Multi-Kernel Coupled Projections for Domain Adaptive Dictionary Learning

Yuhui Zheng, Xilong Wang, Guoqing Zhang, Baihua Xiao, Fu Xiao, *Member, IEEE*, and Jianwei Zhang

***Abstract*—Dictionary learning has produced state-of-the-art results in various classification tasks. However, if the training data have a different distribution than the testing data, the learned sparse representation might not be optimal. Recently, several domain-adaptive dictionary learning (DADL) methods and kernels have been proposed and have achieved impressive performance. However, the performance of these single kernel-based methods heavily depends heavily on the choice of the kernel, and the question of how to combine multiple kernel learning (MKL) with the DADL framework has not been well studied. Motivated by these concerns, in this paper, we propose a multi-kernel domain-adaptive sparse representation-based classification (MK-DASRC) and then use it as a criterion to design a multi-kernel sparse representation-based domain-adaptive discriminative projection method, in which the discriminative features of the data in the two domains are simultaneously learned with the dictionary. The purpose of this method is to maximize the between-class sparse reconstruction residuals of data from both domains, and minimize the within-class sparse reconstruction residuals of data in the low-dimensional subspace. Thus, the resulting representations can satisfactorily fit MK-DASRC and simultaneously display discriminability. Extensive experimental results on a series of benchmark databases show that our method performs better than the state-of-the-art methods.**

***Index Terms*—Dictionary learning, multiple kernel learning, discriminative projections, domain adaptation.**

Manuscript received August 13, 2018; revised December 17, 2018 and January 24, 2019; accepted January 25, 2019. Date of publication February 18, 2019; date of current version August 23, 2019. This work was support in part by the Natural Science Foundation of China undere Grants 61806099 and 61672293, in part by the Natural Science Foundation of Jiangsu Province, China under Grant BK20180790, in part by the Talent Start Foundation of Nanjing University of Information Science and Technology (2243141701077), and in part by the PAPD (a project funded by the priority academic program development of Jiangsu Higher Education Institutions). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Zixiang Xiong. *(Corresponding author: Guoqing Zhang.)*

Y. Zheng, X. Wang and G. Zhang are with the School of Computer and Software, Nanjing University of Information Science and Technology, Jiangsu 210044, China (e-mail: zheng_yuhui@nuist.edu.cn; wangxilong1150@gmail.com; xiayang14551@163.com).

B. Xiao is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: baihua.xiao@ia.ac.cn).

F. Xiao is with the School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: xiaof@njupt.edu.cn).

J. Zhang is with the School of Mathematics and Statistics, Nanjing University of Information Science and Technology, Jiangsu 210044, China (e-mail: zhangjw@nuist.edu.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/TMM.2019.2900166

Fig. 1. Selected images from the back-pack category; Amazon (left two columns), DLSR (right two columns).

I. Introduction

With the rapid development of computer and internet technology, large amounts of media data are generated every day from social media sites such as Google, Baidu, Weibo, Facebook, etc. These multimedia data increase greatly in cross-domain scenes, i.e., different websites produce media data in different domains. Fig. 1 illustrates selected samples of a back-pack from two different domains. Although these samples have same object class label, they are visually dissimilar. If we use source domain images as the training data to learn a classifier, and subsequently test the target images, no matter how specific the cause is, any change in distribution after classifier learning will degrade its performance at test time. Domain adaptation (DA) attempts to tackle the problem in which the source domain data used to learn a model have a distribution different from that of the data on which the model is applied [1]–[11]. This problem is often encountered in practical applications. In addition, collection of sufficient labeled target images is expensive and time-consuming. If we use limited labeled target data to train the classifiers, the learned classifiers are not usually robust for vision recognition tasks. DA is able to learn robust classifiers with only a few labeled samples from the target domain by exploiting many labeled samples from other source domains [12]–[16], [56].

Different types of domain-adaptive methods have been developed in recent years [17]–[25], and sparse representation and dictionary-based methods have achieved highly competitive performance [26]–[32] because of the robust discriminant representation that they supply by adapting to a particular data sample. Shekhar *et al.* [29], [30] proposed a generalized DADL

approach to jointly learn the projections of data in the source and target domains. To further improve discrimination, based on the classification criteria of sparse representation-based classification (SRC), Zhang *et al.* [32] presented an optimal couple of projections for a domain-adaptive SRC method such that SRC can achieve better performance in DA recognition problem.

Recently, deep neural networks have had excellent success in cross-domain recognition tasks and achieved observable improvement [13], [33]–[36], partly because deep networks can learn highly powerful hierarchical nonlinear representations of the inputs [37]–[39], [59]–[65], making them suitable for domain adaptation. Bousmalis *et al.* [38] presented an unsupervised domain adaptation method that learns a transformation in the pixel space using generative adversarial networks. Herath *et al.* [39] addressed both unsupervised and semi-supervised domain adaptation problems. Although deep neural networks have exhibited promising performance, certain limitations remain (e.g., requirements of a large amount of additional training samples, the collection of which is difficult, and supercomputational machines). As a result, in this paper, we focus only on the situation in which a large number of additional training samples is not used.

In many real-world applications, samples usually lie in a nonlinear feature space, and kernel technology is an effective way to tackle nonlinear data structures [40]–[42]. Instead of directly using a fixed kernel, multiple kernel learning (MKL) [43]–[45] has been widely applied to learn an optimal kernel, which is a linear combination of multiple predetermined kernel functions. Recently, several multi-kernel sparse representation or dictionary learning approaches have been proposed [46]–[48]. Nevertheless, these methods usually assume that the test data and training data originate from the same domain. Thus, MKL algorithms are invalid for obtaining the optimal kernel with a combination of data for the DA problem. Therefore, the performance of MKL algorithms trained in the source domain are degraded in the target domain [49], [50].

Based on the above motivations, we present a multi-kernel domain-adaptive sparse representation-based classification method (MK-DASRC), and then based on the decision criteria of the MK-DASRC, we propose a multi-kernel domain-adaptive based discriminative projection method (MK-DADP), which jointly learns the transformation of data in different domains, and a discriminative dictionary in a common space. The proposed method aims to learn the coupled projections and a common dictionary such that in the transformed low-dimensional space, the within-class sparse reconstruction residuals of data are minimized and the between-class sparse reconstruction residuals are maximized. This approach encourages the sparse codes to be discriminative across classes. As a result, MK-DASRC is able to obtain better recognition accuracy in the projected space. Moreover, the proposed method is easily extensible for addressing multiple domains.

The remainder of the paper is organized as follows. Section II presents our multiple kernel domain-adaptive sparse representation-based classification algorithm. The proposed multiple kernel discriminative projection method is described in Section III, and the optimization algorithm is described in Section IV. Experimental results are reviewed in Section V, and Section VI summarizes our conclusions.

II. Multi-Kernel Domain Adaptive Sparse Representation-Based Classification

To achieve better results in object recognition tasks, combining multiple feature representations is important. The search for better feature combinations requires proper design of kernel functions among a set of candidate kernels. MKL aims to determine the mixing weights of multiple kernels [46], [51], assuming that $\{\kappa_m\}_{m=1}^{M}$ represents a group of base kernel functions ($\{\mathbf{K}_m\}_{m=1}^{M}$ are the kernel matrices), where M is the number of base kernels. The weighted kernel can be computed as follows

$$\kappa_\beta = \sum_{m=1}^{M} \beta_m \kappa_m, \mathbf{K}_{\boldsymbol{\beta}} = \sum_{m=1}^{M} \beta_m \mathbf{K}_m, \tag{1}$$

where $\beta \geq 0$ is the kernel weight. MKL aims to learn the optimal kernel weights β_m.

Let $\mathbf{Y_1} = [\mathbf{y_1^1}, \mathbf{y_2^1}, \ldots, \mathbf{y_{N_1}^1}] \in R^{m_1 \times N_1}$ and $\mathbf{Y_2} = [\mathbf{y_1^2}, \mathbf{y_2^2}, \ldots, \mathbf{y_{N_2}^2}] \in R^{m_2 \times N_2}$ denote the source and target domain data, respectively. In practice, samples usually lie in nonlinear feature space, and thus, the linear classifier is not able to satisfactorily characterize the corresponding feature space. The kernel method is an effective approach to address this problem. In this work, ϕ is a nonlinear mapping associated with the kernel function $\kappa(\mathbf{y}_i, \mathbf{y}_j) = \phi(\mathbf{y}_i)^T \phi(\mathbf{y}_i)$. Therefore $\mathbf{Y_1}$ and $\mathbf{Y_2}$ in space $\mathcal{F}$ can be separately expressed as $\phi(\mathbf{Y_1})$ and $\phi(\mathbf{Y_2})$. For a testing sample $\mathbf{y}_{te}$, from domain $\mathbf{i}$, $i = 1, 2$, the sparse representation in the space $\mathcal{F}$ can be expressed as shown

$$\arg\min_{\boldsymbol{\alpha}} = \|\phi(\mathbf{y}_{te}) - \widetilde{\mathbf{D}}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1, \tag{2}$$

where $\|\cdot\|_1$ is the ℓ_1-norm, $\widetilde{\mathbf{D}} \in R^{\widetilde{m} \times K}$ denotes the dictionary in feature space $\mathcal{F}$, K is the number of atoms in the dictionary, and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_K]^T$ is the representation vector associated with sample $\phi(\mathbf{y}_{te})$. Since the dimension of feature space $\mathcal{F}$ is quite high or possibly infinite, dimensionality reduction is necessary in $\mathcal{F}$. As such, $\mathbf{P_i} \in R^{\widetilde{m} \times d}$ denotes as the projection, which is no longer linear from space $\mathcal{F}$ to the reduced space, where d is the dimension of the projection space. Eq. (2) can be rewritten as follows:

$$\arg\min_{\boldsymbol{\alpha}} = \|\mathbf{P_i}^T \phi(\mathbf{y}_{te}) - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1, \tag{3}$$

where $\mathbf{D} = \mathbf{P}^T\widetilde{\mathbf{D}} = [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_c]$ is the learned dictionary in the common space, $\mathbf{P} = [\mathbf{P_1}; \mathbf{P_2}]$, c is the total number of classes, and $K = \sum_{i=1}^{c} K_i$, where K_i is the dictionary atoms in each sub-dictionary i. Because any element in $\mathcal{F}$ lies in the span of the transformed training samples, we can represent the projection as $\mathbf{P_i} = \phi(\mathbf{Y_i})\mathbf{A_i}$, where $\mathbf{A_i} \in R^{n_i \times d}$ is the coefficient matrix for $\mathbf{P_i}$. Thus, we obtain the following:

$$\mathbf{P_i}^T \phi(\mathbf{y}_{te}) = \mathbf{A_i}^T \phi(\mathbf{Y_i})^T \phi(\mathbf{y}_{te}) = \mathbf{A_i}^T \mathbb{K}_{\mathbf{i}}^{(\mathbf{y})} \boldsymbol{\beta} \tag{4}$$

where

$$\mathbb{K}_{\mathbf{i}}^{(\mathbf{y})} = \begin{bmatrix} \mathbf{K_1}(\mathbf{y}_{\mathbf{1}}^{\mathbf{i}}, \mathbf{y}) & \cdots & \mathbf{K}_M(\mathbf{y}_{\mathbf{1}}^{\mathbf{i}}, \mathbf{y}) \\ \vdots & \ddots & \vdots \\ \mathbf{K}_1(\mathbf{y}_{N_i}^{\mathbf{i}}, \mathbf{y}) & \cdots & \mathbf{K}_M(\mathbf{y}_{N_i}^{\mathbf{i}}, \mathbf{y}) \end{bmatrix} \in R^{N_i \times M}, \quad (5)$$

and $\mathbf{K}_m(i,j) = \kappa_m(\mathbf{y}_i, \mathbf{y}_j)$.

We can represent the projection $\mathbf{P}$ as $\mathbf{P} = \phi(\mathbf{Y})\mathbf{A}$, where $\phi(\mathbf{Y}) = [\begin{smallmatrix} \phi(\mathbf{Y_1}) & \mathbf{0} \\ \mathbf{0} & \phi(\mathbf{Y_2}) \end{smallmatrix}]$, and $\mathbf{A} = [\mathbf{A_1}; \mathbf{A_2}]$. Dictionary $\widetilde{\mathbf{D}}$ can be computed as $\widetilde{\mathbf{D}} = \phi(\mathbf{Y})\mathbf{B}$, where $\mathbf{B} \in R^{\sum N_i \times K}$. Thus Eq. (3) can be written as follows

$$\arg\min_{\boldsymbol{\alpha}} \|\mathbf{A}_{\mathbf{i}}^T \mathbb{K}_{\mathbf{i}}^{(\mathbf{y})} \boldsymbol{\beta} - \mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B} \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1, \quad (6)$$

where $\mathbf{K}_{\boldsymbol{\beta}} = \phi(\mathbf{Y})^T \phi(\mathbf{Y}) = \sum_{m=1}^{M} \beta_m \mathbf{K}_m$ is the kernel Gram matrix, and $\mathbf{A}^T = [\mathbf{A}_{\mathbf{1}}^T, \mathbf{A}_{\mathbf{2}}^T]$. Once we obtain the coefficient vector $\boldsymbol{\alpha}$, the reconstruction residual can be computed as shown

$$e_i = \|\mathbf{A}_{\mathbf{i}}^T \mathbb{K}_{\mathbf{i}}^{(\mathbf{y})} \boldsymbol{\beta} - \mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B} \delta_i(\boldsymbol{\alpha})\|_2^2, \quad (7)$$

where $\delta_i(\cdot)$ is a characteristic function that chooses the coefficients with respect to class i, $i = 1, 2, \ldots, c$. We use $identity(\mathbf{y}_{te}) = \arg\min_i \{e_i\}$ to determine the label of $\mathbf{y}_{te}$.

III. Multi-Kernel Domain Adaptive Discriminative Projections

A. Proposed Framework

A multi-kernel domain-adaptive based discriminative projections method (MK-DADP) is proposed in this section. Based on the classification criteria of our proposed MK-DASRC, we learn the coupled projections of data from both domains and simultaneously learn a latent dictionary. Furthermore, we note that MK-DADP and MK-DASRC depend on each other, and the sparse coefficient vectors required in MK-DADP must be computed by MK-DASRC. The projections, kernel weights and structured dictionary required in MK-DASRC must be computed by MK-DADP. We use an alternative iterative algorithm to optimize the proposed method.

In the source domain, given a training sample $\phi(\mathbf{y}_{i,j}^{\mathbf{1}})$, where $\mathbf{y}_{i,j}^{\mathbf{1}}$ represents the j-th sample of class i, we compute its sparse representation vector $\boldsymbol{\alpha}_{i,j}^{\mathbf{1}}$, and then define the source domain within-class sparse reconstruction residual in the projected space $J_w^{\mathbf{1}}$ as follows:

$$\begin{aligned} J_w^{\mathbf{1}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{1}}} (\mathbf{P_1}^T \phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})) \right. \\ &\quad \left. \times \left(\mathbf{P_1}^T \phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})\right)^T \right) \\ &= tr\left((\mathbf{P_1}^T \phi(\mathbf{Y_1}) - \mathbf{D}\boldsymbol{\Lambda}_w^{\mathbf{1}})(\mathbf{P_1}^T \phi(\mathbf{Y_1}) - \mathbf{D}\boldsymbol{\Lambda}_w^{\mathbf{1}})^T\right) \\ &= \|\mathbf{P_1}^T \phi(\mathbf{Y_1}) - \mathbf{D}\boldsymbol{\Lambda}_w^{\mathbf{1}}\|_F^2, \end{aligned} \quad (8)$$

where $\boldsymbol{\Lambda}_w^{\mathbf{1}} = [\delta_1(\boldsymbol{\alpha}_{1,1}^{\mathbf{1}}), \delta_1(\boldsymbol{\alpha}_{1,2}^{\mathbf{1}}), \ldots, \delta_c(\boldsymbol{\alpha}_{c,n_i^{\mathbf{1}}}^{\mathbf{1}})] \in R^{K \times N_1}$, $n_i^{\mathbf{1}}$ is the training samples in class i, and $N_1 = \sum_{i=1}^{c} n_i^{\mathbf{1}}$. $\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})$ is a vector whose only nonzero coefficients are the entries in $\boldsymbol{\alpha}_{i,j}^{\mathbf{1}}$ corresponding to the i-th class.

The source domain between-class sparse reconstruction residual $J_b^{\mathbf{1}}$ can be defined as follows:

$$\begin{aligned} J_b^{\mathbf{1}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{1}}}\sum_{s\neq i} (\mathbf{P_1}^T \phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})) \right. \\ &\quad \left. \times \left(\mathbf{P_1}^T \phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})\right)^T \right) \\ &= \|\mathbf{P_1}^T \phi(\mathbf{Y_1}) - \mathbf{D}\boldsymbol{\Lambda}_b^{\mathbf{1}}\|_F^2, \end{aligned} \quad (9)$$

where $\boldsymbol{\Lambda}_b^{\mathbf{1}} = [\delta_s(\boldsymbol{\alpha}_{1,1}^{\mathbf{1}}), \delta_s(\boldsymbol{\alpha}_{1,2}^{\mathbf{1}}), \ldots, \delta_s(\boldsymbol{\alpha}_{c,n_i^{\mathbf{1}}}^{\mathbf{1}})] \in R^{K \times N_1}$, $\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})$ is a vector whose only nonzero coefficients are the entries in $\boldsymbol{\alpha}_{i,j}^{\mathbf{1}}$ corresponding to the s-th class, and $s \neq i$.

In the target domain, we define the within-class and between-class residuals in the reduced space as shown

$$\begin{aligned} J_w^{\mathbf{2}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{2}}} (\mathbf{P_2}^T \phi(\mathbf{y}_{i,j}^{\mathbf{2}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{2}})) \right. \\ &\quad \left. \times (\mathbf{P_2}^T \phi(\mathbf{y}_{i,j}^{\mathbf{2}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{2}}))^T \right) \\ &= \|\mathbf{P_2}^T \phi(\mathbf{Y_2}) - \mathbf{D}\boldsymbol{\Lambda}_w^{\mathbf{2}}\|_F^2, \end{aligned} \quad (10)$$

and

$$\begin{aligned} J_b^{\mathbf{2}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{2}}}\sum_{s\neq i} (\mathbf{P_2}^T \phi(\mathbf{y}_{i,j}^{\mathbf{2}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{2}})) \right. \\ &\quad \left. \times (\mathbf{P_2}^T \phi(\mathbf{y}_{i,j}^{\mathbf{2}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{2}}))^T \right) \\ &= \|\mathbf{P_2}^T \phi(\mathbf{Y_2}) - \mathbf{D}\boldsymbol{\Lambda}_b^{\mathbf{2}}\|_F^2, \end{aligned} \quad (11)$$

where $\boldsymbol{\Lambda}_w^{\mathbf{2}} = [\delta_1(\boldsymbol{\alpha}_{1,1}^{\mathbf{2}}), \delta_1(\boldsymbol{\alpha}_{1,2}^{\mathbf{2}}), \ldots, \delta_c(\boldsymbol{\alpha}_{c,n_i^{\mathbf{2}}}^{\mathbf{2}})] \in R^{K \times N_2}$ and $\boldsymbol{\Lambda}_b^{\mathbf{2}} = [\delta_s(\boldsymbol{\alpha}_{1,1}^{\mathbf{2}}), \delta_s(\boldsymbol{\alpha}_{1,2}^{\mathbf{2}}), \ldots, \delta_s(\boldsymbol{\alpha}_{c,n_i^{\mathbf{2}}}^{\mathbf{2}})] \in R^{K \times N_2}$, $n_i^{\mathbf{2}}$ is the training samples in the i-th class, and $N_2 = \sum_{i=1}^{c} n_i^{\mathbf{2}}$.

We expect to simultaneously maximize $J_b^{\mathbf{1}}$ and $J_b^{\mathbf{2}}$ from both domains in the reduced space.

$$\begin{aligned} \max J_b &= \max\{J_b^{\mathbf{1}} + J_b^{\mathbf{2}}\} \\ &= \max\{\|\mathbf{P_1}^T \phi(\mathbf{Y_1}) - \mathbf{D}\boldsymbol{\Lambda}_b^{\mathbf{1}}\|_F^2 \\ &\quad + \|\mathbf{P_2}^T \phi(\mathbf{Y_2}) - \mathbf{D}\boldsymbol{\Lambda}_b^{\mathbf{2}}\|_F^2\} \\ &= \max \|\mathbf{P}^T \phi(\mathbf{Y}) - \mathbf{D}\boldsymbol{\Lambda}_b\|_F^2 \\ &= \max \|\mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B} \boldsymbol{\Lambda}_b\|_F^2, \end{aligned} \quad (12)$$

and simultaneously minimize the $J_w^{\mathbf{1}}$ and $J_w^{\mathbf{2}}$ residuals

$$\begin{aligned} \min J_w &= \min\{J_w^{\mathbf{1}} + J_w^{\mathbf{2}}\} \\ &= \max \|\mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B} \boldsymbol{\Lambda}_w\|_F^2, \end{aligned} \quad (13)$$

where $\mathbf{\Lambda}_b = [\mathbf{\Lambda}_b^{\mathbf{1}}, \mathbf{\Lambda}_b^{\mathbf{2}}]$ and $\mathbf{\Lambda}_w = [\mathbf{\Lambda}_w^{\mathbf{1}}, \mathbf{\Lambda}_w^{\mathbf{2}}]$. We can learn $\mathbf{A}$ and $\mathbf{B}$ by maximizing the following objective function

$$J(\mathbf{A},\mathbf{B}) = \max_{\mathbf{A},\mathbf{B}} \frac{tr((\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}\mathbf{\Lambda}_b)(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}\mathbf{\Lambda}_b)^T)}{tr((\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}\mathbf{\Lambda}_w)(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}} - \mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}\mathbf{\Lambda}_w)^T)} = \max_{\mathbf{A},\mathbf{B}} \frac{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}$$

$$\text{s.t.} \quad \mathbf{A}_{\mathbf{i}}^T\mathbf{K}_{\mathbf{i}}\mathbf{A}_{\mathbf{i}} = \mathbf{I}, \forall i = 1,2, \sum_{i=1}^{M}\beta_m = 1, \beta_m \geq 0. \tag{14}$$

where $\mathbf{K}_{\mathbf{i}} = \phi(\mathbf{Y}_{\mathbf{i}})^T\phi(\mathbf{Y}_{\mathbf{i}})$, $\mathbf{S}_b = (\mathbf{I} - \mathbf{B}\mathbf{\Lambda}_b)(\mathbf{I} - \mathbf{B}\mathbf{\Lambda}_b)^T$ is the between-class scatter matrix and $\mathbf{S}_w = (\mathbf{I} - \mathbf{B}\mathbf{\Lambda}_w)(\mathbf{I} - \mathbf{B}\mathbf{\Lambda}_w)^T$ is the within-class scatter matrix. In this work, we require that $\mathbf{P_1}$ and $\mathbf{P_2}$ are orthogonal. Thus, the equality constraint becomes $\mathbf{P}_{\mathbf{i}}^T\mathbf{P}_{\mathbf{i}} = \mathbf{A}_{\mathbf{i}}^T\mathbf{K}_{\mathbf{i}}\mathbf{A}_{\mathbf{i}} = \mathbf{I}$.

B. Multiple Domains

For the multiple domain problem, we construct matrices $\mathbf{P}$, $\phi(\mathbf{Y})$, $\mathbf{\Lambda}_b$, $\mathbf{\Lambda}_w$ as $\mathbf{P}^T = [\mathbf{P}_{\mathbf{1}}^T, \ldots, \mathbf{P}_{\mathbf{M}}^T]$,

$$\phi(\mathbf{Y}) = \begin{bmatrix} \phi(\mathbf{Y_1}) & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \phi(\mathbf{Y_M}) \end{bmatrix},$$

$\mathbf{\Lambda}_b = [\mathbf{\Lambda}_b^{\mathbf{1}}, \ldots, \mathbf{\Lambda}_b^{\mathbf{M}}]$, and $\mathbf{\Lambda}_w = [\mathbf{\Lambda}_w^{\mathbf{1}}, \ldots, \mathbf{\Lambda}_w^{\mathbf{M}}]$.

Using the above definitions, we extend Eq. (14) to multiple domains as follows

$$J(\mathbf{A},\mathbf{B}) = \max_{\mathbf{A},\mathbf{B}} \frac{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}$$

$$\text{s.t.} \quad \mathbf{A}_{\mathbf{i}}^T\mathbf{K}_{\mathbf{i}}\mathbf{A}_{\mathbf{i}} = \mathbf{I}, \forall i = 1, \ldots, M, \sum_{i=1}^{M}\beta_m = 1, \beta_m \geq 0. \tag{15}$$

IV. Optimization

The objective function in Eq. (15) is generally non-convex with respect to $\mathbf{A}$, $\mathbf{B}$ and β. We develop an iterative algorithm to optimize the variables alternatively.

Step 1: Learn $\boldsymbol{A}$, $\boldsymbol{\beta}$ with fixed $\mathbf{B}$. For a fixed $\mathbf{B}$, using the trace ratio optimization method, we can solve the objective function.

To avoiding overfitting, we should ensure that $\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T + \mu\mathbf{I}$ is of full rank. Thus, in the denominator, we add regularization term $\mu\mathbf{I}$.

We know that for certain $\mathbf{A}^*$ and $\boldsymbol{\beta}^*$, there is a maximum ρ^* that can reach them. Thus, for any $\mathbf{A}$ and $\boldsymbol{\beta}$, we have the following

$$\frac{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}{tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T + \mu\mathbf{I})\mathbf{A}\right)} \leq \rho^*, \tag{16}$$

and hence,

$$tr\left(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A}\right) - \rho^* tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T + \mu\mathbf{I})\mathbf{A}\right) \leq 0. \tag{17}$$

To optimize the objective function, we define a function

$$\begin{aligned} f(\rho) &= \max_{\mathbf{A}} G(\mathbf{A},\rho) \\ &= \max_{\mathbf{A}} tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T - \rho^*\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mu\mathbf{I})\mathbf{A}\right) \\ &= \max_{\mathbf{A}} tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}(\mathbf{S}_b - \rho\mathbf{S}_w)\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mu\mathbf{I})\mathbf{A}\right). \end{aligned} \tag{18}$$

In this work, $f(\rho)$ has the following properties, which are proved in [32].

(i). $f(\rho)$ is a decreasing function. (ii). $f(\rho) = 0 \; iff \; \rho = \rho^*$.

Following [32], we known that ρ^* always exists and projection $\mathbf{A}$, weight $\boldsymbol{\beta}$ and the root of the $f(\rho)$ can be found by updating them alternately. Giving a ρ, we denote $\boldsymbol{\beta}(\rho)$ and $\mathbf{A}(\rho)$ as the solution of Eq. (18). Thus

$$f'(\rho) = -tr\left(\mathbf{A}(\rho)^T(\mathbf{K}_\beta(\rho)\mathbf{S}_w\mathbf{K}_\beta^T(\rho) + \mu\mathbf{I})\mathbf{A}(\rho)\right) \tag{19}$$

where $\mathbf{K}_\beta(\rho) = \sum_{m=1}^{M}\boldsymbol{\beta}_m(\rho)\mathbf{K}_m$. With $\boldsymbol{\beta}(\rho)$ and $\mathbf{A}(\rho)$, the root can be found by $\rho_{new} = \rho - \eta_1\frac{f(\rho)}{f'(\rho)}$, where η_1 is the step length. In the following, we will introduce how to find $\boldsymbol{\beta}(\rho)$ and $\mathbf{A}(\rho)$ for a given ρ.

Using constraint $\mathbf{A}_{\mathbf{i}}^T\mathbf{K}_{\mathbf{i}}\mathbf{A}_{\mathbf{i}} = \mathbf{I}$, $\mathbf{A}$ can be solved by the following

$$\max_{\mathbf{A}} tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}(\mathbf{S}_b - \rho\mathbf{S}_w)\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mu\mathbf{I})\mathbf{A}\right).$$

$$\text{s.t.} \quad \mathbf{A}_{\mathbf{i}}^T\mathbf{K}_{\mathbf{i}}\mathbf{A}_{\mathbf{i}} = \mathbf{I}, \forall i = 1, \ldots, M, \sum_{i=1}^{M}\beta_m = 1, \beta_m \geq 0. \tag{20}$$

Because it is difficult to direct the optimization, i.e., Eq. (20), thus, we use a two-step, iterative strategy to optimize $\mathbf{A}$ and $\boldsymbol{\beta}$.

Updating: $\mathbf{A}$: If $\boldsymbol{\beta}$ is fixed, Eq. (20) can be expressed as shown

$$\max_{\mathbf{G}} tr(\mathbf{G}^T\mathbf{H}\mathbf{G}) \quad \text{s.t.} \quad \mathbf{G}_{\mathbf{i}}^T\mathbf{G}_{\mathbf{i}} = \mathbf{I}, \forall i = 1, 2, \ldots, M, \tag{21}$$

where $\mathbf{H} = \mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{V}^T(\mathbf{K}_\beta(\mathbf{S}_b - \rho\mathbf{S}_w)\mathbf{K}_\beta^T - \rho\mu\mathbf{I})\mathbf{V}\mathbf{\Lambda}^{-\frac{1}{2}}$.

Proof: Let $\mathbf{G} = \mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{V}^T\mathbf{A}$, where $\mathbf{V}$ and $\mathbf{\Lambda}$ denote as the eigen decomposition of $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$. Substituting $\mathbf{H}$ and $\mathbf{G}$ into Eq.(21), we can obtain the optimization problem in Eq.(20).

Similar to [30] and [32], Eq. (21) can be solved efficiently using the algorithm presented by [52].

Updating: $\boldsymbol{\beta}$: Fix $\mathbf{A}$, and solve the following optimization problem to find $\boldsymbol{\beta}$

$$\max_{\boldsymbol{\beta}} tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mu\mathbf{I})\mathbf{A}\right)$$

$$\text{s.t.} \quad \sum_{m=1}^{M}\beta_m = 1, \beta_m \geq 0. \tag{22}$$

Eq. (22) is a non-convex problem, and we define the following

$$h(\boldsymbol{\beta}) = tr\left(\mathbf{A}^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T - \rho\mu\mathbf{I})\mathbf{A}\right) \tag{23}$$

and we obtain

$$\frac{\partial h}{\partial \beta_m} = tr\left(\mathbf{A}^T(\mathbf{K}_m(\mathbf{S}_b - \mathbf{S}_w)\mathbf{K}_{\boldsymbol{\beta}} + \mathbf{K}_{\boldsymbol{\beta}}(\mathbf{S}_b - \mathbf{S}_w)\mathbf{K}_m\mathbf{A}\right) \tag{24}$$

To find $\boldsymbol{\beta}$, we update the projection of $\boldsymbol{\beta}$ in the direction of $\frac{\partial h}{\partial \beta_m}$. As mentioned in [45], to satisfy the constraint on $\boldsymbol{\beta}$, the projection $\mathbf{z}$ of $\boldsymbol{\beta}$ on the hyperplane $\boldsymbol{\beta}^T\mathbf{1} = 1$ is defined as shown

$$\pi(\boldsymbol{\beta}) = \arg\min_{\mathbf{z}^T\mathbf{1}=1, \mathbf{z}\geq 0} \|\mathbf{z} - \boldsymbol{\beta}\|_2^2 \tag{25}$$

which is a quadratic programming (QP) problem. In every iteration, we move $\boldsymbol{\beta}$ on the hyperplane and obtain a local solution for $\boldsymbol{\beta}$ and $\mathbf{A}$. Algorithm 2 describes the steps for finding $\mathbf{A}$ and $\boldsymbol{\beta}$.

Step 2: Learn $\boldsymbol{B}$ with fixed $\mathbf{A}$ *and* $\boldsymbol{\beta}$. For fixed $\mathbf{A}$, $\boldsymbol{\beta}$, Eq. (15) can be written as follows:

$$J(\mathbf{B}) = \max_{\mathbf{B}} \frac{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_b\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}{tr(\mathbf{A}^T\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}_w\mathbf{K}_{\boldsymbol{\beta}}^T\mathbf{A})}. \tag{26}$$

To learn the dictionary, the sub-dictionaries $\mathbf{D}_i$ and $\mathbf{D}_s$, $s \neq i$, can be represented as $\mathbf{D}_i = \mathbf{P}^T\phi(\mathbf{Y})\mathbf{B}_i$ and $\mathbf{D}_s = \mathbf{P}^T\phi(\mathbf{Y})\mathbf{B}_s$, respectively, where $\mathbf{B} = [\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_c]$. Our objective function in Eq. (26) can be rewritten as follows:

$$J(\mathbf{B}_i) = \max_{\mathbf{B}_i} \sum_{i=1}^{c} \frac{tr(\mathbf{A}^T\mathbf{S}_b^i\mathbf{A})}{tr(\mathbf{A}^T\mathbf{S}_w\mathbf{A})}, \tag{27}$$

where $\mathbf{S}_b^i = \sum_{s\neq i}(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}_s\boldsymbol{\Lambda}_b^s)(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}_s\boldsymbol{\Lambda}_b^s)^T$, $\mathbf{S}_w = \sum_{i=1}^c \mathbf{S}_w^i = \sum_{i=1}^c(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}_i\boldsymbol{\Lambda}_b^i)(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}_i\boldsymbol{\Lambda}_b^i)^T$ and $\mathbf{K}_{\boldsymbol{\beta}}^i = \phi(\mathbf{Y})^T\phi(\mathbf{Y}_i)$. Additionally, $\boldsymbol{\Lambda}_b^s = [\boldsymbol{\Lambda}_b^{\mathbf{1},s}, \boldsymbol{\Lambda}_b^{\mathbf{2},s} \cdots, \boldsymbol{\Lambda}_b^{\mathbf{M},s}]$ and $\boldsymbol{\Lambda}_w^i = [\boldsymbol{\Lambda}_w^{\mathbf{1},i}, \boldsymbol{\Lambda}_w^{\mathbf{2},i}, \ldots, \boldsymbol{\Lambda}_w^{\mathbf{M},i}]$ represent coefficient matrices associated with class s and i, $s \neq i$, respectively. Eq. (26) and Eq. (27) are the same, and for ease of optimization, we formulate them in a different manner. (See **Appendix A**).

In this work, $\boldsymbol{\Lambda}_b^{\mathbf{1},s} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{1},s}, \boldsymbol{\alpha}_{1,2}^{\mathbf{1},s}, \ldots, \boldsymbol{\alpha}_{c,n_c^1}^{\mathbf{1},s}]$, where $\boldsymbol{\alpha}_{i,j}^{\mathbf{1},s}$ is the sparse coding with respect to class $s, s \neq i$ from domain $\mathbf{1}$, $i = 1, \ldots, c, j = 1, \ldots, n_i^{\mathbf{1}}$. $\boldsymbol{\Lambda}_w^{\mathbf{1},i} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{1},i}, \boldsymbol{\alpha}_{1,2}^{\mathbf{1},i}, \ldots, \boldsymbol{\alpha}_{c,n_c^1}^{\mathbf{1},i}]$, where $\boldsymbol{\alpha}_{i,j}^{\mathbf{1},i}$ is the sparse coding vector with respect to class i.

We update $\mathbf{B}$ in a class-by-class manner. When updating $\mathbf{B}_i$, $\mathbf{B}_s$, $s \neq i$ with respect to the other class is fixed. We exploit the gradient ascent for optimization. Using the chain rule, we obtain the following

$$\nabla_{\mathbf{B}_i} J(\mathbf{B}_i) = \frac{\partial J(\mathbf{B}_i)}{\partial \mathbf{S}_b^i}\frac{\partial \mathbf{S}_b^i}{\partial \mathbf{B}_i} + \frac{\partial J(\mathbf{B}_i)}{\partial \mathbf{S}_w}\frac{\partial \mathbf{S}_w}{\partial \mathbf{B}_i}. \tag{28}$$

Because there is no relationship between $\mathbf{S}_b^i$ and $\mathbf{B}_i$, $\partial \mathbf{S}_b^i / \partial \mathbf{B}_i = 0$, it is easy to observe the following

$$\frac{\partial J(\mathbf{B}_i)}{\partial \mathbf{S}_w} = \frac{-tr(\mathbf{A}^T\mathbf{S}_b^i\mathbf{A})\mathbf{A}^T\mathbf{A}}{\left(tr(\mathbf{A}^T\mathbf{S}_w\mathbf{A})\right)^2}, \tag{29}$$

$$\frac{\partial \mathbf{S}_w}{\partial \mathbf{B}_i} = 2\mathbf{K}_{\boldsymbol{\beta}}(\boldsymbol{\Lambda}_w^i)^T(\mathbf{K}_{\boldsymbol{\beta}}\mathbf{B}_i\boldsymbol{\Lambda}_w^i - \mathbf{K}_{\boldsymbol{\beta}}^i). \tag{30}$$

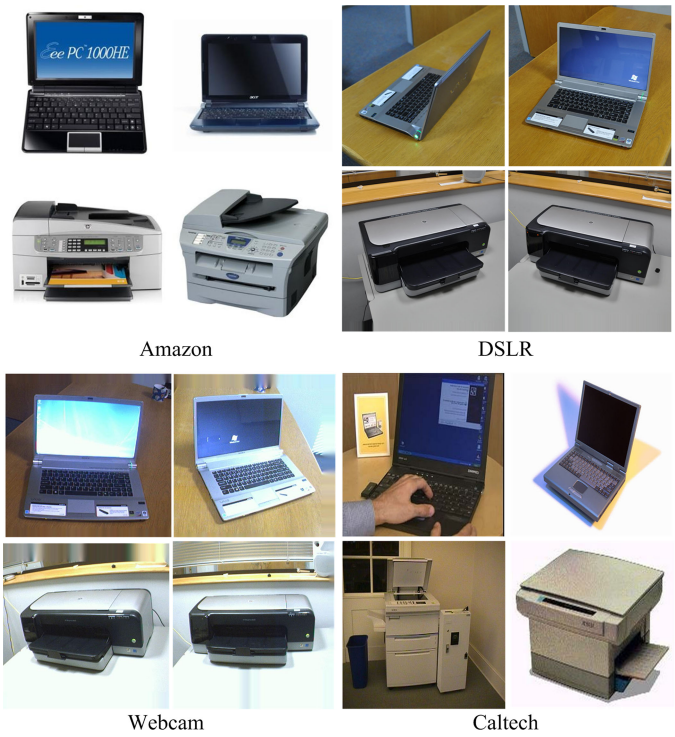

Fig. 2. Example images from laptop and printer categories in the domain adaptation dataset.

When the gradient in Eq. (28) is calculated, the sub-dictionary $\mathbf{B}_i$ can be updated using a projected gradient ascent procedure

$$\mathbf{B}_i = \mathbf{B}_i + \eta_2 \nabla_{\mathbf{B}_i} J(\mathbf{B}_i). \tag{31}$$

where η_2 is the step length.

When we obtain $\mathbf{A}$ and $\mathbf{B}$, we calculate the coding vector for each input sample from both domains. Algorithm 1 summaries the detailed steps of the proposed method.

V. Experiments

We use a domain adaptation dataset that consists of the Office [7] and Caltech-256 [55] datasets to verify the performance of MK-DADP for object recognition. The Office dataset includes 3 domains: Amazon, DSLR and Webcam. Every domain contains 31 categories. The Caltech 256 is the fourth domain, which contains 30,607 images of 256 categories, and each category has at least 80 images. Selected images from these domains are shown in Fig. 2, and obviously highlighting the differences between them.

To demonstrate the superiority of MK-DADP, we compare our approach with certain state-of-the-art domain adaptive algorithms, including SGF [6], GFK [17], DASRC [28], SDDL [30], OCPD-SRC [32], CDADL [57], DsGsDL [58], and another non-domain adaptation multi-kernel learning method MK-SR-ODP [45]. In addition, we also compare the results of MK-DADP with those of DASH-N [24]. To make a fair comparison, similar to DASH-N, we adopt two-layer networks to learn the feature representation and subsequently perform the recognition using the concatenated features. The experimental setup

Algorithm 1: MK-DADP

Input: Training set $\mathbf{Y_i}$ and corresponding class label $c_\mathbf{i}$, $\mathbf{i} = \mathbf{1}, \mathbf{2}$.
Kernel matrix $\mathbf{K}$, parameters λ, η_1, η_2, μ, dimension d and dictionary atoms K.
Initialize: Calculate SVD of kernel matrix $\mathbf{K_i} = \mathbf{V_i S_i V_i}^T$, then set
$\mathbf{A_i}$ as the matrix of eigenvectors corresponding to the largest d
eigenvalues such that $\mathbf{A_i}^T \mathbf{K_i A_i} = \mathbf{I}$. Randomly initialize $\mathbf{B}$ such that
$\mathbf{A}^T \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B}$ and $\mathbf{A_i}^T \mathbb{K}_\mathbf{i}^{(\mathbf{y})} \boldsymbol{\beta}$ have unit ℓ_2-norm.
Optimization
Repeat
Solve $\mathbf{A}$ and $\boldsymbol{\beta}$ with fixed $\mathbf{B}$ via Algorithm 2.
Solve $\mathbf{B}$ with fixed $\mathbf{A}$ and $\boldsymbol{\beta}$ via Eq.(31);
Until convergence
Output: dictionary $\mathbf{D}$, and projections $\{\mathbf{A}\}_{\mathbf{i=1}}^{\mathbf{2}}$.

Algorithm 2: Alternating Projection

Input: $\mathbf{S}_b$, $\mathbf{S}_w$, step length η_1, iterative number t_1, t_2.
Initialize ρ,
$\beta = 1/M$.
Repeat
Repeat
Compute $\mathbf{A}$ by solving Eq. (21).
Compute $\boldsymbol{\beta}$ by solving Eq. (22).
Until t_2 reached.
Compute$\rho = \rho - \eta_1 \frac{f(\rho)}{f'(\rho)}$.
Until $f(\rho)$=0 or t_1 reached.
Output: $\widetilde{\mathbf{A}}$, $\boldsymbol{\beta}$

is followed as in [24], denoted as MK-DADP (hierarchical). Finally, we compare our method with selected deep learning-based domain adaptation learning methods [59]–[62]. The average recognition rate is used to measure the performances of different methods.

A. Experimental Settings

As described in [32], we use three step-ups to evaluate the proposed method. In the first setup, 10 common classes from four domains are used, i.e., calculator, backpack, touring bike, video projector, coffee mug, computer mouse, and headphones. In each domain, the number of samples per category ranges from 8 to 151 for a total of 2533 images. In the second setup, we evaluate various approaches by using all 31 classes from Amazon, Webcam, and DSLR. Finally, we test the method for adaptation using multiple domains. For both cases, we select 20 samples per category for training from Amazon or Caltech, and each category selects 8 samples for training from DSLR and Webcam when used as the source. Three samples are selected for training when all of them are used in the target domain, and the remaining samples are used in testing. Each experiment is repeated 20 times for random train/test splits.

We have four parameters λ, η_1, η_2 and μ in our proposed MK-DADP method. In all experiments, these tuning parameters are determined by five-fold cross-validation on the training data. The parameters that yielded the best classification result are selected for the testing data. Since four parameters must be tuned and it is generally difficult to determine them simultaneously, we apply a stepwise selection strategy. Specifically, we first prefix the other parameters, seek the optimal value for one parameter and subsequently update these parameters with the newly learned parameters. The parameters are determined when the recognition performance reaches a stable rate. Concretely, we use $\lambda = 0.001$ for training and $\lambda = 0.001$ for testing to obtain satisfactory performance. Additionally, η_1 is the step length for updating ρ, and η_2 is the step size for updating the sub-dictionary $\mathbf{B}_i$. Parameters η_1 and η_2 are set to 1 and 0.1, respectively, which work well in all experiments. We specify parameter μ as 0.001. For other compared approaches, we exploit their original settings supplied in the corresponding papers. The base kernels are predetermined as a Gaussian kernel [i.e., $\kappa(\mathbf{y}_i, \mathbf{y}_j) = exp(-\|\mathbf{y}_i - \mathbf{y}_j\|/\gamma)$] with six degrees ($10^{-4}$, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10), inverse square distance kernel [i.e.,$\kappa(\mathbf{y}_i, \mathbf{y}_j) = 1/(\gamma\|\mathbf{y}_i - \mathbf{y}_j\|^2 + 1)$], and inverse distance kernel [i.e., $1/(\sqrt{\gamma}\|\mathbf{y}_i - \mathbf{y}_j\| + 1)$] with four degrees ($2^{-1}$, 2^{-2}, 2^{-3}, 2^{-4}). Finally, the pre-computed 800-bin SURF features supplied in [5], [7] are used in all of datasets.

B. Single Source

In this section, we set the dictionary atoms to 50, i.e., each class contains 5 atoms, and MK-SR-ODP and DASRC use all of the training samples as the dictionary. The dimension of the reduced space is set to 65. Table I tabulates the classification results of different methods on 8 pairs of source-target domains. The proposed MK-DADP obtains the highest recognition accuracies in most of the domain pairs. MK-DADP significantly outperforms OCPD-SRC [32] in all pairs of source-target domains. This result indicate that a combined kernel yields better performance than a predetermined one, and multiple kernel learning is more effective in the domain adaptation problem. We also observe that MK-SR-ODP achieves poorer performance than our method, and this result indicates that MKL is inefficient when the training data and test data are drawn from different domains. In addition, from Table I, we note that our proposed MK-DADP outperforms CDADL [57] (a common dictionary combined with a set of domain-specific dictionaries) and Ds-GsDL [58] in 6 out of 8 pairs of source-target domains. For pairs such as Amazon-Webcam, Webcam-Amazon, or DSLR-Amazon, we achieve more than 8% improvement over CDADL. Three reasons can explain this phenomenon. First, MK-DADP is a semi-supervised domain adaptation method in which the source domain contains sufficient labeled data, and the target domain contains a small amount of labeled data. However, CDADL

TABLE I
RECOGNITION RATES (%) ON A SINGLE DOMAIN

Method	C → A	C → D	A → C	A → W	W → C	W → A	D → A	D → W
MK-SR-ODP [45]	50.5±1.4	72.4±1.2	34.1±0.9	71.3±1.9	29.5±1.3	45.4±1.5	43.7±1.8	72.0±1.3
SGF [6]	40.2±0.7	36.6±0.8	37.7±0.5	37.9±0.7	29.2±0.7	38.2±0.6	39.2±0.7	69.5±0.9
GFK [17]	46.1±0.6	55.0±0.9	39.6±0.4	56.9±1.0	32.8±0.1	46.2±0.6	46.2±0.6	80.2±0.4
DASRC [28]	54.3±2.7	77.1±3.4	37.6±2.8	71.6±3.9	28.2±2.1	44.4±1.3	46.0±2.2	71.3±1.7
SDDL [30]	49.5±2.6	76.7±3.9	27.4±2.4	72.0±4.8	29.7±1.9	49.4±2.1	48.9±3.8	72.6±2.1
OCPD-SRC [32]	61.1±1.8	79.5±3.6	44.3±2.1	75.4±4.2	45.8±2.1	62.1±1.9	55.3±2.6	78.7±2.6
CDADL [57]	59.3±1.9	77.4±2.8	48.9±2.0	68.6±3.5	45.0±1.9	55.7±2.2	48.3±2.4	91.5±2.3
DsGsDL [58]	63.8±2.1	78.3±2.6	**49.0±1.7**	67.2±3.3	45.6±2.4	64.7±2.4	56.5±1.9	**95.6±2.2**
MK-DADP	**63.9±1.6**	**81.1±2.3**	45.5±2.3	**77.2±3.7**	**47.0±1.9**	**65.4±2.2**	**57.3±2.1**	81.2±2.5
DASH-N [24]	71.6±2.2	81.4±3.5	54.9±1.8	75.5±4.2	50.2±3.3	70.4±3.2	68.9±2.9	77.1±2.8
MK-DADP(Hierarchical)	**76.7±1.6**	**86.6±2.9**	**60.2±2.1**	**80.4±3.7**	**52.8±1.8**	**77.2±2.5**	**71.9±2.0**	81.6±2.3

and DsGsDL are unsupervised domain adaptation methods in which the target domain is completely unlabeled. Thus, selected discriminant information in the target domain is not fully exploited. Second, CDADL and DsGsDL optimize the combinations of common reconstruction errors or domain-specific reconstruction errors. Thus, the learned dictionaries might not be optimal for the final recognition task. Although our proposed MK-DADP is designed based on the proposed multi-kernel domain adaptive sparse representation-based classification (MK-DASRC), i.e., our model achieves an overall optimality for recognition in that the learned projections and dictionary are directly tailored for recognition. In addition, the problems for computing the sparse codes in the training and testing phases can be the same. Therefore, the resulting representations can satisfactorily fit MK-DASRC and simultaneously have discriminability. Finally, CDADL and DsGsDL are linear methods, which means that the classifier learned in a linear manner cannot satisfactorily characterize the corresponding feature space. Our method learns a classifier by using the multiple kernel learning technique in which the optimal kernel is learned by explicitly minimizing the within-class sparse reconstruction residuals of data from both domains and maximizing the between-class reconstruction residuals of data in the low-dimensional subspace. Hence, the learned dictionary can characterize the non-linear data.

As the high-level features contain more useful information than low-level ones, similar to DASH-N [24], MK-DADP also uses the idea of hierarchical networks to learn the feature representation and uses the concatenated features to perform classification. Using only two-layer networks, MK-DADP (hierarchical) significantly outperforms DASH-N and obtains the best results. In each layer of DASH-N, SDDL is used to learn the transformations and dictionary. Our method uses MK-DADP to learn the projections and dictionary in each layer, and thus MK-DADP (hierarchical) performs better than DASH-N.

We also investigate the classification results of different methods on all 31 classes. The results are listed in Table II. MK-DADP obtains the best performances on 2 domain pairs except for the results obtained for the Webcam-DLSR pair by DsGsDL [58]. As expected, MK-DADP outperforms OCPD-SRC and CDADL [57] for all pairs. In addition, MK-DADP (hierarchical) is superior to DASH-N and obtains the best performance in all domain pairs. This result indicates that the use of hierarchical structure is meaningful for transferring knowledge from the source domain to the target domain.

TABLE II
SINGLE SOURCE RECOGNITION RATES (%) ON ALL 31 CLASSES

Method	A → W	D → W	W → D
MK-SR-ODP [45]	50.3±0.8	40.8±1.7	37.4±0.8
SGF [6]	57±3.5	36±1.1	37±2.3
GFK [17]	46.4±0.5	61.3±0.4	66.3±0.4
DASRC [28]	53.3±1.9	47.6±2.4	47.1±2.8
SDDL [30]	50.1±2.5	51.2±2.1	50.6±2.6
OCPD-SRC [32]	56.7±2.2	61.8±1.9	63±2.5
CDADL [32]	52.7±0.7	59.1±1.9	64.0±2.2
DsGsDL [58]	51.2±0.6	62.5±2.0	**66.8±2.4**
MK-DADP	**59.1±1.6**	**63.3±1.7**	65.7±2.2
DASH-N [24]	60.6±3.5	67.9±1.1	71.1±1.7
MK-DADP(Hierarchical)	**64.4±2.7**	**72.3±1.9**	**75.4±2.3**

TABLE III
RECOGNITION RATES (%) ON MULTIPLE DOMAINS

Method	{D,A}→ W	{A,W}→ D	{W,D}→ A
MK-SR-ODP [45]	47.8±01.2	46.9±1.6	26.5±0.9
SGF [6]	52±2.5	39±1.1	28±0.8
DASRC [28]	56.5±1.3	56.1±1.6	27.9±2.2
SDDL [30]	57.8±2.4	56.7±2.3	24.1±1.6
OCPD-SRC [32]	61.3±2.1	59.5±2.2	38.7±1.9
MK-DADP	**62.8±1.9**	**61.9±1.9**	**41.3±1.6**
DASH-N [24]	64.5±2.3	68.6±3.7	41.8±1.1
MK-DADP (Hierarchical)	**67.2±2.1**	**69.7±2.5**	**47.6±1.6**

C. Multiple Sources

We extend our method to handle the multiple domain classification problem, and in this experiment, only the Office dataset is used. The dictionary atoms are set to 186 (i.e., each class contains 6 atoms), and the dimension of the feature space after projection is set to 90. Table III lists the results. We observe that MK-DADP obtains the best recognition performance. This result proves that MKL can address the

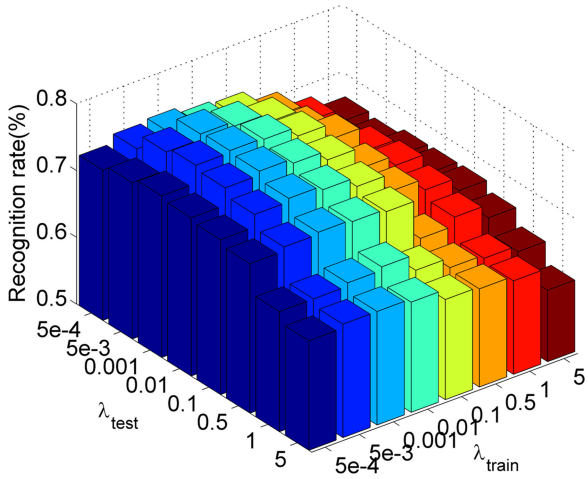

Fig. 3. Recognition rate of MK-DADP versus different values of λ.

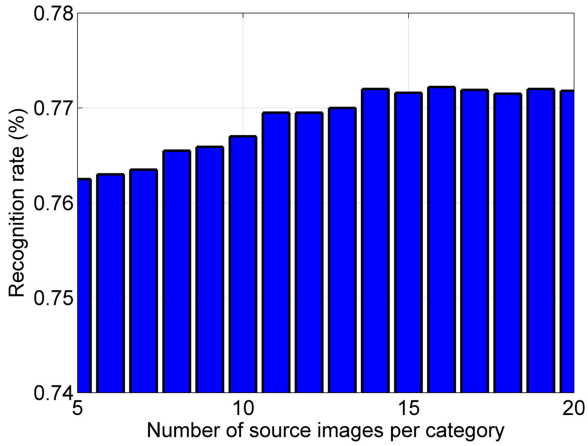

Fig. 4. Recognition rate of MK-DADP versus different number of source images.

cross-domain problem with combined data from multiple domains. Similarly, MK-SR-ODP performs poorly when the training data have a different distribution than the test data. Using multi-layer networks, MK-DADP (hierarchical) obtains the highest recognition rate and always outperforms DASH-N in all domain pairs.

D. Parameter Settings

In this section, we evaluate the performance of our method by tuning different parameters. We first estimate the influence of different sparsity regularization parameters λ_{train} and λ_{test} on the proposed method (MK-DADP). We use the Amazon/Webcam domain pair to conduct experiments and evaluate one parameter, while the other is fixed. Fig. 3 illustrates the performance of MK-DADP versus different λ. We note that MK-DADP is able to achieve stable performance when it is set as a suitable range.

We change the number of source images to estimate the performance of MK-DADP. Fig. 4 shows the results of MK-DADP versus different number of source images. As the number of source images increases, the MK-DADP's performance increases slightly. This indicates that we can increase the accuracy of our method by selecting additional source images. Fig. 5 illustrates the recognition rates of MK-DADP versus different dimensions under the Amazon/Webcam and Caltech/DLSR source-target pairs. We can observe that MK-DADP can obtain better performance when d is in the range of [40 70]. Similar results can also be obtained in the other domain pairs.

We also investigate the MK-DADP's performance versus different dictionary sizes (K_i, i.e., the number of atoms for each class). To observe the effect of different K_i, we perform experiments by varying K_i in the range of [1 8] with step 1 under the Amazon/Webcam and Caltech/DLSR source-target pairs. Fig. 6 illustrates the results for MK-DADP versus different values of K_i, and we can observe that our methods maintains better performance when the dictionary size reaches 4 or 5. With

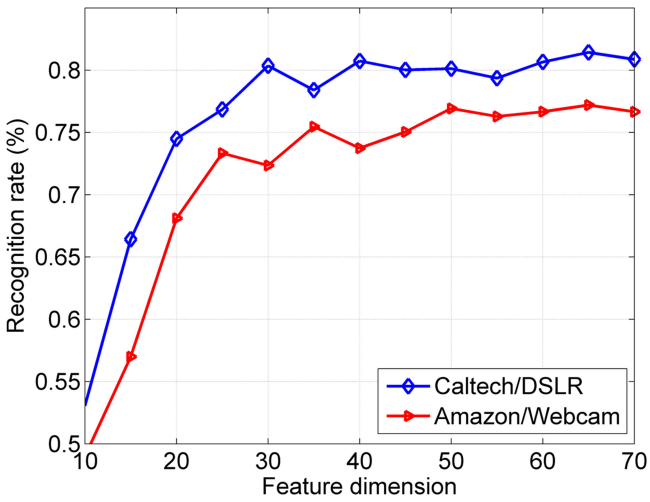

Fig. 5. Recognition rate of MK-DADP under different feature dimensions.

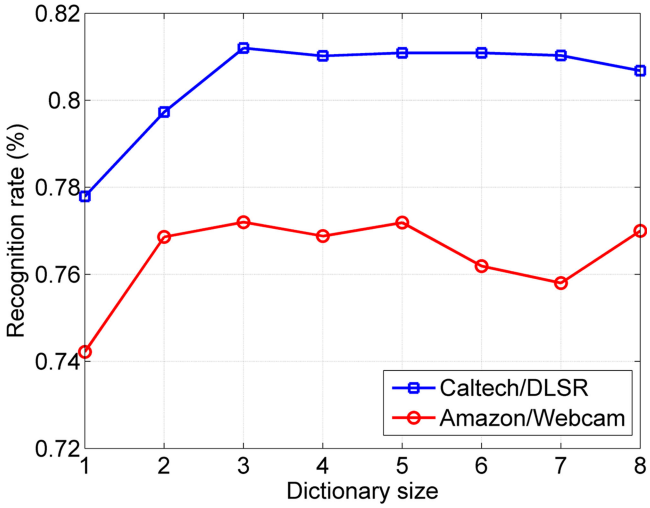

Fig. 6. Recognition rate of MK-DADP versus different K_i.

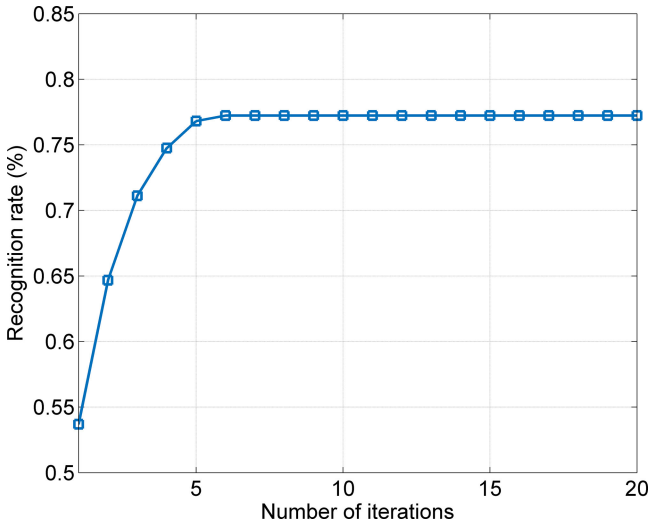

Fig. 7. Recognition rate of MK-DADP versus different number of iterations on the Amazon/Webcam source-target pair.

TABLE IV
COMPUTATION TIME (SECONDS) FOR TRAINING AND TESTING

Method	Training time	Testing time
SDDL [30]	$11.46s$	$0.33ms$
DASRC [28]	$8.57s$	$0.61ms$
OCPD-SRC [32]	$70.02s$	$0.58ms$
MK-DADP	$163.76s$	$0.64ms$

additional dictionary atoms, the performance of MK-DADP varies in a small range.

Finally, we investigate the performance of our MK-DADP versus different number of iterations. Fig. 7 shows the recognition rate of our MK-DADP over different number of iterations on the Amazon/Webcam source-target pair. We observe that our proposed MK-DADP can achieve stable performance in several iterations.

E. Running Time

In this section, we compare the running time of MK-DADP with those of OCPD-SRC [32], SDDL [30] and DASRC [28]. We conduct this experiment using single-source domain adaptation. The parameters are set the same as in the previous experiment. Our hardware configuration includes a 3.30 GHz CPU and an 8GB RAM. Table IV lists the computation times of different methods on the Amazon/Webcam domain pair. The reported testing time refers to the time needed to recognize one image. We note that our method requires more training time than OCPD-SRC, DASRC and SDDL. However, the testing time for MK-DADP is comparable to that of other domain adaptation methods. Note that the projections and dictionary can be learned offline and that classification can be performed online. Therefore, this option does not affect our method for practical application.

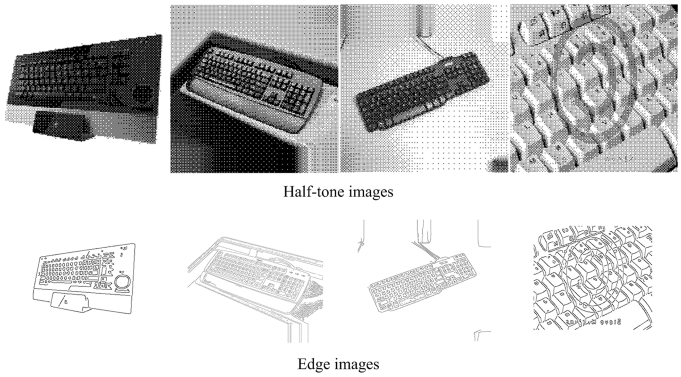

Fig. 8. Example images from the keyboard class in halftone and edge domains.

F. Deep Features

In this section, we compare the proposed MK-DADP with four state-of-the-art deep learning-based domain adaptation methods, such as DDC [60], DANN [61], DAN [62], DeCAF_6 S+T [59], and CDADL+DeCAF_6 [57], on the Office dataset. For fair comparison, we evaluate our method using the deep features (DeCAF_6) supplied by [59] and use the standard experimental protocol from [7]. The results for 6 pairs of source-target domains are reported in Table V. We note that the deep feature representation indeed improves the performance of our MK-DADP method if the SURF features [7] are used in recognition. Our method achieves the best performance in 4 out of 6 pairs of source-target domains when using deep features. This result demonstrates that the proposed method is competitive with the deep-learning based domain adaptation methods.

G. Halftone and Edge Images

In this section, to verify the performance of MK-DADP in a wide range of domains, we report the experimental results on two new datasets obtained by performing half-toning and edge detection from the office dataset. Fig. 8 illustrates selected images from different domains. We obtain half-toning with the dithering algorithm in [53], which imitates the effect of jet-printing technology in the past. We use the Canny edge detector [54] algorithm to obtain edge images, and the threshold is set to 0.07. Following the same experimental setup as described in the above section, we extract SURF features for both domains. To compare with DASH-N, a two-layer networks is used to learn the feature representation and subsequently perform classification [24].

The recognition results are reported in Table VI and Table VII. From these tables, we observe that MK-DADP obtains the best performances in all pairs of source-target domains and that MK-DADP (Hierarchical) also outperforms DASH-N. Again, this demonstrates the capability of the proposed method for adapting well to new domains.

TABLE V
RECOGNITION RATE (%) ON OFFICE DATASET COMPARED WITH DEEP-LEARNING BASED METHODS

Method	A → D	A→ W	D→ A	D→ W	W→ A	W→ D
DDC [60]	-	59.4±0.8	-	92.5±0.3	-	91.7±0.8
DANN[61]	-	53.6±0.2	-	71.2±0.0	-	83.5±0.0
DAN [62]	-	66.0±0.4	-	93.5±0.2	-	95.3±0.3
$DeCAF_6$ S+T [59]	-	80.7±2.3	-	94.8±1.2	-	-
CDADL+$DeCAF_6$ [57]	84.3±0.6	81.4±1.2	65.0±0.5	**95.9±0.3**	63.3±0.7	**97.6±0.3**
MK-DADP+$DeCAF_6$	**87.6±0.4**	**85.8±1.1**	**69.2±0.3**	95.7±0.4	**67.8±0.6**	97.2±0.3

TABLE VI
RECOGNITION RATES OF DIFFERENT METHODS ON HALF-TONING DATASET: 10 COMMON CLASSES

Method	C → A	C → D	A → C	A → W	W → C	W → A	D → A	D → W
MK-SR-ODP [45]	55.3±4.5	63.9±3.1	43.7±2.8	62.4±3.2	34.3±2.5	49.3±2.4	54.6±3.2	57.4±2.3
DASRC [28]	59.2±4.7	68.2±5.1	36.6±2.3	67.5±3.9	33.7±2.2	48.7±2.8	52±3.7	68.5±4.1
SDDL [30]	52.2±3.9	66.7±5.5	34.1±3.5	69.2±4.2	34.6±2.8	51.2±3.4	54.1±2.7	71.6±5.3
OCPD-SRC [32]	64.3±3.3	73.4±3.8	45.8±2.7	77.7±3.0	43.0±2.6	60.9±3.5	59.7±2.7	74.7±4.4
MK-DADP	**65.5±3.1**	**75.1±3.4**	**47.1±3.0**	**78.6±3.1**	**43.7±2.9**	**62.8±3.2**	**64.2±2.5**	**76.4±3.9**
DASH-N [24]	70.2±2.7	79.6±4.3	52.4±2.3	86.2±4.1	43.3±3.9	66.1±3.7	67.2±3.5	80.7±2.1
MK-DADP (Hierarchical)	**74.8±2.9**	**83.7±3.7**	**57.4±2.5**	**89.0±3.3**	**48.6±3.4**	**71.7±3.4**	**72.3±2.2**	**84.1±2.8**

TABLE VII
RECOGNITION RATES OF DIFFERENT METHODS ON EDGE DATASET: 10 COMMON CLASSES

Method	C → A	C → D	A → C	A → W	W → C	W → A	D → A	D → W
MK-SR-ODP [45]	55.7±1.9	60.3±4.2	36.6±3.8	58.3±2.2	32.7±3.6	52.7±4.4	51.6±3.7	58.4±2.2
DASRC [28]	58.3±5.4	61.9±5.0	34.1±3.6	61.1±4.7	32.8±2.9	52.4±3.3	51.8±2.9	60.6±3.4
SDDL [30]	52.9±5.2	63.8±6.3	32.4±3.2	62.5±5.7	33.5±2.9	55.2±2.8	55.4±3.3	65.3±4.7
OCPD-SRC [32]	65.2±4.7	70.7±5.2	40.9±2.9	66.6±4.1	42.2±3.4	61.4±3.0	59.1±3.5	66.4±3.8
MK-DADP	**65.3±4.4**	**71.3±4.8**	**42.3±2.8**	**67.5±4.1**	**43.9±3.1**	**63.6±2.7**	**63.8±3.3**	**67.7±3.5**
DASH-N [24]	74.2±3.9	75.7±2.6	44.3±2.5	74.2±4.5	46.7±3.1	68.9±2.2	67.7±3.4	74.5±3.2
MK-DADP (Hierarchical)	**79.5±3.8**	**80.3±3.1**	**48.3±2.2**	**76.0±4.1**	**51.7±3.3**	**73.6±3.1**	**72.9±2.9**	**76.8±3.0**

VI. CONCLUSION

We propose a multi-kernel domain-adaptive sparse representation-based classifier (MK-DASRC). Based on the decision criterion of MK-DASRC, a multi-kernel based domain adaptation discriminative projections method (MK-DADP) is presented. By jointly learning the projections of data from both domains and a common discriminative dictionary, MK-DADP obtains a better representation of data from different domains in the projected space. We presented an alternative iterative algorithm used to solve the proposed approach. The experiment results demonstrate that our method performs better than many state-of-the-art approaches.

APPENDIX

To prove that Eq. (26) and Eq. (27) are the same, we rewrite Eq. (8) as follows:

$$\begin{aligned} J_w^{\mathbf{1}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{1}}}(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})) \right. \\ &\qquad \left. \times\ (\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_i(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}}))^T\right) \\ &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{1}}}(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}_i\boldsymbol{\alpha}_{i,j}^{\mathbf{1},i}) \right. \\ &\qquad \left. (\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}_i\boldsymbol{\alpha}_{i,j}^{\mathbf{1},i})^T\right) \\ &= \sum_{i=1}^{c} tr(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{Y}_i^{\mathbf{1}}) - \mathbf{D}_i\boldsymbol{\Lambda}_w^{\mathbf{1},i})(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{Y}_i^{\mathbf{1}}) - \mathbf{D}_i\boldsymbol{\Lambda}_w^{\mathbf{1},i})^T \\ &= \sum_{i=1}^{c} \|\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{Y}_i^{\mathbf{1}}) - \mathbf{D}_i\boldsymbol{\Lambda}_w^{\mathbf{1},i}\|_F^2, \end{aligned} \tag{32}$$

where $\boldsymbol{\Lambda}_w^{\mathbf{1},i} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{1},i}, \boldsymbol{\alpha}_{1,2}^{\mathbf{1},i}, \ldots, \boldsymbol{\alpha}_{c,n_c^{\mathbf{1}}}^{\mathbf{1},i}]$, $\boldsymbol{\alpha}_{i,j}^{\mathbf{1},i}$ is the representation coefficient vector with respect to class i, $i = 1, \ldots, c, j = 1, \ldots, n_i$.

Similarly, Eq. (9) can be rewritten as

$$\begin{aligned} J_b^{\mathbf{1}} &= tr\left(\sum_{i=1}^{c}\sum_{j=1}^{n_i^{\mathbf{1}}}\sum_{s\neq i}(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}})) \right. \\ &\qquad \left. \times\ (\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}\delta_s(\boldsymbol{\alpha}_{i,j}^{\mathbf{1}}))^T\right) \\ &= \sum_{i=1}^{c} tr\left(\sum_{j=1}^{n_i^{\mathbf{1}}}\sum_{s\neq i}(\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}_s\boldsymbol{\alpha}_{i,j}^{\mathbf{1},s}) \right. \\ &\qquad \left. \times\ (\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{y}_{i,j}^{\mathbf{1}}) - \mathbf{D}_s\boldsymbol{\alpha}_{i,j}^{\mathbf{1},s})^T\right) \\ &= \sum_{i=1}^{c} \|\mathbf{P}_{\mathbf{1}}^T\phi(\mathbf{Y}_i^{\mathbf{1}}) - \mathbf{D}_s\boldsymbol{\Lambda}_b^{\mathbf{1},s}\|_F^2, \end{aligned} \tag{33}$$

where $\boldsymbol{\Lambda}_b^{\mathbf{1},s} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{1},s}, \boldsymbol{\alpha}_{1,2}^{\mathbf{1},s}, \ldots, \boldsymbol{\alpha}_{c,n_c^{\mathbf{1}}}^{\mathbf{1},s}]$, $\boldsymbol{\alpha}_{i,j}^{\mathbf{1},s}$ is the representation coefficient vector with respect to s, $s \neq i$.

In the same way, the formulation in Eq. (10) and Eq. (11) for the target domain can be rewritten as:

$$J_w^{\mathbf{2}} = \sum_{i=1}^{c} \|\mathbf{P_2}^T \phi(\mathbf{Y}_i^{\mathbf{2}}) - \mathbf{D}_i \mathbf{\Lambda}_w^{\mathbf{2},i}\|_F^2, \tag{34}$$

and

$$J_b^{\mathbf{2}} = \sum_{i=1}^{c} \|\mathbf{P_2}^T \phi(\mathbf{Y}_i^{\mathbf{2}}) - \mathbf{D}_s \mathbf{\Lambda}_b^{\mathbf{2},s}\|_F^2, \tag{35}$$

where $\mathbf{\Lambda}_w^{\mathbf{2},i} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{2},i}, \boldsymbol{\alpha}_{1,2}^{\mathbf{2},i}, \ldots, \boldsymbol{\alpha}_{c,n_c^{\mathbf{2}}}^{\mathbf{2},i}]$, and $\mathbf{\Lambda}_b^{\mathbf{2},s} = [\boldsymbol{\alpha}_{1,1}^{\mathbf{2},s}, \boldsymbol{\alpha}_{1,2}^{\mathbf{2},s}, \cdots, \boldsymbol{\alpha}_{c,n_c^{\mathbf{2}}}^{\mathbf{2},s}]$. $\boldsymbol{\alpha}_{i,j}^{\mathbf{2},i}$, $\boldsymbol{\alpha}_{i,j}^{\mathbf{2},s}$ are the coefficient vectors which corresponding to class i and class s, respectively. Finally, we maximize $J_b^{\mathbf{1}}$ and $J_b^{\mathbf{2}}$ as follows:

$$\begin{aligned}\max J_b &= \max\{J_b^{\mathbf{1}} + J_b^{\mathbf{2}}\} \\ &= \max \sum_{i=1}^{c} \sum_{s \neq i} \left(\|\mathbf{P}^T \phi(\mathbf{Y}_i) - \mathbf{D}_s \mathbf{\Lambda}_b^s\|_F^2\right),\end{aligned} \tag{36}$$

and simultaneously minimize $J_w^{\mathbf{1}}$ and $J_w^{\mathbf{2}}$ as follows:

$$\begin{aligned}\min J_w &= \min\{J_w^{\mathbf{1}} + J_w^{\mathbf{2}}\} \\ &= \min \sum_{i=1}^{c} \left(\|\mathbf{P}^T \phi(\mathbf{Y}_i) - \mathbf{D}_i \mathbf{\Lambda}_w^i\|_F^2\right),\end{aligned} \tag{37}$$

where

$$\begin{aligned}\mathbf{P}^T &= [\mathbf{P_1}^T, \mathbf{P_2}^T], \phi(\mathbf{Y}_i) = \begin{bmatrix} \phi(\mathbf{Y}_i^{\mathbf{1}}) & \mathbf{0} \\ \mathbf{0} & \phi(\mathbf{Y}_i^{\mathbf{2}}) \end{bmatrix}, \mathbf{\Lambda}_b^s \\ &= [\mathbf{\Lambda}_b^{\mathbf{1},s}, \mathbf{\Lambda}_b^{\mathbf{2},s}]\end{aligned}$$

and $\mathbf{\Lambda}_w^i = [\mathbf{\Lambda}_w^{\mathbf{1},s}, \mathbf{\Lambda}_w^{\mathbf{2},s}]$. Exploiting $\mathbf{P}^T = \mathbf{A}^T \phi(\mathbf{Y})^T$, $\mathbf{D}_i = \mathbf{P}^T \phi(\mathbf{Y}) \mathbf{B}_i$ and $\mathbf{D}_s = \mathbf{P}^T \phi(\mathbf{Y}) \mathbf{B}_s$, the objective function can be rewritten as follows

$$\begin{aligned}J &= \max_{\mathbf{A},\mathbf{B}} \\ &\frac{\sum_{i=1}^{c} \sum_{s \neq i} tr\left(\mathbf{A}^T (\mathbf{K}_\beta^i - \mathbf{K}_\beta \mathbf{B}_s \mathbf{\Lambda}_b^s)(\mathbf{K}_\beta^i - \mathbf{K}_\beta \mathbf{B}_s \mathbf{\Lambda}_b^s)^T \mathbf{A}\right)}{\sum_{i=1}^{c} tr\left(\mathbf{A}^T (\mathbf{K}_\beta^i - \mathbf{K}_\beta \mathbf{B}_i \mathbf{\Lambda}_w^i)(\mathbf{K}_\beta^i - \mathbf{K}_\beta \mathbf{B}_i \mathbf{\Lambda}_w^i)^T \mathbf{A}\right)} \\ &= \max_{\mathbf{B}_i} \sum_{i=1}^{c} \frac{tr(\mathbf{A}^T \mathbf{S}_b^i \mathbf{A})}{tr(\mathbf{A}^T \mathbf{S}_w \mathbf{A})},\end{aligned} \tag{38}$$

where $\mathbf{S}_b^i = \sum_{s \neq i} (\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B}_s \mathbf{\Lambda}_b^s)(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B}_s \mathbf{\Lambda}_b^s)^T$ and $\mathbf{S}_w = \sum_{i=1}^{c} \mathbf{S}_w^i = \sum_{i=1}^{c} (\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B}_i \mathbf{\Lambda}_b^i)(\mathbf{K}_{\boldsymbol{\beta}}^i - \mathbf{K}_{\boldsymbol{\beta}} \mathbf{B}_i \mathbf{\Lambda}_b^i)^T$.

REFERENCES

[1] M. Abdelwahab, and C. Busso, "Supervised domain adaptation for emotion recognition from speech," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 5058–5062.

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

[3] M. Gong *et al.*, "Domain adaptation with conditional transferable components," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2839–2848.

[4] S. Pan, and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[5] K. Brian, 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.

[6] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 999–1006.

[7] 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.

[8] A. Shrivastava, S. Shekhar, and V. M. Patel, "Unsupervised domain adaptation using parallel transport on Grassmann manifold," in *Proc. IEEE Winter Conf. Appl. Comput. Vision*, 2014, pp. 277–284.

[9] T. Yao, Y. Pan, C.-W. Ngo, H. Li, and T. Mei, "Semi-supervised domain adaptation with subspace learning for visual recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 2142–2150.

[10] T. Gebru, J. Hoffman, and F. Li, "Fine-grained recognition in the wild: A multi-task domain adaptation approach," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 1349–1358.

[11] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonly, "Optimal transport for domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 9, pp. 1853–1865, Sep. 2017.

[12] Y. Xiao, T. Zhang, and C. Xu, "Cross-domain feature learning in multimedia," *IEEE Trans. Multimedia*, vol. 17, no. 1, pp. 64–78, Jan. 2015.

[13] H.-C. Shin *et al.*, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans. Multimedia*, vol. 35, no. 5, pp. 1285–1298, May 2016.

[14] S. Qian, T. Zhang, and C. Xu, "Cross-domain collaborative learning via discriminative nonparametric Bayesian model," *IEEE Trans. Multimedia*, vol. 20, no. 8, pp. 2086–2099, Aug. 2018.

[15] P. Jing, Y. Su, L. Nie, and H. Gu, "Predicting image memorability through adaptive transfer learning from external sources," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 1050–1062, May 2017.

[16] C. Sun, B.-K. Bao, and C. Xu, "Knowing verb from object: Retagging with transfer learning on verb-object concept images," *IEEE Trans. Multimedia*, vol. 17, no. 10, pp. 1747–1759, Oct. 2015.

[17] 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.

[18] B. Fernando, A. Habrard, M. Sebban, and T. Tuyelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proc. Int. Conf. Comput. Vision*, 2013, pp. 2960–2967.

[19] W. Li, L. Duan, D. Xu, and I. W. Tsang, "Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1134–1148, Jun. 2014.

[20] Y. Shi and F. Sha, "Information-theoretical learning of discriminative adaption," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1079–1086.

[21] J. Hoffman, E. Rodner, J. Donahue, T. Darrell, and K. Saenko, "Efficient learning for domain-invariant image representation," 2013, arXiv:1301.3224.

[22] W. Li, Z. Xu, D. Xu, D. Dai, and L. V. Gool, "Domain generalization and adaptation using low rank exemplar SVMs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1114–1127, May 2018.

[23] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 53–69, May 2015.

[24] H. V. Nguyen, H. T. Ho, V. M. Patel, and R. Chellappa, "DASH-N: Joint hierarchical domain adaptation and feature learning," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5479–5491, Dec. 2015.

[25] H. Lu *et al.*, "When unsupervised domain adaptation meets tensor representations," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 599–608.

[26] J. Ni, Q. Qiu, and R. Chellappa, "Subspace interpolation via dictionary learning for unsupervised domain adaptation," in *Proc. Int. Conf. Comput. Vision*, 2013, pp. 692–699.

[27] Q. Qiu, V. M. Patel, P. Turaga, and R. Chellappa, "Domain adaptive dictionary learning," in *Proc. Eur. Conf. Comput. Vision*, 2012, pp. 631–645.

[28] H. Zhang, V. M. Patel, S. Shekhar, and R. Chellappa, "Domain-adaptive sparse representation-based classification," in *Proc. Autom. Face Gesture Recognit. Workshops*, 2015, pp. 1–8.

[29] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa, "Generalized domain-adaptive dictionaries," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2013, pp. 361–368.

[30] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa, "Couple projections for adaptation of dictionaries," *IEEE Trans. Image Process.*, vol. 24, no. 10, pp. 2941–2954, Oct. 2015.
[31] B. Lu, A. Chellapp, and N. Nasrabadi, "Incremental dictionary learning for unsupervised domain adaptation," in *Proc. Brit. Mach. Vision Conf.*, 2015, pp. 108.1–108.12.
[32] G. Zhang, H. Sun, F. Porikli, Y. Liu, and Q. Sun, "Optimal couple projections for domain adaptive sparse representation-based classification," *IEEE Trans. Image Process.*, vol. 26, no. 12, pp. 5922–5935, Dec. 2017.
[33] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 513–520.
[34] M. Chen, Z. Xu, K. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1–8.
[35] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proc. Int. Conf. Comput. Vision*, 2015, pp. 4068–4076.
[36] E. Tzeng and J. Hoffman, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 7167–7176.
[37] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.
[38] K. Bousmalis, N. Silberman, and D. Dohan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3722–3731.
[39] S. Herath, M. Harandi, and F. Porikli, "Learning an invariant hilbert space for domain adaptation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3845–3854.
[40] L. Zhang *et al.*, "Kernel sparse representation-based classifier," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1684–1695, Apr. 2012.
[41] G. Zhang, H. Sun, G. Xia, and Q. Sun, "Kernel collaborative representation based dictionary learning and discriminative projection," *Neurocomputing*, vol. 207, pp. 300–309, 2016.
[42] G. Zhang, H. Sun, G. Xia, L. Feng, and Q. Sun, "Kernel dictionary learning based discriminant analysis," *J. Vision Commun. Image Representation*, vol. 40, pp. 470–484, 2016.
[43] S. S. Bucak, R. Jin, and A. K. Jain, "Multiple kernel learning for visual object recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1354–1369, Jul. 2014.
[44] X. Liu, L. Wang, J. Zhang, and J. Yin, "Sample-adaptive multiple kernel learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 1617–1623.
[45] G. Zhang, H. Sun, G. Xia, and Q. Sun, "Multiple kernel sparse representation based orthogonal discriminative projection and its cost-sensitive extension," *IEEE Trans. Image Process.*, vol. 25, no. 9, pp. 4271–4285, Sep. 2016.
[46] A. Shrivastava, V. M. Patel, and R. Chellappa, "Multiple kernel learning for sparse representation-based classification," *IEEE Trans. Image Process.*, vol. 23, no. 7, pp. 3013–3024, Jul. 2014.
[47] A. Shrivastava, J. K. Pillai, and V. M. Patel, "Multiple kernel-based dictionary learning for weakly supervised classification," *Pattern Recognit.*, vol. 48, no. 8, pp. 2667–2675, 2015.
[48] X. Wu, Q. Li, L. Xu, K. Chen, and L. Yao, "Multi-feature kernel discriminant dictionary learning for face recognition," *Pattern Recognit.*, vol. 66, pp. 404–411, 2017.
[49] L. Duan, I. W. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 465–479, Mar. 2012.
[50] W. Wang, H. Wang, C. Zhang, and Y. Gao, "Fredholm multiple kernel learning for semi-supervised domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2732–2738.
[51] Z. Xu, R. Jin, H. Yang, I. King, and M. R. Lyu, "Simple and efficient multiple kernel learning by group lasso," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 1175–1182.
[52] Z. Wen and W. Yin, "A feasible method for optimization with orthogonality constraints," *Math. Program.*, vol. 142, no. 1/2, pp. 397–434, 2013.
[53] V. Monga, N. Damera-Venkata, H. Rehman, and B. L. Evans, *Halftoning Toolbox for MATLAB*, 2005. [Online]. Available: http://users.ece.utexas.edu/bevans/projects/halftoning/toolbox/
[54] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Nov. 1986.
[55] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category data set," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. 7694, 2007.
[56] Y. Han *et al.*, "Image attribute adaptation," *IEEE Trans. Multimedia*, vol. 16, no. 4, pp. 1115–1126, Jun. 2014.
[57] H. Xu, J. Zheng, A. Alavi, and R. Chellappa, "Cross-domain visual recognition via domain adaptive dictionary learning," 2018, arXiv:1804.04687.
[58] B. Yang, A. J. Ma, and P. C. Yuen, "Domain-shared group-sparse dictionary learning for unsupervised domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 7453–7460.
[59] J. Donahue *et al.*, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 647–655.
[60] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014, arXiv:1412.3474.
[61] M. Ghifary, W. B. Kleijn, and M. Zhang, "Domain adaptive neural for object recognition," in *Proc. Int. Conf. Artif. Intell.*, 2014, pp. 898–904.
[62] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
[63] D. Zhang, J. Han, C. Li, J. Wang, and X. Li, "Detection of co-salient objects by looking deep and wide," *Int. J. Comput. Vision*, vol. 120, no. 2, pp. 215–232, 2016.
[64] G. Gheng, P. Zhou, and J. Han, "Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7405–7415, Dec. 2016.
[65] J. Han *et al.*, "Background prior-based salient object detection via deep reconstruction residual," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 8, pp. 1309–1321, Aug. 2015.

Yuhui Zheng received the B.Sc. degree in pharmacy engineering and the Ph.D. degree in pattern recognition and intelligent systems from Nanjing University of Science and Technology, Nanjing, China, in 2004 and 2009, respectively. He is currently an Associate Professor with the College of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China. His research interests include multimedia data analysis and processing, image and video segmentation, and computer vision.

Xilong Wang received the B.Sc. degree in information and computing science from Jilin Agricultural University, Jilin, China, in 2017. He is currently working toward the M.Sc. degree in software engineering, Nanjing University of Information Science and Technology, Nanjing, China. His research field is multimedia processing.

Guoqing Zhang received the B.Sc. and M.Sc. degrees in information engineering from the Yangzhou University, Yangzhou, China, in 2009 and 2012, respectively, and the Ph.D. degree in pattern recognition and intelligent systems from Nanjing University of Science and Technology, Nanjing, China, in 2017. He is currently an Assistant Professor with the College of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China. His main research fields are image processing and computer vision, multimedia analysis, and pattern recognition.

Baihua Xiao received the B.Sc. degree in automatic control from Northwestern Polytechnical University, Xi'an, China, in 1995, and the Ph.D. degree in computer science from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2000. He is currently a Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, CAS, Beijing, China. His main research interests include pattern recognition, artificial intelligence, and computer vision.

Fu Xiao (M'12) received the M.Sc. degree in computer science from Jiangsu University of Science and Technology, Zhenjiang, China, in 2003, and the Ph.D. degree in pattern recognition and intelligent systems from Nanjing University of Science and Technology, Nanjing, China, in 2007. He is currently a Professor with the School of Computer Science, Nanjing University of Posts and Telecommunications. His main research areas cover multimedia computing, wireless networks, and network security.

Jianwei Zhang received the B.Sc. degree in computer science from Wuhan University, Wuhan, China, in 1998, and the Ph.D. degree in pattern recognition and intelligent system from Nanjing University of Science and Technology, Nanjing, China, in 2006. He is currently a Professor with the School of Mathematics and Statistics, Nanjing University of Information Science and Technology, Nanjing, China. His research areas cover artificial intelligence, remote sensing information systems, and data mining.