source domain data $\mathbf{X}^{s,m}(m = 1, \ldots, M)$ with the corresponding label vector $\mathbf{y}_l^m$, and unlabeled target domain data $\mathbf{X}^t$.

Obtain M domain-invariant features $\{\mathbf{z}_i^m, i = 1, \ldots, N_l^m, \ldots, N^m\}, m = 1, \ldots, M$ by using our DASC method as described in Section III.

Training:

Generate the virtual label vector $\mathbf{v}_u$ (Section IV-B);

Initialize $t \leftarrow 1$.

repeat

  if $t = 1$ **then**

    Set the weight coefficient vector $\boldsymbol{\gamma} \leftarrow \mathbf{1}/M$.

  else

    Based on the learnt $(\boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m^*)$, calculate $\mathbf{w}_m$ and $\mathbf{f}^m$ by using (8) and (9), respectively.

    Solve for the weight coefficient vector $\boldsymbol{\gamma}$ as $\gamma_m = \min\{\sqrt{\|\mathbf{w}_m\|^2/(\theta\|\mathbf{f}_u^m - \mathbf{v}_u\|^2)}, 1\}$ and $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma}/(\mathbf{1}'\boldsymbol{\gamma})$.

  end if

  Obtain $(\boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m^*)|_{m=1}^M$ by solving M SVR problems one by one as in (10) using LIBSVM [32].

  Set $t \leftarrow t + 1$;

until The change of the objective in (4) is less than a predefined threshold.

Testing:

Predict the labels of target domain samples by using (13).

Output: The labels of target domain samples.

where $\mathbf{w}_m$ and $\mathbf{f}_u^m$ can be respectively calculated from (8) and (9) by using the learnt dual variables $(\boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m^*)$.

The above problem is a convex optimization problem with a box constraint. By setting the derivative with respect to $\boldsymbol{\gamma}$ to zero and applying the box constraint, we reach the solution as $\gamma_m = \min\{\sqrt{\|\mathbf{w}_m\|^2/(\theta\|\mathbf{f}_u^m - \mathbf{v}_u\|^2)}, 1\}$, and then normalize $\boldsymbol{\gamma}$ as $\boldsymbol{\gamma} \leftarrow \frac{\boldsymbol{\gamma}}{(\mathbf{1}'\boldsymbol{\gamma})}$.

D. Algorithm

We describe the whole algorithm for multiple source domain adaptation in Algorithm 1. We first initialize the weight coefficient vector to $\frac{1}{M}\mathbf{1}$. Then, we solve the M SVR problems one by one as in (10) to obtain the dual variables $(\boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m^*)|_{m=1}^M$. After that, we can calculate $\mathbf{w}_m$ and $\mathbf{f}^m$, respectively, using (8) and (9), and obtain $\boldsymbol{\gamma}$ by solving (12). The weight coefficient vector $\boldsymbol{\gamma}$ is further normalized as $\boldsymbol{\gamma} \leftarrow \frac{\boldsymbol{\gamma}}{\mathbf{1}'\boldsymbol{\gamma}}$. We repeat the above two steps until the change of the objective value in (4) is less than a predefined threshold. Actually, this algorithm always converges in our experiments (see Section V-D for the details).

Finally, the prediction of a target unlabeled sample $\mathbf{x}$ can be obtained by fusing these M classifiers, that is

$$f(\mathbf{x}) = \sum_{m=1}^{M} \gamma_m(\phi_m(\mathbf{z}^m)'\boldsymbol{\Phi}_m(\boldsymbol{\alpha}_m^* - \boldsymbol{\alpha}_m) + b_m) \tag{13}$$

where $\mathbf{z}^m$ is the mth feature of $\mathbf{x}$, and $\boldsymbol{\alpha}_m$ and $\boldsymbol{\alpha}_m^*$ are the learnt dual variables from the mth subproblem in (10).

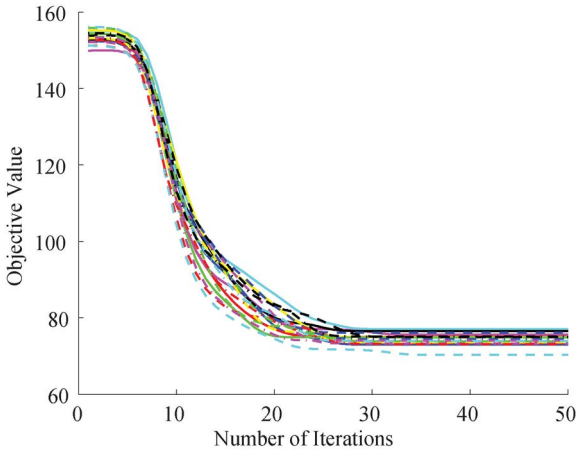


Fig. 4. Convergence characteristics of the objective function in (4) from 20 rounds of experiments with different randomly selected training samples. The target domain is set to Caltech.

E. Discussion

The most related works are domain adaptation machine (DAM) [7] and domain selection machine (DSM) [15] since similar data-dependent regularizers were also used. Our method is intrinsically different with [7] and [15]. Specifically, both methods [7] and [15] are designed for homogenous sources, while our method aims to handle heterogeneous sources. Only one type of feature for unlabeled target data is considered in their regularizers [7], [15]. In contrast, we consider different types of features for unlabeled target data in our data-dependent regularizer. Moreover, the source classifiers in DAM and DSM are prelearnt. In contrast, we also learn the source classifiers for different sources to better cope with heterogeneous sources. Recently, the work in [36] studied the same domain adaptation setting, where the samples from different source domains may be represented by different types of features while the samples in the target domain have all types of features. However, their work [36] needs to infer the labels of target domain samples, which is computationally expensive. In contrast, we use the data-dependent regularizer to utilize the unlabeled target domain samples without inferring their labels, thus our work is more efficient.

V. Experiments

In this section, we evaluate our methods on two unsupervised domain adaptation settings: single source domain adaptation and multiple source domain adaptation.

A. Experimental Setup

For single source domain adaptation, we strictly follow the experimental setting in [2] by using four datasets with ten common classes, i.e., Amazon, Webcam, and DSLR collected in [5], and Caltech-256 [6]. Similar to [1], [2], and [5], we extract SURF features [33] and encode the images with 800-bin token frequency (TF) feature by using a codebook trained from a subset of Amazon images. Then, the features are normalized and z-scored to have zero mean and unit standard deviation in each dimension.

TABLE I

MEANS AND STANDARD DEVIATIONS OF CLASSIFICATION ACCURACIES (%) FOR UNSUPERVISED SINGLE SOURCE DOMAIN ADAPTATION. FIRST SIX CASES (A: AMAZON, C: CALTECH, D: DSLR, W: WEBCAM)

Method			C→A	C→W	C→D	A→C	A→W	A→D
NN	**OrigFeat**		27.79±0.55	25.75±1.05	25.73±0.90	24.58±0.33	26.97±0.52	22.64±0.76
	SGF [1]	PCA	*37.90±0.50*	*34.19±0.98*	*38.41±0.85*	*33.62±0.30*	***33.81±0.66***	*34.81±0.98*
		PLS	44.34±0.39	36.53±0.77	39.20±0.82	36.93±0.30	33.59±0.60	34.46±0.66
		PCA+LDA	43.48±0.42	34.61±0.72	40.73±0.87	37.60±0.34	33.76±0.74	36.18±0.69
	GFK [2]	(PCA,PCA)	*37.32±0.73*	*33.83±1.34*	*36.40±1.14*	*31.13±0.44*	*30.95±1.07*	*32.32±0.97*
		(PLS,PCA)	40.65±0.53	39.76±1.52	43.41±1.15	34.10±0.43	34.20±0.83	37.07±1.02
		(LDA,PCA)	41.63±0.66	38.69±1.55	42.99±1.00	31.95±0.22	33.51±0.71	34.62±1.04
	ITL [10]		37.85±0.40	34.36±0.69	39.33±0.77	33.82±0.22	33.56±0.70	34.33±0.82
	DASC	PCA	***38.87±0.56***	***36.59±1.05***	***40.96±0.93***	***33.83±0.29***	*33.69±0.68*	***35.57±0.85***
		PCA+LDA	**47.58±0.54**	**41.64±0.95**	**44.17±0.81**	**38.15±0.43**	**38.00±0.91**	**39.17±0.68**
SVM	**OrigFeat**		45.35±0.36	39.56±0.87	43.06±0.80	38.18±0.33	35.46±0.46	38.50±0.53
	SGF	PCA	46.46±0.42	41.44±0.85	45.16±1.00	39.23±0.31	37.22±0.65	38.98±0.76
		PLS	48.85±0.65	42.88±0.77	44.04±0.98	39.94±0.33	34.93±0.69	34.84±0.62
	GFK	(PCA,PCA)	46.65±0.62	40.44±0.88	42.17±0.91	38.17±0.36	35.63±0.62	34.81±0.94
		(PLS,PCA)	47.63±0.60	42.73±0.81	44.17±0.92	38.82±0.45	36.03±0.61	36.08±0.9
	ITL		48.53±0.45	43.31±0.71	47.20±0.97	**39.65±0.29**	37.47±0.56	**39.59±0.64**
	DASC		**49.83±0.44**	**45.42±0.88**	**48.47±0.77**	39.09±0.33	**37.68±0.68**	39.33±0.83

TABLE II

MEANS AND STANDARD DEVIATIONS OF CLASSIFICATION ACCURACIES (%) FOR UNSUPERVISED SINGLE SOURCE DOMAIN ADAPTATION. REMAINING SIX CASES

Method			W→C	W→A	W→D	D→C	D→A	D→W	Average
NN	**OrigFeat**		19.29±0.56	20.10±0.31	50.35±0.79	24.81±0.43	28.21±0.35	61.10±0.78	29.77
	SGF [1]	PCA	*28.88±0.39*	*32.58±0.44*	*74.20±0.79*	*31.80±0.41*	*33.56±0.39*	*82.22±0.43*	*41.33*
		PLS	29.10±0.38	32.76±0.42	65.67±0.59	31.62±0.39	32.91±0.41	73.42±0.65	40.88
		PCA+LDA	30.27±0.40	34.79±0.39	67.71±0.64	32.39±0.32	34.20±0.38	76.93±0.46	41.89
	GFK [2]	(PCA,PCA)	*30.45±0.32*	***34.74±0.49***	*72.74±0.87*	*31.26±0.22*	*33.99±0.38*	*82.85±0.45*	*40.66*
		(PLS,PCA)	31.59±0.40	35.97±0.61	73.57±1.36	32.37±0.37	**36.66±0.46**	**84.85±0.40**	43.68
		(LDA,PCA)	29.76±0.50	34.55±0.45	62.32±0.84	30.50±0.49	32.83±0.66	71.90±0.44	40.44
	ITL [10]		30.57±0.28	33.05±0.49	**76.82±0.71**	31.51±0.37	33.40±0.23	83.92±0.22	41.88
	DASC	PCA	***31.33±0.27***	*33.86±0.56*	***79.90±0.80***	***33.66±0.31***	***36.41±0.29***	***86.83±0.30***	***43.46***
		PCA+LDA	**33.46±0.38**	**36.39±0.43**	76.18±0.71	**34.61±0.36**	36.10±0.26	84.34±0.46	**45.82**
SVM	**OrigFeat**		30.34±0.38	34.35±0.34	69.68±0.70	32.59±0.28	33.64±0.28	76.59±0.66	43.11
	SGF	PCA	32.14±0.25	35.08±0.39	68.34±0.64	34.34±0.28	34.78±0.29	78.64±0.44	44.32
		PLS	32.26±0.37	35.10±0.47	72.90±0.65	34.88±0.33	34.68±0.35	82.05±0.57	44.78
	GFK	(PCA,PCA)	29.18±0.33	31.65±0.62	70.35±0.78	29.51±0.31	31.21±0.57	71.80±0.91	41.80
		(PLS,PCA)	28.37±0.88	31.56±0.61	71.37±0.85	29.37±0.33	32.03±0.57	71.44±0.89	42.47
	ITL		32.15±0.31	35.16±0.33	75.64±0.82	34.69±0.27	35.25±0.20	83.66±0.36	46.02
	DASC		**33.33±0.32**	**36.25±0.37**	**79.75±0.86**	**35.64±0.27**	**36.54±0.29**	**88.31±0.37**	**47.47**

We treat each dataset as one domain, and perform the unsupervised domain adaptation task using each pair of domains, so in total we have 12 cases. Twenty labeled samples (resp., eight labeled samples) per class are selected randomly as training data when using Amazon, Webcam, and Caltech (resp., DSLR), as the source domain. All the samples are used as unlabeled training data when the dataset is used as the target domain. The test data is as the same as the unlabeled training data as in [1] and [2].

To validate the proposed learning method for multiple source domain adaptation, we additionally use ImageNet [34] and collect another dataset called Google-Image which contains the top ranked 100 images returned from Google-Image search engine by using the class name as the query. We perform the unsupervised multiple source domain adaptation task by leaving one dataset out as the target domain and using the rest datasets as five source domains. Due to the lack of label information on Google-Image dataset, we only use it as the source domain. For ImageNet, 20 labeled images per class are selected as training samples when it is used as one of the source domains. For other datasets, we use the same settings as in the single source domain adaptation task.

For all the settings, we perform 20 rounds of experiments with different randomly selected samples. We empirically set the number of intermediate covariance matrices to $n = 8$ as suggested in [1], and fix the feature dimension as $d = 30$ after using PCA, which results in 300-dim domain-invariant features. For LDA, we set the feature dimension to the class number minus one as in [4]. RBF kernel is used in SVM by setting the bandwidth parameter as the mean distance and we use the default tradeoff parameter (i.e. $C = 1$). In multiple source DA, the tradeoff parameter θ is set to 1. We set the bandwidth parameter δ to 100, when calculating the virtual labels, and we observer the performance is stable when δ is in the ranges of $[10, 1000]$.

B. Results for Single Source Domain Adaptation

We compare our DASC with four baselines. OrigFeat, SGF [1], GFK [2], and information theoretical learning (ITL) [10], where OrigFeat uses the original TF features, SGF and its extended version GFK are the most relevant methods in which the subspaces are exploited, and ITL is a recently proposed unsupervised DA method. Specifically, in SGF [1], PCA is used to compute the subspaces for source

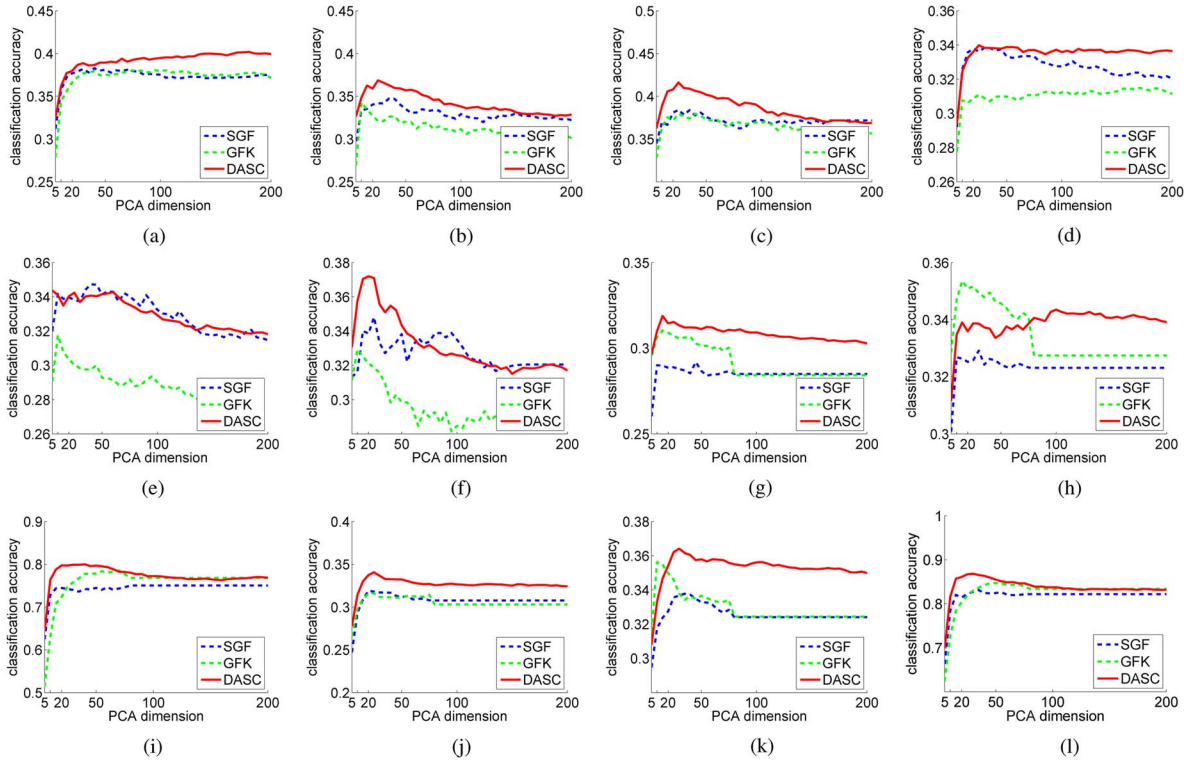


Fig. 5. Mean accuracies of different methods with respect to the feature/subspace dimension d. (a) C→A. (b) C→W. (c) C→D. (d) A→C. (e) A→W. (f) A→D. (g) W→C. (h) W→A. (i) W→D. (j) D→C. (k) D→A. (l) D→W.

and target domains and then partial least squares (PLS) [35] is used to generate the discriminative features after interpolating intermediate subspaces, which is referred as SGF(PLS) here. In GFK [2], PCA is used to generate the subspace for the target domain while PCA and PLS are employed for the source domain, which are referred as GFK (PCA, PCA) and GFK (PLS, PCA) here, respectively. For our DASC, we employ PCA on the intermediate covariance matrices to obtain the projection matrices and then apply LDA to obtain the discriminative features, therefore we refer to it as DASC (PCA+LDA). To fairly compare with SGF and GFK, we additionally report SGF (PCA+LDA) which follows our method to use LDA for extracting the discriminative features and GFK (LDA, PCA) in which LDA is used to obtain the subspace for the source domain. Moreover, we also report the results only using PCA for SGF and our DASC, which are referred as SGF (PCA) and DASC (PCA), respectively.

Following [1], [2], and [10], we report the performances using the NN classifier for the above settings. For ITL, we strictly follow the validation scheme in [10] to select the best parameters. For SGF, we also strictly follow their parameter settings and report their best results by selecting the best subspace dimension d according to the average accuracy over all the 12 cases. For GFK, we also follow their parameter settings, and the subspace dimension d is determined by using the RoD criterion as in [2]. Moreover, we also report the results using the SVM classifier. We do not apply LDA for our DASC and other methods since label information can be exploited in SVM. But we still report the results for SGF and GFK when using PLS for dimension reduction or computing the subspace. Tables I and II summarize the mean accuracies and standard errors for all the 12 cases, and we also report the average accuracy over all cases for each method in Table II (see the last column).

We first consider the NN classifier using PCA only in which we do not use any label information when generating the features. Compared with OrigFeat, SGF (PCA), and GFK (PCA, PCA), our DASC (PCA) achieves the best results in 10 out of 12 cases, which indicates our domain-invariant features can better handle domain difference when compared with them. It also implies that it is better to reduce the data distribution mismatch by directly coping with the covariance matrix rather than using the subspace as in SGF and GFK (see the detailed discussion in Section III-B). We also note that DASC (PCA) generally achieves better results than ITL, which indicates it is better to shift covariance for domain adaptation rather than to use the information theory criterion as in ITL [10].

In general, domain adaptation methods are better than OrigFeat by using either NN or SVM classifier. When compared with other methods, our proposed method DASC (PCA+LDA) (resp. DASC) is the best in 9 (resp. 10) out of the 12 cases when using the NN (resp. SVM) classifier, and is comparable with other methods in the rest cases. In term of the average accuracy, our method is better than the second best result by 2.14% (resp. 1.45%) when using the NN (resp. SVM) classifier.

C. Results for Multiple Source Domain Adaptation

For multiple source unsupervised domain adaptation, we also compare our method with SGF [1] and GFK [2] as well as several state-of-the-art multiple source DA methods including multi-KMM [24], conditional probability-based multi-source

TABLE III
MEAN ACCURACIES (%) FOR UNSUPERVISED MULTIPLE SOURCE DOMAIN ADAPTATION

Method	Caltech	Amazon	Webcam	DSLR	ImageNet
SVM(single best)	40.14	53.04	76.59	69.68	31.21
SVM(fusion)	44.63	50.14	58.47	67.51	33.23
Multi-KMM [24]	47.07	54.22	64.54	70.44	**35.69**
CP-MDA [25]	42.90	47.81	53.35	46.88	30.73
DAM [7]	44.33	49.97	58.47	67.26	32.91
DSM [15]	33.68	42.59	45.25	66.30	27.50
SGF [1]	47.52	55.37	57.41	53.76	33.96
GFK [2]	39.79	50.26	45.00	42.42	26.57
Our method	**49.17**	**57.90**	**84.63**	**80.13**	35.48

domain adaptation (CP-MDA) [25], DAM [7], and DSM [15]. The original TF features are used for [7], [15], [24], and [25] since all these methods are designed for homogenous source domains. We additionally compare two commonly used SVM baselines, SVM (single best) which is the best among all the SVMs trained on each source domain and SVM (fusion) which equally fuses the decision values from all the SVMs trained on different source domains. For SGF and GFK, we strictly follow their approaches in [1] and [2] for multiple source domain adaptation except using the SVM classifier to replace the NN classifier. For these baseline methods, we either use the recommended parameters in their papers or select the best ones according to the test results.

We report the mean accuracies in Table III. SVM (single best) generally performs better than SVM(fusion), which indicates it is necessary to select relevant source domains for multiple source domain adaptation. The proposed method achieves the best performances in four out of five cases. Even for the remaining one our method still achieves the second best result and the performance is also comparable with the best result. These results demonstrate the effectiveness of our domain-invariant features similarly as in the single source DA task. Moreover, the results also demonstrate our proposed learning method can effectively integrate different heterogeneous domain-invariant features and thus it achieves much better performance compared with other multiple source domain adaptation methods.

D. Convergence

We study the convergence of our multiple source domain adaptation algorithm as described in Algorithm 1. Fig. 4 shows the objective values of 20 rounds of experiments with different randomly selected training samples when using Caltech as the target domain. We can observe that our algorithm usually converges after 30∼50 iterations. We also have similar observations in other cases.

E. Covariance Versus Subspace

We have discussed the advantages of using the covariance matrix on a Riemannian manifold when compared with using the subspace on a Grassmann manifold in Section III-B. SGF and GFK cannot bridge the data distribution mismatch in the common subspace of two domains. In contrast, our DASC does not suffer from such a problem because we directly cope with the covariance matrices for domain adaptation without the aid of subspaces. Fig. 5 shows the performance comparison of our DASC with SGF and GFK for the single source domain adaptation task using PCA and NN classifier.[2] Our DASC is generally better than SGF and GFK in most cases when using different d's (i.e., the dimension of subspaces for SGF and GFK and the feature dimension after using PCA for our DASC), which clearly demonstrates the effectiveness of our method.

VI. CONCLUSION

In this paper, we have proposed an effective method to generate domain-invariant features for unsupervised domain adaptation. Different from the existed methods such as SGF and GFK, which adopted the subspace assumption on a Grassmann manifold, we directly use the covariance matrix to represent a domain and construct a geodesic path between the source and target domains on a Riemannian manifold. Then, we extract domain-invariant features by projecting the samples onto the intermediate domains along the geodesic path. For the multiple source domain adaptation task, as the domain-invariant features from each pair of source and target domains may be different, we further propose a new SVM-based approach to simultaneously learn the target classifier as well as the optimal weights for multiple source domains. Extensive experimental results clearly demonstrate the effectiveness of our work.

REFERENCES

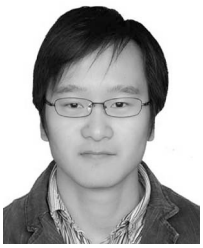
[1] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *Proc. IEEE Int. Conf. Comput. Vision*, 2011, pp. 999–1006.
[2] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 2066–2073.
[3] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Geometric means in a novel vector space structure on symmetric positive-definite matrices," *SIAM J. Matrix Anal. Applicat.*, vol. 29, no. 1, pp. 328–347, 2007.
[4] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.
[5] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Eur. Conf. Comput. Vision*, 2010, pp. 213–226.
[6] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, Berkeley, CA, USA, Tech. Rep. 7694, 2007.
[7] L. Duan, I. Tsang, D. Xu, and T. Chua, "Domain adaptation from multiple sources via auxiliary classifiers," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 3, pp. 504–518, Mar. 2012.
[8] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2011, pp. 1785–1792.
[9] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko, "Discovering latent domains for multisource domain adaptation," in *Proc. Eur. Conf. Comput. Vision*, 2012, pp. 702–715.
[10] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1079–1086.
[11] J. Lim, R. Salakhutdinov, and A. Torralba, "Transfer learning by borrowing examples for multiclass object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 118–126.
[12] S. Si, W. Liu, D. Tao, and K.-P. Chan, "Distribution calibration in riemannian symmetric space," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 4, pp. 921–930, Aug. 2011.
[13] H.-S. Seok, K.-B. Hwang, and B.-T. Zhang, "Feature relevance network-based transfer learning for indoor location estimation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 5, pp. 711–719, Sep. 2011.
[14] L. Duan, D. Xu, I. W. Tsang, and J. Luo, "Visual event recognition in videos by learning from web data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1667–1680, Sep. 2012.

[2]Note that the results in Table I and II are obtained with the fixed parameter d, therefore the performances in this figure could be better by varying d.

[15] L. Duan, D. Xu, and S. Chang, "Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 1338–1345.
[16] L. Duan, I. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 465–479, Mar. 2012.
[17] S. Pan, I. Tsang, J. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.
[18] X. Zhu, "Cross-domain semi-supervised learning using feature formulation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 6, pp. 1627–1638, Dec. 2011.
[19] J. Yang, R. Yan, and A. Hauptmann, "Cross-domain video concept detection using adaptive svms," in *Proc. ACM Int. Conf. Multimedia*, 2007, pp. 188–197.
[20] L. Bruzzone and M. Marconcini, "Domain adaptation problems: A DASVM classification technique and a circular validation strategy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 5, pp. 770–787, May 2010.
[21] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. Empirical Methods Natural Lang. Process.*, 2006, pp. 120–128.
[22] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 601–608.
[23] H. Daumé, III, "Frustratingly easy domain adaptation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2007, pp. 256–263.
[24] G. Schweikert, C. Widmer, B. Schölkopf, and G. Raetsch, "An empirical analysis of domain adaptation algorithms for genomic sequence analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1433–1440.
[25] R. Chattopadhyay, Q. Sun, W. Fan, I. Davidson, S. Panchanathan, and J. Ye, "Multisource domain adaptation and its application to early detection of fatigue," *ACM Trans. Knowl. Discovery Data*, vol. 6, no. 4, p. 18, 2012.
[26] H. Daumé, III, and D. Marcu, "Domain adaptation for statistical classifiers," *J. Artif. Intell. Res.*, vol. 26, no. 1, pp. 101–126, 2006.
[27] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "Learning bounds for domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 129–136.
[28] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19. 2007, p. 137.
[29] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1, pp. 151–175, 2010.
[30] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[31] K. A. Gallivan, A. Srivastava, X. Liu, and P. Van Dooren, "Efficient algorithms for inferences on grassmann manifolds," in *Proc. IEEE Stat. Signal Process. Workshop*, 2003, pp. 315–318.
[32] C. Chang and C. Lin, "LibSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[33] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vision*, 2006, pp. 404–417.
[34] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2009, pp. 248–255.
[35] P. Geladi and B. Kowalski, "Partial least-squares regression: A tutorial," *Analytica chimica acta*, vol. 185, no. 1, pp. 1–17, 1986.
[36] L. Chen, L. Duan, and D. Xu, "Event recognition in videos by learning from heterogeneous web sources," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2013, pp. 2666–2673.

Zhen Cui (S'13) received the B.S. degree from Shandong Normal University, Jinan, China, in 2004, and then the M.S. degree from Sun Yat-sen University, Guangzhou, China, 2006, and is currently pursuing the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China.

He is currently a Lecturer with Huaqiao University, Xiamen, China. His current research interests include pattern recognition and computer vision, especially face recognition and image super resolution based on recently emerged theories, such as sparse coding, manifold learning, and deep learning.

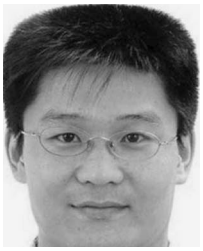


Wen Li (S'12) received the B.S. and M.Eng degrees from the Beijing Normal University, Beijing, China, in 2007 and 2010, respectively and is currently pursuing the Ph.D. Degree with the School of Computer Engineering, Nanyang Technological University, Singapore. His current research interests include ambiguous learning, domain adaptation, and multiple kernel learning.

Dong Xu (M'07–SM'13) received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2001 and 2005, respectively.

While pursuing the Ph.D., he was with Microsoft Research Asia, Beijing, China, and the Chinese University of Hong Kong, Shatin, Hong Kong, for over two years. He was a Post-Doctoral Research Scientist with Columbia University, New York, NY, USA, for one year. He is currently an Associate Professor with Nanyang Technological University, Singapore. His current research interests include computer vision, statistical learning, and multimedia content analysis.

Dr. Xu was the co-author of a paper that won the Best Student Paper Award in the IEEE International Conference on Computer Vision and Pattern Recognition in 2010.



Shiguang Shan (M'04) received the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 2004.

He has been with ICT, CAS since 2002 and has been a Professor since 2010. He is also the Vice Director of the Key Laboratory of Intelligent Information Processing of CAS. His current research interests include image analysis, pattern recognition, and computer vision. He is focusing especially on face recognition related research topics.

Dr. Shan has received the China's State Scientific and Technological Progress Awards in 2005 for his work on face recognition technologies.

Xilin Chen (M'00–SM'09) received the B.S., M.S., and Ph.D. degrees in computer science from the Harbin Institute of Technology (HIT), Harbin, China, in 1988, 1991, and 1994, respectively.

He was a Professor with the HIT from 1999 to 2005 and was a Visiting Scholar with Carnegie Mellon University, Pittsburgh, PA, USA, from 2001 to 2004. He has been a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, since 2004. His current research interests include image processing, pattern recognition, computer vision, and multimodal interface.

Dr. Chen was the recipient of several awards, including the China's State Scientific and Technological Progress Award in 2000, 2003, 2005, and 2012 for his research work.

Xuelong Li (M'02–SM'07–F'12) is currently a Full Professor with the Center for Optical Imagery Analysis and Learning, State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.