Semi-Supervised Domain Adaptation by Covariance Matching

Limin Li and Zhenyue Zhang

Abstract—Transferring knowledge from a source domain to a target domain by domain adaptation has been an interesting and challenging problem in many machine learning applications. The key problem is how to match the data distributions of the two heterogeneous domains in a proper way such that they can be treated indifferently for learning. We propose a covariance matching approach DACoM for semi-supervised domain adaptation. The DACoM embeds the original samples into a common latent space linearly such that the covariance mismatch of the two mapped distributions is minimized, and the local geometric structure and discriminative information are preserved simultaneously. The KKT conditions of DACoM model are given as a nonlinear eigenvalue equation. We show that the KKT conditions could at least ensure local optimality. An efficient eigen-updating algorithm is then given for solving the nonlinear eigenvalue problem, whose convergence is guaranteed conditionally. To deal with the case when homogeneous information could only be matched nonlinearly, a kernel version of DACoM is further considered. We also analyze the generalization bound for our domain adaptation approaches. Numerical experiments on simulation datasets and real-world applications are given to comprehensively demonstrate the effectiveness and efficiency of the proposed approach. The experiments show that our method outperforms other existing methods for both homogeneous and heterogeneous domain adaptation.

Index Terms—Covariance matching, transfer learning, domain adaptation

1 Introduction

Traditional supervised learning approaches often require sufficient labelled samples to train a good learning model. In many real-world applications, however, labelled samples are very deficient generally. domain adaptation is a newly developed method for learning a classifier in target domain with a very few or no labelled samples, by making use of a large amount of labelled data from a source domain [1]. In the resent years, domain adaptation has been applied in various fields successfully, such as object recognition [2], [3], [4], part of speech tagging [5], [6], cross language text classification [7], [8], [9], Wi-Fi localization [10], [11], sentiment classification [12], [13], [14], [15], [16]. And so on.

Domain adaptation aims to correct the discrepancy of source domain to match the target domain so that the transformed source samples with labels can be efficiently utilized for classifying the target samples. Many approaches have been proposed for domain adaptation along the two lines: *instance transfer* and *feature-representation-transfer*. The instance transfer utilizes source instances as weighting or importance sampling in the target domain [17], [18]. The feature-representation-transfer attempts to learn an adaptive classifier directly [19], [20], [21], [22], or learn a common subspace through which the source and target data can be well matched to each other via corresponding learning [5], [6], [8], [12], [20], property preservation [20], [22], [23], distribution adaptation [21], [24], [25], [26], [27], or optimal transport [28].

The key of domain adaptation is to build a suitable connection between the source domain and target domain for matching. It was shown in [29] and [30] that the probability divergence of the distributions between the two domains plays an important role in the generalization bound. Feature representation is a simple and commonly used way for reducing the divergence, building the connection, and finally learning a classifier on the feature projections [24]. In the homogenous case when the source samples $\{x_i^s\}$ and the target samples $\{x_j^t\}$ live in the same space, one may look for a single mapping ϕ that makes the projections $\{\tilde{x}_i^s = \phi(x_i^s)\}$ and $\{\tilde{x}_j^t = \phi(x_j^t)\}$ well matched under a given measurement. For instance, in the early work [31], the difference of two distributions of $\{x_i^s\}$ and $\{x_j^t\}$ is measured by Maximum Mean Discrepancy, if ϕ is defined by the kernel mapping of the samples into the reproducing kernel Hilbert space (RKHS) with a given kernel, the MMD is given $\|\frac{1}{n_s}\sum_i \tilde{x}_i^s - \frac{1}{n_t}\sum_i \tilde{x}_j^t\|_{\mathcal{H}}^2$. MMDE [24] looks for an optimal kernel to minimize the MMD, which results in a semi-definite programming. In [26] or [32], the Bregman divergence and the Kullback-Leibler (KL) divergence were adopted for optimizing ϕ, respectively. Notice that both the two divergences require density estimation.

The heterogeneous domain adaptation has wide applications such as cross-language text classification, text-aided image classification and cross-platform disease classification. For instance, the word features for representing English text

- *L. Li is with the School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China. E-mail: liminli@mail.xjtu.edu.cn.*
- *Z. Zhang is with the School of Mathematics Science and State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, Zhejiang 310027, China. E-mail: zyzhang@zju.edu.cn.*

Manuscript received 8 Jan. 2017; revised 25 June 2018; accepted 8 Aug. 2018. Date of publication 22 Aug. 2018; date of current version 10 Oct. 2019. (Corresponding author: Zhenyue Zhang.)
Recommended for acceptance by J. Ye.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2018.2866846

documents are different with those for German text documents. In text-aided image classification, the source domain has word features and the target domain has visual features [4]. Because heterogeneous domain adaptation tries to match the heterogeneous information, it is more complicated, especially when the feature representation spaces of the two domains are completely different. This is why there are less work on heterogeneous domain adaptation, compared with works on homogeneous domain adaptation. In the literature, the strategies adopted for heterogeneous domain adaptation are spectral mapping [27], feature transformation [2], [3], [7], domain specific feature selection [33], kernel matching [20], manifold alignment [23], and exploiting auxiliary resources [4], [9]. We will briefly discuss these related methods. It is commonly assumed in these approaches that there is a shared latent feature representation between the source domain and target domain. Such an assumption will be released in our new method.

In this work, we tackle both the homogeneous and heterogeneous problems. We consider the semi-supervised learning case: we utilize the labelled samples in source domain for domain adaptation, when a few labelled samples in the target domain are used to support the adaption. As shown in [29] and [30] that the probability divergence of the distributions between the two domains plays an important role in domain adaptation. Our work follows the same principle of reducing distribution discrepancy and preserving the neighborhood structure in the sample projections, but adopt three strategies that are different from that commonly used in the literature.

1. We use the covariance discrepancy to define the distribution discrepancy of the projected samples. The commonly used mean discrepancy is automatically minimized in the covariance discrepancy.
2. We highlight the discrimination information of the labelled samples, which takes into account the class connections between source samples and target samples only.
3. We adopt the normalized Laplacian graph for matching the latent spectral structure in the projected samples, which is slightly different from the laplacian graph that is commonly used in the homogenous domain adaptation. This modification can weight the connection of small classes.
4. We further consider two kernel projections of the samples to increase the ability of covariance matching, together with the above coherence.

Taking into account of the above strategies together, we propose a semi-supervised domain adaptation method, named as Domain Adaptation by Covariance Matching (DACoM). The DACoM is effective and efficient in many numerical examples that we tested including image datasets, cross-language text datasets, and cross-platform gene expression datasets, for both homogeneous and heterogeneous cases. We will report the efficiency of DACoM on multiple real-world data sets from applications.

The remainder of this paper is organized as follows. Section 2 summarizes partial related works including some homogeneous and heterogeneous approaches. In Section 3, we formalize the semi-supervised domain adaptation problem and propose our method DACoM. Section 4 contains two independent contents: (1) KKT condition of DACoM and its convergence to local optimality, and (2) kernel version of DACoM. The optimization algorithm and its convergence analysis are given in Section 5. We also analyze the generalization bound in Section 6. In Section 7, we evaluate our DACoM model by comparing it with several baseline methods on eight synthetic datasets and ten real datasets. Convergence analysis, parameter sensitivity and time analysis of DACoM by experiments are also given in this section. Finally, concluding remarks and future work are briefly discussed in Section 8.

2 Related Work

In this section, we briefly discuss two homogeneous approaches Transfer component analysis (TCA) [25] and ARTL [21], and five heterogeneous approaches HeMap [27], DAMA [23], Arc-t [3], HFA [7], and SSKM [20].

TCA looks for a kernel projection $\tilde{x}_k = P^T K e_k$ for all the samples to have the small mean discrepancy of $\{\tilde{x}_i^s\}$ and $\{\tilde{x}_j^t\}$ as MMDE, where K is a known kernel matrix of the two sample sets. This simplifies the kernel construction as $\tilde{K} = KPP^TK$, rather than learning a kernel matrix $\tilde{K}$ in MMDE. It also requires that the projected vector $\{\tilde{x}_k\}$ can match the neighborhood graph W, i.e., minimize

$$g(\{\tilde{x}_k\}, W) = \sum_{ij} w_{ij} \|\tilde{x}_i - \tilde{x}_j\|^2, \tag{1}$$

subjected to $(KP)^T(H\tilde{K}_yH)(KP) = I$, where $\tilde{K}_y$ is the label graph with $\tilde{k}_{ij} = 1$ if $y_i = y_j$ or $\tilde{k}_{ij} = 0$ otherwise. The restriction imposes the label dependence on the projection.

ARTL aims to learn a label-prediction function $\tilde{x}_k = \phi(x_k)$ in a RKHS space directly that takes into account the marginal distribution and conditional distribution: small mean discrepancy of $\{\tilde{x}_i^s\}$ and $\{\tilde{x}_j^t\}$ and also small class mean discrepancy between the two domains. As in TCA, $\{\tilde{x}_k\}$ is also required to match the neighborhood graph W.

HeMap is a heterogeneous method. It looks for the representations $x_i^s \approx P_s c_i^s$ and $x_j^t \approx P_t c_j^t$ for the samples with suitable basis matrices P_s and P_t such that their coordinates $\{c_i^s\}$ and $\{c_j^t\}$ are approximately equal to each other in the sense that we also have $x_i^s \approx P_s c_i^t$ and $x_j^t \approx P_t c_j^s$. Obviously, it requires equal samples in the two domains—this can be done by repeating copies, and $\{x_i^s\}$ and $\{x_i^t\}$ are well ordered to match with each others in each class. HFA, DAMA, SSKM, and Arc-t are a semi-supervised heterogeneous approach. HFA optimizes two linear projections $\{\tilde{x}_i^s = P_s x_i^s\}$ and $\{\tilde{x}_j^t = P_t x_j^t\}$ such that the extended vectors $\left(x_i^{sT}, \tilde{x}_i^{sT}, 0_{d_t}^T\right)$ and $(0_{d_s}^T, \tilde{x}_i^{tT}, x_j^{tT})$, together with the class labels of the original samples, can be well separated by the classical SVM. Here, the projections $\{\tilde{x}_i^s, \tilde{x}_j^t\}$ play the role of hinging the heterogeneous domains.

DAMA focuses on manifold alignment—the linear projections $\tilde{x}_i^s$ and $\tilde{x}_j^t$ are optimized to achieve the three goals: $\{\tilde{x}_i^s\}$ and $\{\tilde{x}_j^t\}$ match the neighborhood graphs W_s and W_t in their own domains, respectively, and $\{\tilde{x}_k\}$ match instances with the same labels $W^{(w)}$ and separate instances with different labels $W^{(b)}$. Indeed, DAMA minimizes the ratio function

$$\frac{g\big(\{\tilde{x}_i^s\}, W_s\big) + g\big(\{\tilde{x}_i^t\}, W_t\big) + g\big(\{\tilde{x}_k\}, W^{(w)}\big)}{g\big(\{\tilde{x}_k\}, W^{(b)}\big)},$$

with the matching function g in (1), which results in a nonlinear optimization problem.

SSKM seems a bit complicated. Basically, it wants to learn a matrix $M \in \{0,1\}^{n_t \times n_s}$ to match the two RKHS's and link the labels between the two domains:

$$M\boldsymbol{\phi}_s = \boldsymbol{\phi}_t, \quad My^s = y^t.$$

Meanwhile, it also learns a prediction function in the RKHS in the source domain for classification:

$$f_s(x_i^s) = \sum_j \langle \boldsymbol{\phi}_s(x_i^s), \boldsymbol{\phi}_s(x_j^s)\rangle \alpha_j = \sum_j K_{ij}^s \alpha_j.$$

That is, $f_s = K^s\alpha$. Because the link $M\boldsymbol{\phi}_s = \boldsymbol{\phi}_t$, a prediction function f_t in the target domain is also implicitly defined as

$$\begin{aligned} f_t(x_i^t) &= \sum_j \langle \boldsymbol{\phi}_t(x_i^t), \boldsymbol{\phi}_t(x_j^t)\rangle \beta_j \\ &= \sum_j \left\langle \sum_\ell m_{i\ell}\boldsymbol{\phi}_s(x_\ell^s), \sum_{\ell'} m_{j\ell'}\boldsymbol{\phi}_s(x_{\ell'}^s) \right\rangle \beta_j \\ &= \sum_\ell \sum_{\ell'} m_{i\ell} K_{\ell,\ell'}^s \sum_j m_{j\ell'}\beta_j = e_i^T M f_s, \end{aligned}$$

that is, $f_t = Mf_s$, if we set $\alpha = M^T\beta$. The prediction f_s, or equivalently, the combination coefficients are optimized such that $f_s(x_i^s) \approx y_i^s$ and $e_j^T M f_s \approx y_j^t$.

Arc-t aims to learn an asymmetric kernel transformation for transforming feature knowledge between two domains, by performing nonlinear metric learning with similarity and dissimilarity constraints.

3 DACoM Method

3.1 Problem Statement

Suppose we are given two sets of labeled samples $\{(x_i^s, y_i^s)|_{i=1}^{n_s}\}$ and $\{(x_i^t, y_i^t)|_{i=1}^{n_t'}\}$ from the source space and target domain, respective, for training. Meanwhile, we also have a set of unlabelled samples $\{x_{n_t'+1}^t, \ldots, x_{n_t}^t\}$ from the target domain, and we should predict labels for them. Generally, the dimension p_s of source samples is different from the dimension p_t of target samples, y_i^s and $y_i^t \in \{-1, 1\}$ or have multiple class labels, and the number of labelled target samples n_t' is small. For simplicity, let

$$X_s = [x_1^s, \ldots, x_{n_s}^s], \quad X_t = [x_1^t, \ldots, x_{n_t'}^t, \ldots, x_{n_t}^t],$$

and let the centered matrices of X_s and X_t be

$$\bar{X}_s = X_s - \bar{x}_s\mathbf{1}^T, \quad \bar{X}_t = X_t - \bar{x}_t\mathbf{1}^T,$$

where $\bar{x}_s$ and $\bar{x}_t$ are the mean of all the columns of X_s and X_t, respectively, $\mathbf{1}$ represents a column vector with all ones. Similar notation will be also used for other matrices such as Z_s and Z_t mentioned latter.

3.2 Motivation

Since we have a small number of labelled samples in the target domain, the class information of the target domain is deficient for assign the other unlabelled samples if the domain is relatively "dense'. To transform the heterogenous information from the source domain to the target domain, we also consider the projections of samples from the two heterogenous domains into a common low-dimensional space $\mathcal{C} \subset \mathbb{R}^d$.

$$z_i^s = \boldsymbol{\phi}_s(x_i^s) \in \mathbb{R}^d, \quad z_i^t = \boldsymbol{\phi}_t(x_i^t) \in \mathbb{R}^d.$$

In this work, we consider the affine transformations

$$z_i^s = P_s^T(x_i^s - \bar{x}_s) \in \mathbb{R}^d, \quad z_i^t = P_t^T(x_i^t - \bar{x}_t) \in \mathbb{R}^d,$$

where $P_s \in \mathbb{R}^{p_s \times d}$, $P_t \in \mathbb{R}^{p_t \times d}$. Let Z_s and Z_t be the matrices of these projected samples of the two domains

$$\begin{aligned} Z_s &= [z_1^s, \ldots, z_{n_s}^s] = P_s^T\bar{X}_s \in \mathbb{R}^{d \times n_s}, \\ Z_t &= [z_1^t, \ldots, z_{n_t}^t] = P_t^T\bar{X}_t \in \mathbb{R}^{d \times n_t}. \end{aligned}$$

Obviously, both Z_s and Z_t has zero column mean, and hence, $\bar{Z}_s = Z_s$ and $\bar{Z}_t = Z_t$.

For simplicity, hereafter we always assume that the samples X_s and X_t have been centered such that $\bar{x}_s = 0$ and $\bar{x}_t = 0$. So, we also have $\bar{X}_s = X_s$ and $\bar{X}_t = X_t$.

Different from the other heterogeneous methods, we hope the projected samples can obey the inside distribution coherence in the following three aspects.

(C1). *Domain adaptation*. The projected sample sets have approximately equal higher order moments.
(C2). *Latent spectral structure*. The latent spectral structure of samples in each domain can be preserved in the projections.
(C3). *Consistent discriminative information*. The projected sample also contain the consistent discriminative information between the two domains as the labelled samples.

Below we describe how to model the above three coherences for the projected samples one by one.

Domain Adaptation. Higher order moments are good measurements for characterize the distribution of samples, say $\{z_k\}$. The first two moments are defined by the mean and the covariance of the samples

$$\bar{z} = \frac{1}{n}\sum_i z_i, \quad \frac{1}{n}\sum_i (z_i - \bar{z})(z_i - \bar{z})^T = \frac{1}{n}\bar{Z}\bar{Z}^T.$$

For some distributions of two samples sets, such as normal distributions, the more consistences of the two moments, the more conform their distributions are. Hence, we ask for similar moments of the projected sample sets $\{z_i^s\}$ and and $\{z_j^t\}$. Because the structures we have taken for the projections, the first moment $\bar{z}_s$ and $\bar{z}_t$ are automatically equal. So we want the two sets to have the second moments as equal as possible by minimizing the gap of their covariance matrices

$$\mathrm{Dist}(Z_s, Z_t) = \left\| \frac{Z_s Z_s^T}{n_s} - \frac{Z_t Z_t^T}{n_t} \right\|_F^2. \tag{2}$$

Different from MMDE that considers the mean discrepancy (the first order moment) only, our model can learn

more uniform distributions of the projected samples because the covariances are matched with each other. This approach is also flexible—it can handle the homogeneous information or matchable heterogeneous information. Combining kernel technique, it can also handle more complicated domains since kernel transformation can reduce the difference.

Latent Spectral Structure. Given a set of column vectors $\{x_i\}$, graph matching minimizes the function

$$\sum_{ij} w_{ij}\|x_i - x_j\|^2 = 2\mathrm{tr}(XLX^T)$$

where $X = [x_1, \ldots, x_n]$ and $L = D - W$ is the Laplacian matrix of W with diagonal matrix D of the column sum of W. In our model, we consider the latent spectral structure of the normalized Laplacian $\tilde{L} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$. That is, we minimize $\sum_{ij} w_{ij}\|\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}}\|^2 = 2\mathrm{tr}(X\tilde{L}X^T)$ for both $\{z_i^s\}$ and $\{z_j^t\}$. Let $\tilde{L}_s$ and $\tilde{L}_t$ be the two normalized Laplacian matrices of the source samples and target samples, respectively. We can preserve the data structures in the projected samples Z_s and Z_t, by minimizing

$$g_{\mathrm{structure}}(Z_s, Z_t) = \frac{\mathrm{tr}(Z_s\tilde{L}_sZ_s^T)}{n_s^2} + \frac{\mathrm{tr}(Z_t\tilde{L}_tZ_t^T)}{n_t^2}.$$

Discriminative Information. To highlight the transfer of label information from source domain to the target domain, we extract the discriminative information between the two domains only

$$w_{ij}^{(w)} = \begin{cases} 1, & \text{if } x_i^s \text{ and } x_j^t \text{ have the same label;} \\ 0, & \text{otherwise.} \end{cases}$$
$$w_{ij}^{(b)} = \begin{cases} 1, & \text{if } x_i^s \text{ and } x_j^t \text{ have different labels;} \\ 0, & \text{otherwise.} \end{cases}$$

These two label graphs are quite different to that in DAMA since we ignore the class connection in the same domain. We hope the projections can match these discriminative information. That is, we need to minimize the cost function

$$g_{\mathrm{class}}^{(w)}(Z_s, Z_t) = \frac{1}{n_sn_t'}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t'} w_{ij}^{(w)}\|z_i^s - z_j^t\|_2^2.$$

to have close projections for samples within classes, and meanwhile, to maximize the

$$g_{\mathrm{class}}^{(b)}(Z_s, Z_t) = \frac{1}{n_sn_t'}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t'} w_{ij}^{(b)}\|z_i^s - z_j^t\|_2^2,$$

to have separate projections for samples from different classes.

3.3 DACoM Model

Now we are ready to propose our new method for domain adaptation, combining the above three techniques for the purposes (C1), (C2), and (C3).

$$\begin{aligned} \min_{Z_s, Z_tP_s, P_t} & \Big\{\mathrm{Dist}(Z_s, Z_t) + \alpha g_{\mathrm{structure}}(Z_s, Z_t) \\ & + \beta g_{\mathrm{class}}^{(w)}(Z_s, Z_t) - \gamma g_{\mathrm{class}}^{(b)}(Z_s, Z_t)\Big\} \\ \text{s.t. } & Z_s = P_s^TX_s,\ Z_t = P_t^TX_t,\ P_s^TP_s + P_t^TP_t = I_d. \end{aligned} \tag{3}$$

The three parameters α, β, and γ are used for tuning the contributions of the related terms. Here, we impose an orthogonal restriction on the two linear projectors P_s and P_t to avoid naive zero solution. We name the model as DACoM.

3.4 Optimization Problem

The optimization problem (3) is quartic and non-convex. To solve this problem, we represent it in a simple form. Let

$$P = \begin{bmatrix} P_s \\ P_t \end{bmatrix} \in \mathbb{R}^{p\times d}, \quad Z = [Z_s,\ Z_t] \in \mathbb{R}^{d\times n},$$

and let $X = \mathrm{diag}(X_s, X_t) \in \mathbb{R}^{p\times n}$ be the block diagonal matrix of X_s and X_t, where $n = n_s + n_t$ and $p = p_s + p_t$. The constraints in (3) become

$$Z = P^TX, \quad P^TP = I_d.$$

Obviously, $\mathrm{Dist}(Z_s, Z_t) = \|P^TAP\|_F^2$, where

$$A = \mathrm{diag}(X_sX_s^T/n_s, -X_tX_t^T/n_t). \tag{4}$$

The second term of the objective function in (3) is

$$g_{\mathrm{structure}}(Z_s, Z_t) = \mathrm{tr}(ZLZ^T) = \mathrm{tr}(P^TXLX^TP),$$

where $L = \mathrm{diag}(\frac{1}{n_s^2}\tilde{L}_s, \frac{1}{n_t^2}\tilde{L}_t)$ is the block diagonal matrix. The last two terms deal with the cross term of Z_s and Z_t. Let $L^{(w)} = D^{(w)} - W^{(w)}$, $L^{(b)} = D^{(b)} - W^{(b)}$, where

$$W^{(w)} = \frac{1}{2n_sn_t'}\begin{bmatrix} \mathbf{0} & W_{st}^{(w)} \\ W_{st}^{(w)T} & \mathbf{0} \end{bmatrix},$$
$$W^{(b)} = \frac{1}{2n_sn_t'}\begin{bmatrix} \mathbf{0} & W_{st}^{(b)} \\ W_{st}^{(b)T} & \mathbf{0} \end{bmatrix},$$

and $D^{(w)}$ and $D^{(b)}$ are the diagonal matrices of the column sum of $W^{(w)}$ and $W^{(b)}$, respectively, as D. The last two terms in (3) can be rewritten as

$$g_{\mathrm{class}}^{(w)}(Z_s, Z_t) = \mathrm{tr}(ZL^{(w)}Z^T) = \mathrm{tr}(P^TXL^{(w)}X^TP),$$
$$g_{\mathrm{class}}^{(b)}(Z_s, Z_t) = \mathrm{tr}(ZL^{(b)}Z^T) = \mathrm{tr}(P^TXL^{(b)}X^TP).$$

Hence, letting

$$B = X(\alpha L + \beta L^{(w)} - \gamma L^{(b)})X^T, \tag{5}$$

the sum of the last three terms of the objective function is $\mathrm{tr}(P^TBP)$. Therefore, the DACoM model can be rewritten in the simple form

$$\begin{aligned} \min_{P\in R^{p\times d}} & \|P^TAP\|_F^2 + \mathrm{tr}(P^TBP) \\ \text{s.t.} \quad & P^TP = I. \end{aligned} \tag{6}$$

Since adding a bI_n with a scale b to B does not change the solution but can make B to be positive definite if b is large

enough, we can assume that B is positive definite, without loss of generality. Notice that A is positive semi-definite.

4 Optimality and Kernel Generalization

Let $f(P)$ be the objective function of (6),

$$f(P) = \|P^T AP\|_F^2 + \mathrm{tr}(P^T BP).$$

It is clearly that $f(PG) = f(P)$ for any orthogonal matrix G of order d, which means that the solution for the problem (6) is not unique. Hence, when we talk about KKT conditions of problem or convergence of the algorithm given later for solving this the problem (6), it is not necessary to ditinguish P and PG with an orhtogonal matrix of order d.

4.1 KKT Conditions

It is easy to verify that a stationary point P of $f(P)$ should be a matrix of d eigenvectors of $M(P)$, where

$$M(P) = APP^T A + \frac{1}{2}B.$$

That is, P satisfies the equations

$$M(P)P = P\Lambda, \quad P^T P = I,$$

where Λ is a symmetric matrix of order d whose eigenvalues are also the eigenvalues of $M(P)$. For convenience, let

$$\lambda_1(M(P)) \leq \cdots \leq \lambda_p(M(P)).$$

be the eigenvalues of $M(P)$ in a nondecreasing order. Hence, Λ has the eigenvalue decomposition $\Lambda = G\mathrm{diag}(\lambda_{i_1}, \ldots, \lambda_{i_d})G^T$ with an orthogonal matrix G. Replacing P by $\hat{P} = PG$, the above equations hold for $\hat{P}$ and $\hat{\Lambda} = \mathrm{diag}(\lambda_{i_1}, \ldots, \lambda_{i_d})$. Therefore, hereafter we always assume that the matrix Λ is a diagonal matrix of some eigenvalues of $M(P)$.

Recall that we want to minimize $f(P)$, and thus it is natural to guess that these eigenvalues should be the smallest ones of $M(P)$, and the KKT condition of (6) is

$$\begin{cases} M(P)P = P\Lambda, \quad P^T P = I, \\ \Lambda = \mathrm{diag}\big(\lambda_1(M(P)), \ldots, \lambda_d(M(P))\big). \end{cases} \tag{7}$$

The following theorem shows that a point P_* satisfies the KKT condition above must be at least a minimizer of $f(P)$ if the gap

$$\delta_d^* = \lambda_{d+1}(M(P_*)) - \lambda_d(M(P_*)),$$

is not small for a P_* satisfying the KKT condition.

Theorem 4.1. *If P_* satisfies the KKT condition (7) and $\delta_d^* \geq 2\|A\|_2^2$, then P_* is a local minimizer of the problem (6).*

The proof of above theorem can be found in the Appendix A.1 of the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2018.2866846.

4.2 Kernelization

The DACoM implicitly assumes that the covariances of the source/target samples are matchable under linear transformation. If the covariances are not linear matchable, nonlinear projections should be considered to increase the matchability. In this subsection, we consider a kernel version of DACoM that can be referred as the DACoM on the samples projected into RKHS spaces.

Assume that we have two kernels K_s and K_t, and corresponding to them, we have two nonlinear transformations $\tilde{x}_i^s = \phi_s(x_i^s)$ and $\tilde{x}_j^t = \phi_t(x_j^t)$ in two higher dimensional spaces. The linear projection in form $\tilde{P}_s^T \tilde{x}_i^s$ become $\int p_s(x)k^{(s)}(x, x_i^s)dx \approx \sum_\ell p_\ell k_{\ell i} = P_s^T K_s e_i$. Hence, the matrices of the centered projections are

$$Z^s = P_s^T K_s H, \quad Z^t = P_t^T K_t H,$$

where $H_s = I_s - \frac{1}{n_s}\mathbf{1}_s\mathbf{1}_s^T$ and $H_t = I_t - \frac{1}{n_t}\mathbf{1}_t\mathbf{1}_t^T$ are two centered matrices of order n_s and n_t, respectively. The required $P = \begin{bmatrix} P_s \\ P_t \end{bmatrix}$ is then the solution of the algorithm DACoM with the inputs

$$\tilde{A} = \mathrm{diag}(K_s H_s K_s^T/n_s, -K_t H_t K_t^T/n_t),$$
$$\tilde{B} = K(\alpha L + \beta L^{(w)} - \gamma L^{(b)})K^T,$$

where $K = \mathrm{diag}(K_s H_s, K_t H_t)$.

The kernel method can help to handle more complicated adaptation of heterogeneous domains. If the target data have more heterogeneous features, taking suitable kernels for two domains, the kernel version of DACoM may work better than the original one. Besides, the kernel strategy has another obvious benefit for computation since the order of the matrix $\tilde{A}$ or $\tilde{B}$ is only n that is much smaller than the matrix order p of the original A or B, when the number of samples is smaller than that of features. If both the numbers of samples and features are very large, this strategy makes it possible to train the projections based on a small set of samples first, and then obtain the new representations for all the samples. We call the kernel version DACoM as KDACoM.

5 Algorithm

By Theorem 4.1, a local optimal solution of (6) can be obtained by solving the Equation (7) which is a nonlinear eigenvalue problem. In this section, we give an iterative algorithm for solving the KKT conditions (7). The convergence of the algorithm will be also discussed in details.

5.1 Eigen-Updating Algorithm

Note that solving Equations (7) is a nonlinear eigenvalue problem. However, fixing P_*, the KKT conditions (7) are equivalent to the linear eigenvalue problem

$$\min_{Q^T Q = I_d} \mathrm{tr}\big(Q^T M(P_*)Q\big).$$

Hence, we propose the following simple iterative procedure:

$$P_{k+1} = \arg\min_{Q^T Q = I} tr(Q^T M(P_k)Q). \tag{8}$$

That is, we first compute $M_k = AP_k P_k^T A + \frac{1}{2}B$ given a current P_k, and then update P_k by P_{k+1} that consists of d

eigenvectors of M_k corresponding to its first d smallest eigenvalues, which yields

$$M_k P_{k+1} = P_{k+1}\Lambda_k, \quad \Lambda_k = \mathrm{diag}(\lambda_1^{(k)}, \ldots, \lambda_d^{(k)}).$$

The details of this iterative algorithm DACoM are given in the algorithm box.

5.2 Preprocessing for Large-Scale Problems

The computational complexity depends on the sum of features $p = p_s + p_t$ and the number of total samples $n = n_s + n_t$. It is required to reduce the data scale with a large p or n before applying DACoM. Here we give two kinds of preprocessing to reduce the feature scale p and the sample scale n, respectively, without decreasing the efficiency of DACoM.

It is simple to reduce the feature scale when $p_s \geq n_s$ or $p_t \geq n_t$. For example, if $p_s \gg n_s$, we take the QR decomposition $X^s = Q_s\hat{X}^s$ with $\hat{X}^s \in \mathbb{R}^{n_s \times n_s}$ and $Q_s \in \mathbb{R}^{p_s \times n_s}$ is an orthonormal matrix. Then P_s should be set as $P_s = Q_s\hat{P}_s$ with $\hat{P}_s \in \mathbb{R}^{n_s \times d}$. What we want to learn is $\hat{P}_s$ in a smaller scale since $P_s^T X^s = \hat{P}_s^T \hat{X}^s$. Thus, under this preprocessing on both X^s and X^t, we can reduce the scale of A and B to be the order n, rather than the original p. Therefore, we can assume that $p_s \leq n_s$ and $p_t \leq n_t$, without loss of generality.

When the sample scale is large, we use n_l landmark points from the souce damain and target domain as the samples to train the projection matrices P_s and P_t in DACoM, where n_l can be smaller than n_s or n_t. Thus, the sample scale can be reduced from $n = n_s + n_t$ to $2n_l$. To guarantee the efficiency of DACoM, the landmark points should have the same probability distributions as the samples in the two domains approximately. To this end, one can randomly choose n_l samples from the original sample set in the souce or target domain as the landmark points if n_l is not very small. We call it as *random sampling*. When n_l is small, a better way is to reconstruct landmark points in the source domain via subspace iteration as follows. Let X_c is the sample matrix in class c of the souce samples. We randomly choose a small number of columns X_c' of X_c and get an orthonormal matrix U_c by the QR factorization of X_c' initially. Then we compute an orthonormal matrix V_c via QR factorization of $X_c^T U_c$. It is then used to update U_c by the orthonormal factor of $X_c V_c$. Repeating this iteration for seveal times, we can get a good approximation $U_c H_c V_c^T$ of X_c with $H_c = U_c^T X_c V_c$. Hence, randomly choosing several rows of V_c to form $\tilde{V}_c$, we get $\tilde{X}_c = U_c H_c \tilde{V}_c^T$ whose columns can be used as lamdmark points from this class. All these construncted sets $\{\tilde{X}_c\}$ form the landmark points in the source domain. We call the approach as *QR sampling*.

5.3 Computational Complexity

The main cost of the DACoM is to compute the d eigenvectors of M (or B initially) corresponding to its d smallest eigenvalues. If n is not large, the classical eigen-solver can be used to solve P_{k+1} in a computation cost $O(n^3)$. For a middle large n, we can compute the eigenvectors via inverse iteration. That is, starting with an initial guess P, solve the equation with $M = M_k$

$$M^{-1}P = \hat{P}R, \tag{9}$$

for an orthonormal matrix $\hat{P}$ of d columns and a upper triangular matrix R of oder d. It is not required to compute the inverse matrix as shown below. If we can previously compute $U = M^{-1}P = [u_1, \ldots, u_d]$, then (9) is a classical QR factorization of U: $r_{11} = \|u_1\|_2$, $\hat{p}_1 = u_1/r_{11}$, and

$$r_{\ell\ell}\hat{p}_\ell = u_\ell - \sum_{j=1}^{\ell-1} r_{j\ell}\hat{p}_j =: v_\ell, \quad \text{with} \quad r_{j\ell} = \hat{p}_j^T u_\ell. \tag{10}$$

$r_{\ell\ell} = \|v_\ell\|_2$ and $\hat{p}_\ell = v_\ell/r_{\ell\ell}$, for $\ell = 2, \ldots, d$. It costs $O(d^2 n)$ plussing the cost of solving the linear system $MU = P$. Because of the positive definition of M, we can solve the linear system iteratively by Conjugated Gradients (CG) method, staring at the current estimate P. Because we have a good initial guess for CG, the iteration should converge quickly, especially when the outer iteration of P_k approximately converges. The cost of CG is about $O(kn^2)$ if we use k iterations of CG.

The other cost is the evaluation of objective function $f(P) = \|P^T AP\|_F^2 + \mathrm{tr}(P^T BP)$ given P. Because of the block diagonal structure of A,

$$\|P^T AP\|_F^2 = \|P_s^T X_s X_s^T P_s/n_s - P_t^T X_t X_t^T P_t/n_t\|_F^2.$$

The costs for $\hat{P}_s^T = P_s^T X_s$ is $O(dp_s n_s)$, and for $\hat{P}_t^T = P_t^T X_t$, it is $O(dp_t n_t)$. So, the total cost is only

$$O\big(d^2 n + d(p_s n_s + p_t n_t)\big) = O\big(d(d + p_{\max})n\big),$$

where $p_{\max} = \max(p_s, p_t)$. Writing $B = \begin{pmatrix} B_{ss} & B_{st} \\ B_{ts} & B_{tt} \end{pmatrix}$, we get $\mathrm{tr}(P^T BP) = \mathrm{tr}(P_s^T B_{ss} P_s) + \mathrm{tr}(P_t^T B_{tt} P_t) + 2\mathrm{tr}(P_s^T B_{st} P_t)$, where the blocks are

$$\begin{aligned} P_s^T B_{ss} P_s &= \rho\bar{p}_s\bar{p}_s^T + \hat{P}_s^T\big(\alpha L_s + \beta D_s^{(w)} - \gamma D_s^{(b)}\big)\hat{P}_s, \\ P_t^T B_{tt} P_t &= \rho\bar{p}_t\bar{p}_t^T + \hat{P}_t^T\big(\alpha L_t + \beta D_t^{(w)} - \gamma D_t^{(b)}\big)\hat{P}_t, \\ 2P_s^T B_{st} P_t &= -2\rho\bar{p}_s\bar{p}_t^T - \hat{P}_s^T\big(\beta W_{st}^{(w)} + \gamma W_{st}^{(w)}\big)\hat{P}_t, \end{aligned}$$

where $\hat{p}_s$ is the mean of columns in P_s^T, and so does $\hat{p}_t$. Furthermore, we can rewrite them as $\hat{p}_s = (\hat{p}_s^+ + \hat{p}_s^-)/n_s$ and $\hat{p}_t = (\hat{p}_t^+ + \hat{p}_t^- + \hat{p}_t^0)/n_t$, where $\hat{p}_s^+$ or $\hat{p}_s^-$ are the sums of columns of P_s^T corresponding to label $+1$ or -1, respectively, and it is similar for $\hat{p}_t^+$ or $\hat{p}_t^-$, but $\hat{p}_t^0$ is the sums of those unlabelled columns of P_s^T. Thus, taking into account the special structures of $W_{st}^{(w)}$ and $W_{st}^{(b)}$,

$$\begin{aligned} \mathrm{tr}\big(\hat{P}_s^T W_{st}^{(w)} \hat{P}_t\big) &= \langle \hat{p}_s^+, \hat{p}_t^+ \rangle + \langle \hat{p}_s^-, \hat{p}_t^- \rangle, \\ \mathrm{tr}\big(\hat{P}_s^T W_{st}^{(b)} \hat{P}_t\big) &= \langle \hat{p}_s^+, \hat{p}_t^- \rangle + \langle \hat{p}_s^-, \hat{p}_t^+ \rangle. \end{aligned}$$

whose cost is ignorable. Therefore, the cost of computing $\mathrm{tr}(P^T BP)$ is about

$$dn_s(2 + n_s) + dn_t(2 + n_t) = d(2n + n_s^2 + n_t^2).$$

The overall computational complexity of the DACoM algorithm is thus around O(n^2) without the landmark strategies. The complexity could be further reduced to O(n_l^2) by landmark strategies if $n = n_s + n_t$ is very large.

5.4 Convergence Analysis

Because $f(PG) = f(P)$ for any orthogonal matrix of order d, it is reasonable to take P and PG as the same point for any

orthogonal matrix of order d. Hence, we say the iteration sequence $\{P_k\}$ converges to P_* if any accumulation point of the boundary $\{P_k\}$ can be represented as P_*G with an orthogonal matrix G of order d.

Algorithm 1. Algorithm DACoM

Inputs. $X_s, X_t, y_s, y_t, \alpha, \beta, \gamma, d, T_{\max}, \tau$
Outputs. P, Z
1. Compute $H, E, L, L^{(w)}, L^{(b)}$.
2. Compute A and B.
3. Initially set $f_{old} = 0$ and let P be the matrix of d eigenvectors of B corresponding to the d smallest eigenvalues.
4. Repeat at most $T_{\max}$ iterations:
5. Save $f_{old} = f$ and compute $f = \|P^T AP\|_F^2 + \mathrm{tr}(P^T BP)$. If $|f - f_{old}| < \tau f$, terminate the iteration.
6. Compute the d eigenvectors $\{q_i\}$ of $M = APP^T A + \frac{1}{2}B$ corresponding to the d smallest eigenvalues.
7. Update $P = [q_1, \ldots, q_d]$.
8. End iteration
9. Compute $Z = P^T X$.

Suppose all the eigenvalues of M_k are $\lambda_1^{(k)} \leq \cdots \leq \lambda_p^{(k)}$. Let $\delta_k = \min_{i \leq d < j} |\lambda_i^{(k-1)} - \lambda_j^{(k)}|$. The following theorem ensures a local convergence of the eigen-updating algorithm. It says that $\{P_k\}$ converges to a KKT point within an orthogonal factor. This makes sense since $f(P)$ is invariant within such an orthogonal transformation.

Theorem 5.1. *If $\delta_k \geq a\|A\|_2^2$ for a constant value $a > 1$ and sufficiently large k, then $\{P_k\}$ converges to P_* that satisfies the KKT condition (7).*

By Theorem 4.1, $\{P_k\}$ converges to at least a local minimizer of f if $\delta_* \geq 2\|A\|_2^2$. The proof of Theorem 5.1 can be found in the Appendix A.2 of the supplementary material, available in the online supplemental material.

The condition $\delta_k \geq a\|A\|_2^2$ for sufficiently large k is approximately equal to that $|\lambda_d^* - \lambda_{d+1}^*| \geq a\|A\|_2^2$. In practice, the perturbation theory on eigenvalues gives

$$|\lambda_i^{(k)} - \lambda_i^{(k-1)}| \leq \|M_k - M_{k-1}\|_2 = \|A\Delta_k A\|_2 \leq \|\Delta_k\|_2 \|A\|_2^2,$$

for each i, where $\Delta_k = P_k P_k^T - P_{k-1} P_{k-1}^T$. By

$$|\lambda_i^{(k-1)} - \lambda_j^{(k)}| \leq |\lambda_i^{(k)} - \lambda_j^{(k)}| + |\lambda_i^{(k-1)} - \lambda_i^{(k)}|.$$

we get that $\delta_k \leq |\lambda_{d+1}^{(k)} - \lambda_d^{(k)}| + \|\Delta_k\|_2 \|A\|_2^2 \to |\lambda_d^* - \lambda_{d+1}^*|$. Hence, the condition $\delta_k \leq \alpha\|A\|_2^2$ is approximately equal to that $|\lambda_d^* - \lambda_{d+1}^*| \geq a\|A\|_2^2$ for a sufficiently large k.

Although we only prove the local minimization of the algorithm, it seems that the algorithm is not very sensitive to its initial setting for the iteration. Hence, we do not carefully choose an initial point for each example in our numerical experiments. We always use the eigenvector matrix of B corresponding to the first d smallest eigenvalues as the initial P of DACoM, which works well in all the examples.

6 Generalization Bound

We adopt the error bound in [30] to explain the proposed DACoM model. Before the error bound is explained, it is necessary to introduce some notations.

Definition 6.1 ([29]). *Given a domain $\mathcal{X}$ and a collection $\mathcal{A}$ of subsets of $\mathcal{X}$. Let μ_1 and μ_2 be probability distributions over $\mathcal{X}$, and every set in $\mathcal{A}$ is measurable with respect to both distributions. The $\mathcal{A}$-distance between μ_1 and μ_2 is defined as*

$$d_{\mathcal{A}}(\mu_1, \mu_2) = 2 \sup_{A \in \mathcal{A}} |\mu_1(A) - \mu_2(A)|.$$

Note that the different choices of $\mathcal{A}$ may result in different distance measures. A typical example is the hypothesis class-specific distance measure based on symmetric difference hypothesis space $\mathcal{H}\triangle\mathcal{H}$ as follows

$$d_{\mathcal{H}\triangle\mathcal{H}} = 2 \sup_{A \in \mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}} |\mu_1(A) - \mu_2(A)|,$$

where $\mathcal{H} = \{h : \mathcal{X} \to \{0, 1\}\}$ is a binary hypothesis space for $\mathcal{X}$, and $\mathcal{A}_{\mathcal{H}\triangle\mathcal{H}}$ is the set of all sets A such that $A = \{x : x \in \mathcal{X}, h(x) \neq h'(x)\}$.

Theorem 6.1. *[30] Let $\mathcal{H}$ be a hypothesis space of VC-dimension d, and let Z_S and Z_T be unlabeled samples of size m' each, drawn from $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. With probability at least $1 - \delta$ (over the choice of samples), then for every $h \in \mathcal{H}$, the learning risks $\epsilon_T(h)$ and $\epsilon_S(h)$ in the target and source domains, respectively, satisfy*

$$\epsilon_T(h) \leq \epsilon_S(h) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(Z_S, Z_T) + \lambda + 4\sqrt{\frac{2d\log(2m') + \log(4/\delta)}{m'}},$$

where $\lambda = \min_h \epsilon_S(h) + \epsilon_T(h)$ is the combined optimal risk of both the domains, and $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}$ is the empirical distance on Z_S and Z_T.

To find the optimal projection, DACoM model (3) attempts to minimize the combined source and target risk (C3) with preserving the intrinsic structure of both domains (C2). DACoM also minimizes the difference between the source distribution and target distribution (C1). In the following theorem, we show that the covariance matching in DACoM is necessary for minimizing the difference between the two distributions.

For two continuous probability distributions μ_1 and μ_2 over $\mathcal{X}$, we use the $\mathcal{A}$-distance of

$$d_0(\mu_1, \mu_2) = 2 \sup_{A \in \mathcal{A}} |\mu_1(A) - \mu_2(A)|,$$

with $\mathcal{A}$ including all the subsets of $\mathcal{X}$. Obviously, $d_0(\mu_1, \mu_2)$ is an upper bound of $d_{\mathcal{H}\triangle\mathcal{H}}(\mu_1, \mu_2)$ defined above.

Let μ be a continuous probability distribution for a random variable, its moment of order k be $m^{(k)}$ if any, and its moment generating function be $M(s)$ if any. We define a set of probability distributions for $\alpha > 0$ as follows:

$$\mathcal{U}_\alpha = \left\{ \begin{array}{l} \mu : \mu \text{ is continuous and has a moment} \\ \text{generating function } M(s) \text{ in } N(0, \alpha) \end{array} \right\}.$$

Theorem 6.2. *Let $\alpha > 0$. Then $\forall \epsilon > 0$, $\exists \delta > 0$ such that $d_0(\mu, \mu_0) < \epsilon$ for all μ_0, $\mu \in \mathcal{U}_\alpha$ whose moments $\{m_0^{(k)}\}$ and $\{m^{(k)}\}$ of all orders satisfying $\sup_k |m^{(k)} - m_0^{(k)}| < \delta$.*

The proof of Theorem 6.2 can be found in the Appendix A.3, available in the online supplemental material. Theorem 6.2 shows that under certain conditions, the difference between

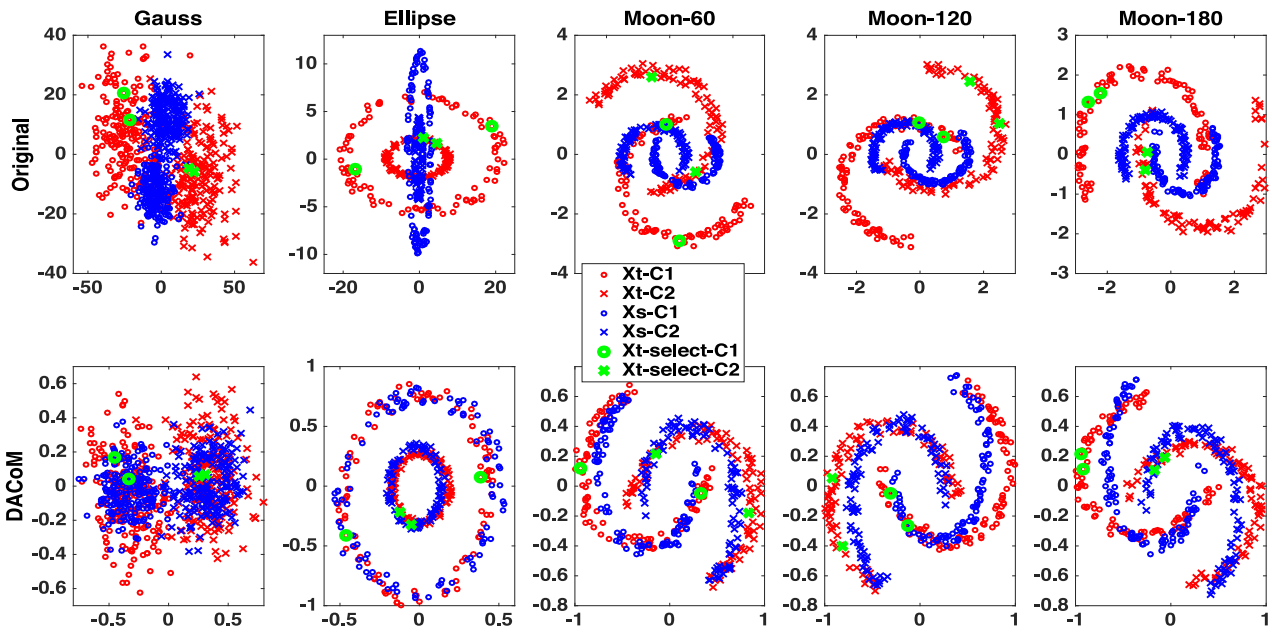

Fig. 1. The figures in the top row show the original data points for Gauss, Ellipse and three two-moon datasets, and the figures in the second row show the new representation for the samples by DACoM. Red circles and crosses represent the samples in class 1 and class 2 from target domain (Xt-C1, Xt-C2), respectively, and blue circles and crosses represent the samples in class 1 and class 2 from source domain (Xs-C1, Xs-C2), respectively. Green circles and crosses represent the target training samples of the two classes, respectively.

two distributions can be infinitely small if their moment generating functions are close enough or their moments of all orders are close enough. We can further notice that when $\alpha < 3$, an upper bound $\sum_{k=1}^{\infty} |m^{(k)} - m_0^{(k)}| \frac{\alpha^k}{k!}$ in Equation (2) of the proof (see the Appendix A.3, available in the online supplemental material) is dominated by the first two moments. Thus the term of covariance matching in DACoM is necessary to make sure the two distributions be similar.

7 Experiments

We simulated three types of synthetic datasets to compare the performance of our DACoM method with others. We also applied DACoM on several real datasets to evaluate its performance.

7.1 Synthetic Datasets

1. *Gauss Dataset.*We simulate the data samples by a mixed gaussian model in both source domain and target domain. Let

$$\mu_1 = \begin{bmatrix} -3 \\ -4 \end{bmatrix}, \mu_2 = \begin{bmatrix} -10 \\ -20 \end{bmatrix}, \Sigma = \begin{bmatrix} 40 & 2 \\ 2 & 40 \end{bmatrix}.$$

In the source domain, we generate 100 positive and 100 negative samples $x \sim N(\mu_1, \Sigma)$ and $x \sim N(\mu_2, \Sigma)$, respectively. In target domain, We first generate 100 positive and 100 negative samples using the same distributions as in the source domain, and then rotate and rescale the samples as follows,

$$x_i = 2Qx_i,$$

where $Q = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$ is a 2-by-2 Givens rotation matrix with $\theta = \frac{\pi}{2}$.

2. *Two-ellipse dataset.* We generate samples distributed as an ellipse using the function

$$F_{a,b}(\theta) = \begin{bmatrix} a\cos(\theta) + a\epsilon \\ b\sin(\theta) + b\epsilon \end{bmatrix}, \quad \theta \in (0, 2\pi),$$

where $\epsilon \sim N(0, 0.2)$. For each domain, we use two values of (a, b) to generate two ellipse-like sets and each has 100 points. That is, the samples in source domain are

$$x_i^{sp} = F_{1,4}(\theta_i^{sp}), \quad x_i^{sn} = F_{3,10}(\theta_i^{sn}), \quad i = 1, \ldots, 100.$$

While in the target domain, we have samples

$$x_i^{tp} = F_{8,2}(\theta_i^{tp}), \quad x_i^{tn} = F_{20,6}(\theta_i^{tn}), \quad i = 1, \ldots, 100.$$

Here $\theta_i^{sp}, \theta_i^{sn}, \theta_i^{tp}, \theta_i^{tn}$ are sampled from the uniform distribution $U[0, 2\pi]$ independently.

3. *Two-moon Dataset.* Similar with the generation in above example, we generate samples using the function

$$T_{r,c_1,c_2}(\theta) = r \begin{bmatrix} \cos(\theta) + \epsilon \\ \sin(\theta) + \epsilon \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix},$$

where $\epsilon \sim N(0, 0.2)$. The samples in source domain and target domain are

$$x_i^{sp} = T_{1,0,0}(\theta_i^{sp}), \quad x_i^{sn} = T_{1,1,0}(\theta_i^{sn}), \quad i = 1, \ldots, 100;$$
$$x_i^{tp} = T_{2,1,1}(\theta_i^{tp}), \quad x_i^{tn} = T_{2,3,1}(\theta_i^{tn}), \quad i = 1, \ldots, 100,$$

respectively, where $\theta_i^{sp}, \theta_i^{tp}$ are uniformly distributed in $U[-\frac{\pi}{6}, \frac{11\pi}{9}]$ independently, and $\theta_i^{sn}, \ldots, \theta_i^{tn}$ are uniformly distributed in $U[\frac{5\pi}{6}, \frac{20\pi}{9}]$ independently.

The above three types of datasets are shown in Fig. 1, where the top row shows the original datasets of Gauss dataset,

TABLE 1
Data Summaries of the Real-World Datasets

Data	WebKB				face-pose(face-eye)				Warnat		Rueters1(2,3,4)					Rueters5		Digit	
Domain	Washington	Wisconsin	Texas	Cornell	an2i	at33	boland	bmp	Bullinger	Valk	English	French	Spanish	Italian	German	French	German	MNIST	USPS
No.sample	230	265	187	195	32	32	32	32	52	97	200(1000,1200,2400) in each domain					29,953	24,039	70,000	11,000
No.features	1703	1703	1703	1703	960	960	960	960	7102	7102	1131	1230	807	1041	1417	34,249	15,506	784	256
No.class	2	2	2	2	4(2)	4(2)	4(2)	4(2)	2	2	2(2,6,6)					6	6	10	10

two-ellipse dataset, and two-moon datasets with different rotation degrees. For each dataset, we randomly chose $n_t = 2$ samples in each class from the target domain as training data. The second row shows the new representations of the data points obtained by DACoM with $\alpha = \beta = \gamma = 0.01$. We can see that DACoM could match the two domains very well for all the cases.

7.2 Real-World Datasets

We apply our DACoM method to five real applications including face pose recognition, eye recognition, webpage classification, cross-domain disease classification, and Multilingual documents classification. Our experiments include extensive applications on homogeneous datasets and heterogeneous datasets, two-class datasets and multiple-class datasets, high-dimensional datasets and low-dimensional datasets, large-sample-size datasets and small-sample-size datasets. Table 1 summarizes the data statistics for all the real-world datasets we used.

1. *Webpage classification.* The WebKB[1] dataset includes different categories of webpages collected from four universities Cornell, Texas, Washington and Wisconsin, which are considered as domains in our experiments. Each university domain is taken as the source domain and target domain once. We used the webpages which belong to "student" or "course" for our experiments and the task is to classify the webpages in the target domain to be in one of the two classes.w
2. *Face recognition.* CMU face data[2] consists of 640 black and white face images of people taken with varying pose (straight, left, right, up) and eye status (open and sunglasses). Each person is considered as a domain, and our task is to classify the images in the target domain to be four poses or two eye status. We only chose the first four persons from the whole dataset (an2i, at33, boland and bmp) for our experiments.
3. *Cross-domain disease classification.* Warnat et al. [34] compared two studies on acute myeloid leukemia (AML): Bullinger et al. [35] and Valk et al. [36]. The dataset Bullinger consists of 52 patients, and the dataset Valk of 97 patients. Both datasets share gene expression levels for n = 7102 genes. The experiments of Bullinger et al. were carried out on a cDNA platform while Valk et al. used oligonucleotide microarrays. We take Bullinger and Valk datasets as source domains and target domains in turn. The prediction task is to differentiate between cancerous and normal tissue in the target domain. Note that this dataset could be considered as heterogeneous since the features from two domains are measured in different ways.
4. *Multilingual documents classification.* The Reuters Multilingual data consists of documents written in one of the five languages (English, French, German, Spanish, and Italian) over a common set of 6 categories [37]. The original data has very high dimensional of features, and thus we first applied PCA to the dataset to reduce the dimensionality. We generate four datasets with different number of samples and categories. For Reuters1 and Reuters2, we randomly choose 200 and 1000 documents per domain, respectively, from categories 'C15' and 'M11'. For Reuters3 and Reuters4, we randomly choose 1200 and 2400 documents per domain, respectively, from all the six categories. Since there are five domains, we have 20 tasks for domain adaptation for each dataset. To test the performance of our methods on large-scale datasets, we also use the original high dimensional datasets Reuters5 with all samples in French and German domains. French domain has 34,249 features and 29,953 samples, and the German domain has 15,506 features and 24,039 samples.
5. *Digit recognition.* MNIST[3] and USPS[4] consist of images for digits from 0 to 9. The number of samples in the two datasets are 70,000 and 11,000, respectively. To recognize the digits, say MNIST, in the target domain, we use the the other set, say USPS, as source domain. So, in this example, the total sample size $n = 81{,}000$.

7.3 Baselines Methods

We compare our approaches with seven other domain adaptation approaches. Among these approaches, HeMap, DAMA, and HFA are heterogeneous domain adaptation approaches, and TCA, GFK and ARTL are homogeneous domain adaptation approaches.

1. SVMT. This method uses only the labelled samples from the target domain to train a standard SVM classification model.
2. TCA [25]
3. GFK [38]
4. ARTL [21].
5. HeMap [27].
6. DAMA [23].
7. HFA [7].

Note that SVMT can only use the labeled data from the target domain, since the source domain and target

1. http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/
2. https://archive.ics.uci.edu/ml/datasets/CMU+Face+Images
3. http://yann.lecun.com/exdb/mnist/
4. https://cs.nyu.edu/ roweis/data.html

TABLE 2
Method Comparison

domain	data	SVMT	TCA	GFK	ARTL	HeMap	DAMA	HFA	DACoM
source	X_s		✓	✓	✓	✓	✓	✓	✓
	y_s		✓	✓	✓		✓	✓	✓
target	X_t^{train}	✓	✓	✓	✓	✓	✓	✓	✓
	y_t^{train}	✓					✓	✓	✓
	X_t^{test}		✓	✓	✓	✓	✓		✓

domain may have completely different feature spaces (such as Reuters and Warnat datasets). HFA only makes use of the labeled instances with their labels from both domains. HeMap, ARTL, DAMA and DACoM exploit both labeled and unlabeled data from the two domains. HeMap is an unsupervised method, and the labels of the instances from both domains are not used. TCA, GFK and ARTL use only the labels of the training instances. DAMA and DACoM use the labels of training instances from both domains. Table 2 summarizes the data used in each of these methods.

7.4 Experimental Setup

We apply our DACoM approach and other comparison partners on all the above datasets. Once the pair of source and target domains is determined, we first preprocess X_s and X_t in the following way. For all datasets, we center X_s and X_t separately such that each row (feature) has zero mean. Specially, we normalize the Warnat datasets such that the rows also has norm 1. We then randomly choose $\tilde{n}_t$ data points per class in target domain as the target training samples. $\tilde{n}_t$ is chosen as a small integer. For simulation, Face, Webkb, Reuters1 and Reuters2 datasets, $\tilde{n}_t$ is chosen as 2. For Warnat datasets, we choose $\tilde{n}_t$ in $\{2, 4, 6, 8, 10\}$. For Reuters3 and Reuters4, $\tilde{n}_t$ is chosen as 10. After choosing the pair of domains and the target training samples, we can apply each method to the dataset and calculate the classification accuracy for the target test samples. To compute the accuracy, we apply SVM with Gaussian kernel to the computed new representations in the low dimensional space by different methods. Both the regularization parameter C and the Gaussian kernel parameter σ are chosen by two-fold cross validation from a set $\{0.01, 0.1, 1, 10, 100\}$ in our experiments if not mentioned. Note that a different set of target training samples may result in a different classification accuracy, and thus we randomly choose the target training set for $T = 20$ times. For each method we report the average accuracy and the standard error over these T runs. The classification accuracy is computed on the test data (unlabeled target data) as

$$\text{Accuracy} = \frac{|x : x \in \mathcal{D} \wedge f(x) = y(x)|}{|x : x \in \mathcal{D}|},$$

where $f(x)$ is the label predicted by the classification algorithm , $y(x)$ is the ground truth label of x, and $\mathcal{D}$ contains all the unlabeled target instances.

For our DACoM approach and other comparison methods, it is impossible to tune the optimal parameters for the target classifier by cross validation, since the number of labelled instances in the target domain is very limited. Thus we evaluate all the methods on our datasets for the optimal parameters, and report the best results of each method. For DACoM, we simply fix $\gamma = 1$, and report the best results when α and β are in the set $\{0.01, 0.1, 1, 10, 100\}$. The dimension d are chosen as 2 for simulation datasets, 3 for Warnat dataset, 5 for face dataset, and 10 for WebKB and Reuters datasets. For simulation datasets, we use DACoM since the number of features are small. We apply DACoM, KDACoM with linear, and Gaussian kernels to all the real datasets. We choose σ in Gaussian kernels for KDACoM to be the median of all the pairwise square root of the distances among samples. For SVMT, the number of training data in the target domain is too small, and thus we can not choose parameters by cross-validation. We report the best results when the parameters C and σ vary in $\{0.01, 0.1, 1, 10, 100\}$. For DAMA, we use the same parameter setting of d and γ with DACoM, and report the best results when α and β in the range of $\{0.01, 0.1, 1, 10, 100\}$. For HeMap, we report the best results when β is in the range of $\{0.01, 0.1, 1, 10, 100\}$. For ARTL, and choose $p = 10$, $\sigma = 1, \lambda = 10$ as suggested by the authors, and report the best results when γ varies in the range of $\{1, 10\}$. For HFA, we choose $C = 1$ following authors' suggestion. For TCA, we report the best results when λ is chosen from $\{10^{-3} : 10^{3}\}$. Our experiments are conducted by Matlab2015b on a workstation with OS X Yosemite, 16 GB RAM and Intel 3.1 GHz CPUs.

7.5 Classification Results

With the experimental setting in Section 7.4, we compare all the methods on the simulation datasets and the real datasets. In all the reported tables, the best results are marked in bold, and the second best results are underlined.

Table 3 shows the results for the eight simulation datasets. Among all the methods, DACoM works the best for almost all the datasets. For the Gauss data, DAMA, HFA and DACoM could achieve average classification accuracies higher than 95 percent. For Ellipse data, Both DACoM and GFK could classify the test samples in target domain with very high accuracy. For the six moon datasets, the average classification accuracies of DACoM are higher than 85 percent. It is significantly better than other methods—their accuracies are 60%-70%. GFK works very well for Moon30. However, its accuracy deceases as the rotation degree increases.

The results for the face-pose and face-eye datasets are listed in Table 4. DACoM or KDACoM work the best for 10 tasks of all the pose recognition tasks, and the second best for the other 2 tasks. TCA works well in most tasks, but for $P3 \rightarrow P1$ and $P1 \rightarrow P3$, its accuracies are relatively low. For eye recognition, DACoM/KDACoM perform the best for 8 tasks, and second best for 3 tasks. GFK seems the second promising method among all the comparison methods. We could also see from the tables that linear KDACoM seems the best among DACoM, linear KDACoM and Gaussian KDACoM. For several cases, Gaussian kernel works the best, and it might be due to the nonlinear matchable heterogeneous information between two domains. In Fig. 2, we further visualize the result of $P3 \rightarrow P1$. We apply multidimensional scaling (MDS) on the original data to visualize

TABLE 3
Means and Standard Deviations of Classification Accuracies (%) on the Simulation Datasets

Data	SVMT	TCA	GFK	ARTL	HeMap	DAMA	HFA	DACoM
Gauss	88.47 ± 1.63	66.60 ± 0.77	36.78 ± 0.21	47.03 ± 0.40	87.75 ± 1.87	95.52 ± 0.60	96.26 ± 0.27	**96.63 ± 0.23**
Ellipse	80.61 ± 0.96	61.26 ± 0.73	**100.00 ± 0.00**	56.46 ± 0.61	75.88 ± 0.60	77.80 ± 1.03	49.47 ± 0.22	99.89 ± 0.07
Moon30	79.78 ± 1.08	81.97 ± 0.12	**99.70 ± 0.04**	75.29 ± 0.09	66.22 ± 1.67	76.79 ± 0.66	69.80 ± 1.39	85.63 ± 1.73
Moon60	75.44 ± 1.00	61.61 ± 0.09	82.32 ± 0.30	64.64 ± 0.08	71.26 ± 1.46	75.07 ± 0.95	62.12 ± 1.26	**88.40 ± 1.78**
Moon90	78.88 ± 1.14	53.70 ± 1.00	53.71 ± 0.51	51.50 ± 0.03	74.94 ± 1.60	75.37 ± 1.09	68.56 ± 1.52	**87.00 ± 1.73**
Moon120	78.36 ± 0.88	50.94 ± 0.49	39.60 ± 0.70	49.73 ± 0.45	76.95 ± 1.77	77.64 ± 0.74	66.06 ± 1.56	**91.81 ± 1.37**
Moon150	78.89 ± 1.01	57.03 ± 0.14	17.41 ± 0.69	34.10 ± 0.72	70.14 ± 1.68	74.89 ± 0.72	64.59 ± 1.49	**87.05 ± 1.57**
Moon180	81.13 ± 1.01	60.70 ± 0.46	0.23 ± 0.10	17.98 ± 0.36	71.87 ± 1.89	78.94 ± 0.60	70.27 ± 1.64	**86.17 ± 1.66**

TABLE 4
Means and Standard Deviations of Classification Accuracies (%) on Face-Pose and Face-Eye Datasets

Data	Domains	SVMT	TCA	GFK	ARTL	HeMap	DAMA	HFA	DACoM	KDACoM-linear	KDACoM-Gauss
Face-Pose	P2→P1	73.80 ± 3.52	82.14 ± 0.00	75.36 ± 1.57	85.36 ± 0.36	68.04 ± 1.38	88.04 ± 1.38	78.57 ± 2.06	87.68 ± 1.52	**90.54 ± 1.22**	89.29 ± 1.04
	P3→P1	76.53 ± 3.58	78.57 ± 0.00	53.39 ± 2.10	30.36 ± 4.52	75.18 ± 1.96	86.07 ± 2.40	77.14 ± 1.50	90.36 ± 1.82	**93.93 ± 1.54**	89.64 ± 1.53
	P4→P1	72.80 ± 3.39	**89.46 ± 0.18**	70.18 ± 2.59	86.07 ± 0.36	60.18 ± 3.62	86.43 ± 1.50	75.54 ± 2.26	87.68 ± 1.75	87.86 ± 2.08	83.93 ± 1.71
	P1→P2	72.98 ± 3.63	83.21 ± 0.59	84.46 ± 1.52	70.00 ± 0.40	60.18 ± 1.96	82.14 ± 0.93	73.75 ± 1.58	82.68 ± 1.25	83.93 ± 1.63	**89.64 ± 1.42**
	P3→P2	78.13 ± 3.92	87.14 ± 0.48	60.00 ± 2.45	52.50 ± 3.45	61.43 ± 2.16	82.68 ± 0.98	77.50 ± 1.10	83.75 ± 1.33	85.00 ± 1.20	**87.68 ± 1.43**
	P4→P2	74.25 ± 3.60	**88.04 ± 0.39**	66.79 ± 1.47	85.71 ± 0.00	65.71 ± 2.06	78.75 ± 1.08	70.54 ± 1.66	79.82 ± 1.08	83.93 ± 0.88	80.36 ± 1.52
	P1→P3	78.63 ± 4.27	71.25 ± 0.18	65.54 ± 2.75	16.79 ± 1.72	86.43 ± 2.88	75.71 ± 1.41	71.43 ± 0.00	**91.43 ± 2.03**	89.11 ± 2.48	89.11 ± 1.95
	P2→P3	81.61 ± 3.63	86.96 ± 0.70	64.46 ± 2.49	52.14 ± 1.61	73.39 ± 0.88	75.00 ± 1.42	71.43 ± 0.00	91.25 ± 1.80	84.46 ± 2.26	**91.79 ± 1.92**
	P4→P3	80.68 ± 2.43	83.75 ± 0.61	67.86 ± 2.14	57.14 ± 0.00	68.93 ± 0.90	73.57 ± 1.17	71.43 ± 0.00	**86.07 ± 2.36**	80.89 ± 2.88	81.79 ± 2.99
	P1→P4	75.44 ± 2.64	**100.00 ± 0.00**	87.68 ± 1.59	**100.00 ± 0.00**	76.96 ± 1.41	**100.00 ± 0.00**	98.39 ± 0.55	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
	P2→P4	75.67 ± 2.54	**100.00 ± 0.00**	82.86 ± 2.16	92.86 ± 0.00	99.29 ± 0.33	**100.00 ± 0.00**	99.64 ± 0.36	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
	P3→P4	77.68 ± 3.10	**100.00 ± 0.00**	82.86 ± 1.97	65.54 ± 3.18	97.50 ± 0.97	**100.00 ± 0.00**	99.29 ± 0.42	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
Face-Eye	P2→P1	50.00 ± 0.00	60.00 ± 0.00	**70.17 ± 1.42**	63.33 ± 0.00	58.67 ± 1.63	55.67 ± 1.08	52.50 ± 1.58	65.67 ± 2.08	69.17 ± 1.73	69.17 ± 1.73
	P3→P1	50.00 ± 0.00	53.33 ± 0.00	69.83 ± 1.24	53.33 ± 0.00	55.17 ± 0.70	53.50 ± 0.98	53.00 ± 1.53	**72.83 ± 1.80**	62.50 ± 1.67	68.67 ± 1.67
	P4→P1	50.00 ± 0.00	66.67 ± 0.00	58.00 ± 1.97	60.00 ± 0.00	56.17 ± 0.61	57.67 ± 1.14	52.50 ± 1.60	71.33 ± 1.89	**73.00 ± 1.70**	71.50 ± 1.70
	P1→P2	50.17 ± 0.17	63.33 ± 0.00	69.17 ± 1.23	60.00 ± 0.00	61.67 ± 1.77	56.67 ± 1.47	58.33 ± 1.80	76.00 ± 2.99	**79.67 ± 2.45**	65.17 ± 2.45
	P3→P2	50.17 ± 0.17	66.67 ± 0.00	69.67 ± 1.13	50.00 ± 0.00	66.67 ± 1.83	56.67 ± 1.86	61.50 ± 2.29	**78.67 ± 2.01**	74.00 ± 2.21	66.00 ± 2.21
	P4→P2	50.17 ± 0.17	73.33 ± 0.00	81.83 ± 0.74	**83.33 ± 0.00**	59.33 ± 1.60	57.83 ± 1.78	59.50 ± 2.13	77.33 ± 2.45	78.83 ± 2.21	69.83 ± 2.21
	P1→P3	56.00 ± 0.86	73.33 ± 0.00	68.00 ± 1.70	60.00 ± 0.00	61.33 ± 1.24	71.50 ± 2.04	61.17 ± 3.42	78.50 ± 1.24	**81.17 ± 1.14**	71.67 ± 1.14
	P2→P3	56.00 ± 0.86	73.33 ± 0.00	72.83 ± 2.01	76.67 ± 0.00	74.67 ± 1.73	70.83 ± 1.69	61.00 ± 4.20	76.33 ± 1.87	**78.67 ± 1.55**	72.00 ± 1.55
	P4→P3	56.00 ± 0.86	83.33 ± 0.00	**84.83 ± 2.07**	76.67 ± 0.00	60.67 ± 0.96	70.83 ± 1.84	61.17 ± 4.03	78.50 ± 1.38	78.17 ± 1.12	76.17 ± 1.12
	P1→P4	52.83 ± 0.91	56.67 ± 0.00	43.67 ± 1.37	70.00 ± 0.00	58.33 ± 0.71	54.00 ± 1.04	51.83 ± 1.27	**77.17 ± 3.44**	72.83 ± 1.88	66.00 ± 1.88
	P2→P4	52.83 ± 0.91	66.67 ± 0.00	**78.67 ± 0.85**	63.33 ± 0.00	51.00 ± 0.84	53.50 ± 1.04	59.33 ± 2.67	76.33 ± 3.68	71.50 ± 2.45	65.83 ± 2.45
	P3→P4	52.83 ± 0.91	63.33 ± 0.00	64.17 ± 1.95	50.00 ± 0.00	54.83 ± 0.95	51.83 ± 1.31	60.17 ± 2.90	**75.83 ± 3.67**	68.50 ± 2.63	65.67 ± 2.63

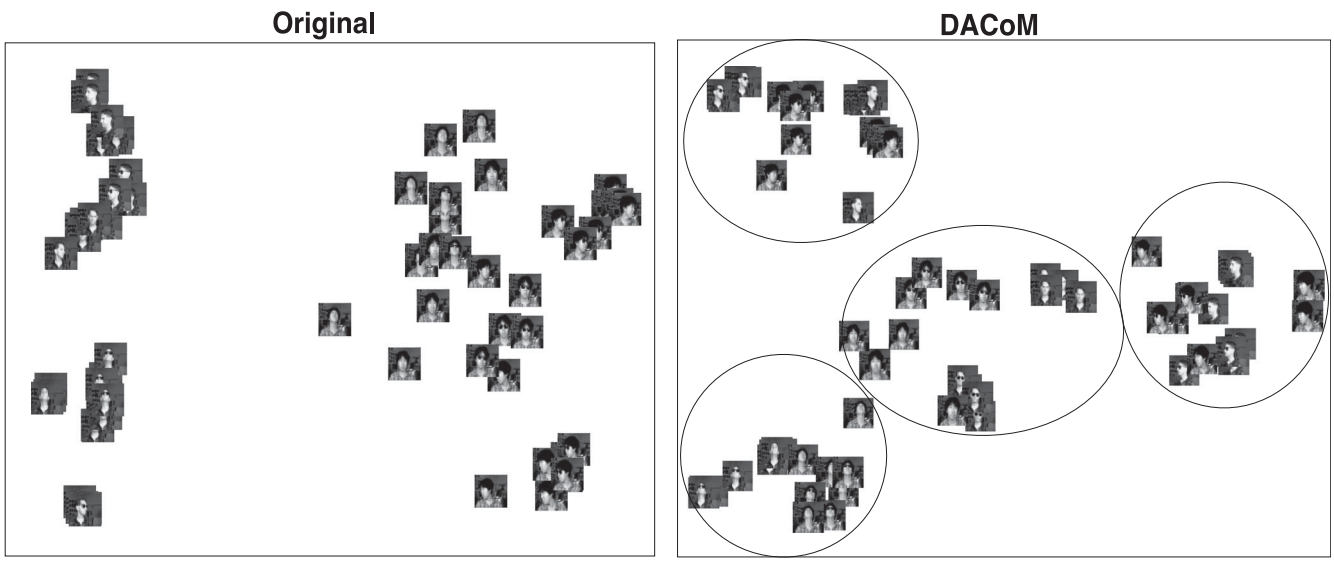

Fig. 2. The left figure shows the MDS embedding of the original images, which are clustered by their person ids(P3 or P1). The right figure shows the MDS embedding of new DACoM representations, which are clustered by their poses (up, straight, left and right).

TABLE 5
Means and Standard Deviations of Classification Accuracies (%) on the Webkb Datasets

Data	SVMT	TCA	GFK	ARTL	HeMap	DAMA	HFA	DACoM	KDACoM-linear	KDACoM-Gauss
Washington→Wisconsin	77.96 ± 1.98	86.44 ± 0.47	89.36 ± 0.24	**95.00 ± 0.07**	83.48 ± 1.58	85.90 ± 1.35	87.47 ± 1.51	91.80 ± 0.86	94.38 ± 0.61	90.52 ± 1.14
Texas→Wisconsin	77.96 ± 1.98	75.67 ± 0.50	86.08 ± 0.26	84.38 ± 0.82	75.28 ± 1.35	75.52 ± 1.41	81.73 ± 1.60	87.58 ± 0.69	**92.53 ± 0.53**	87.24 ± 0.80
Cornell→Wisconsin	77.96 ± 1.98	79.82 ± 0.39	90.75 ± 0.22	90.41 ± 0.09	86.37 ± 0.85	80.05 ± 1.47	86.83 ± 1.46	91.24 ± 0.85	**94.54 ± 0.68**	89.18 ± 0.85
Wisconsin→Washington	80.50 ± 2.45	86.51 ± 0.43	88.76 ± 0.41	94.11 ± 0.54	89.53 ± 0.66	94.91 ± 0.66	88.70 ± 1.05	**96.86 ± 0.44**	96.07 ± 0.64	93.46 ± 0.99
Texas→Washington	80.50 ± 2.45	74.70 ± 0.60	84.41 ± 0.49	81.54 ± 0.69	73.40 ± 0.89	77.84 ± 1.63	80.27 ± 1.39	89.76 ± 0.49	**95.00 ± 0.63**	90.95 ± 1.11
Cornell→Washington	80.50 ± 2.45	81.83 ± 0.46	92.60 ± 0.13	92.01 ± 0.13	88.31 ± 1.14	87.57 ± 0.99	86.36 ± 1.15	95.24 ± 0.60	**96.15 ± 0.50**	93.11 ± 0.85
Wisconsin→Texas	85.34 ± 1.43	**94.32 ± 0.16**	93.65 ± 0.29	92.41 ± 0.49	84.40 ± 1.38	90.38 ± 0.93	89.62 ± 1.14	93.65 ± 1.01	94.06 ± 0.71	90.00 ± 1.16
Washington→Texas	85.34 ± 1.43	93.23 ± 0.37	82.82 ± 0.31	92.97 ± 0.35	85.38 ± 1.41	89.89 ± 0.92	89.92 ± 1.17	**94.96 ± 0.80**	93.53 ± 1.01	88.23 ± 1.35
Cornell→Texas	85.34 ± 1.43	92.86 ± 0.38	90.56 ± 0.30	91.32 ± 0.30	85.53 ± 1.81	84.85 ± 0.63	89.77 ± 1.15	94.51 ± 0.88	**94.62 ± 0.81**	90.86 ± 1.27
Wisconsin→Cornell	77.31 ± 1.89	80.74 ± 0.52	90.08 ± 0.30	89.21 ± 0.69	83.10 ± 1.51	82.98 ± 1.43	84.42 ± 1.56	**92.64 ± 0.94**	92.60 ± 0.83	88.93 ± 1.37
Washington→Cornell	77.31 ± 1.89	84.63 ± 0.42	90.45 ± 0.55	**95.87 ± 0.00**	77.93 ± 2.16	81.32 ± 1.06	83.31 ± 1.52	90.79 ± 0.94	93.35 ± 0.89	87.81 ± 2.06
Texas→Cornell	77.31 ± 1.89	75.17 ± 0.46	**89.50 ± 0.31**	86.78 ± 0.71	76.61 ± 1.13	76.16 ± 0.77	79.21 ± 1.14	87.11 ± 0.89	89.21 ± 1.41	82.98 ± 1.36

them in a 2-dimensional euclidean space (as shown in the left of Fig. 2). We then choose $\alpha = \beta = \gamma = 1$ and $\tilde{n_t} = 1$ target training sample per class, and compute the new representations of source samples and target samples by linear KDACoM. MDS is also used to visualize these new representations as shown in the right of Fig. 2. From the figure, we can see that the original images from same person tend to get together, and the classification rule trained by one person can not be used for another person. The new representations computed by DACoM tend to cluster by their poses, which makes the trained classification rule can be used for both domains.

For the data set Webkb, the unsupervised DA approaches TCA and GFK tend to outperform the semi-supervised approaches DAMA and HFA. This may imply that the divergence between source and target domains are not very large, and thus the training target samples are in fact inactive for the domain adaptation. The covariance matching term in DACoM could effectively adapt the two domains in this case. See Table 5 for the all results of the algorithms on Webkb.

Table 6 lists the results of Warnat data with different choices of $\tilde{n_t} \in \{2, 4, 6, 8, 10\}$. Note that homogeneous methods are also applied to this dataset for comparison, although this dataset is heterogeneous since the features from two domains are measured in different ways. When the number of training samples from the target domain $\tilde{n_t}$ is 2, TCA and HeMap work the best for the two tasks, respectively, and (K)DACoM works the second best. When $\tilde{n_t}$ increases, (K) DACoM could obtain the highest classification accuracies. We can also see that, increasing the number of training samples in the target domain ($\tilde{n_t}$) tends to improve the performances of DAMA, HFA and DACoM. The improvement is less significant for TCA, GFK, ARTL and HeMap. The reason is that the new representations computed by TCA, GFK, ARTL and HeMap do not depend on the selected training samples in the target domain.

We report the results in the Tables 7 and 8, where by KDACoM we mean the linear KDACoM. DACoM or KDACoM always work the best for almost all the 20 tasks in the four datasets. In practice, DAMA or HFA work at least the second best for almost all the tasks. The unsupervised DA approach HeMap works the worst for almost all the cases, possibly due to the large shift between domains. In this this case, the training samples in the target domain play the main role during the adaptation. With the covariance matching, our DACoM could further improve DAMA.

As shown in Tables 4, 5, and 6, DACoM or its kernel version with linear kernel or Gauss kernels performs the best on the four datasets face, webKB, and warnat. Compared with the original DACoM, the kernel version could achieve the highest performance sometimes. For instance, in the domain adaptation tasks of P1→P2,P3→P2,P2→P3 in face-pose datasets, and one task of Valk→ Bullinger in Warnat datasets. This might be due to that KDACoM with Gaussian kernel could capture nonlinear matching information from two domains. Although KDACoM with nonlinear kernels such as Gaussian kernel and polynomial kernel may have advantages over DACoM or KDACoM with linear kernel, it will introduce extra parameters when generating kernels, the selection of which is a challenging problem for domain adaptation.

7.6 Comparison of the Sampling Strategies

We show the performance of the random sampling and OR sampling in DACoM or KDACoM with landmark strategy on Reuters4 for the two tasks: German→English and French→English. For each task, the selected training samples from target domain are fixed, but different number of landmark points are chosen with the number n_l various from 60 to 600. For a fixed n_l, we run the DACoM or KDACoM with the two

TABLE 6
Means and Standard Deviations of Classification Accuracies (%) of All Methods on Warnat

Data	n_t	SVMT	TCA	GFK	ARTL	HeMap	DAMA	HFA	DACoM	KDACoM-linear	KDACoM-Gauss
Bullinger→Valk	2	68.39 ± 1.99	**78.39 ± 0.11**	66.02 ± 0.65	67.85 ± 0.25	68.01 ± 0.33	68.71 ± 0.50	66.13 ± 1.23	74.30 ± 1.05	74.46 ± 1.26	72.74 ± 0.98
	4	75.17 ± 1.21	78.15 ± 0.51	71.57 ± 1.06	68.20 ± 0.32	67.98 ± 0.25	72.72 ± 0.61	72.87 ± 1.24	**82.75 ± 1.04**	82.58 ± 1.16	80.62 ± 0.97
	6	77.47 ± 1.83	83.12 ± 1.04	74.29 ± 0.99	67.94 ± 0.29	68.71 ± 0.28	74.89 ± 0.64	72.88 ± 1.66	85.71 ± 1.30	**87.06 ± 1.10**	84.59 ± 1.36
	8	85.12 ± 1.02	86.79 ± 0.41	77.16 ± 1.04	69.57 ± 0.26	69.26 ± 0.65	78.02 ± 0.71	84.14 ± 1.14	88.40 ± 0.61	**92.59 ± 0.61**	89.26 ± 0.77
	10	86.75 ± 1.04	85.32 ± 0.27	80.97 ± 1.30	68.44 ± 0.21	70.13 ± 0.09	81.09 ± 0.62	84.48 ± 1.19	90.45 ± 0.93	**94.29 ± 0.69**	90.97 ± 0.93
Valk→Bullinger	2	69.27 ± 2.28	63.54 ± 0.35	65.94 ± 1.40	61.04 ± 0.43	**76.04 ± 0.41**	64.67 ± 1.86	68.23 ± 2.39	71.67 ± 1.62	72.81 ± 1.66	74.58 ± 1.21
	4	69.89 ± 2.20	71.25 ± 0.60	67.61 ± 1.46	63.64 ± 0.00	77.50 ± 0.16	70.50 ± 2.06	63.41 ± 2.46	**79.55 ± 1.82**	76.14 ± 1.62	76.93 ± 1.70
	6	76.25 ± 2.05	72.75 ± 0.99	69.25 ± 0.93	66.75 ± 2.34	80.38 ± 0.27	73.05 ± 2.12	62.25 ± 2.79	**82.38 ± 0.86**	78.50 ± 1.79	79.50 ± 1.35
	8	78.19 ± 1.65	75.14 ± 0.14	74.31 ± 1.30	70.14 ± 1.27	83.75 ± 0.23	79.06 ± 1.79	69.72 ± 2.67	**87.08 ± 0.84**	85.00 ± 0.84	86.39 ± 1.17
	10	82.19 ± 2.25	73.28 ± 0.92	75.31 ± 1.06	85.94 ± 0.66	87.66 ± 0.16	85.06 ± 1.09	72.66 ± 3.32	88.28 ± 0.93	87.34 ± 1.10	**89.38 ± 0.92**

TABLE 7
Means and Standard Deviations of Classification Accuracies (%) on Reuters1 and Reuters2

	Reuters1 (200samples/domain, 2 classes)						Reuters2 (1000 samples/domain, 2 classes)					
Data	SVMT	HeMap	DAMA	HFA	DACoM	KDACoM	SVMT	HeMap	DAMA	HFA	DACoM	KDACoM
German → English	67.76 ± 2.12	77.73 ± 1.48	87.58 ± 0.61	78.04 ± 1.96	**89.54 ± 0.34**	88.90 ± 0.53	70.99 ± 1.68	70.40 ± 2.43	77.92 ± 1.83	82.95 ± 0.72	86.82 ± 0.46	**88.96 ± 0.29**
French → English	71.07 ± 1.73	80.54 ± 1.24	88.16 ± 0.75	79.62 ± 1.99	89.31 ± 0.35	**90.10 ± 0.42**	72.34 ± 1.86	69.00 ± 2.02	79.12 ± 1.22	80.90 ± 1.69	88.29 ± 0.42	**89.89 ± 0.46**
Spanish → English	70.48 ± 1.95	66.40 ± 1.97	86.99 ± 0.77	81.20 ± 0.94	89.21 ± 0.39	**89.90 ± 0.38**	70.18 ± 1.76	61.32 ± 2.27	83.77 ± 0.80	82.38 ± 1.12	88.89 ± 0.40	**89.83 ± 0.37**
Italian → English	73.57 ± 1.54	77.83 ± 2.14	85.77 ± 0.83	80.92 ± 1.05	88.88 ± 0.31	**89.90 ± 0.40**	73.84 ± 1.71	67.52 ± 1.98	83.56 ± 0.81	82.45 ± 1.01	88.32 ± 0.46	**90.33 ± 0.47**
English → German	69.92 ± 2.32	77.09 ± 2.87	88.09 ± 0.66	80.18 ± 1.24	90.77 ± 0.18	**91.35 ± 0.28**	67.66 ± 2.46	75.18 ± 2.71	85.85 ± 1.88	79.56 ± 1.84	88.54 ± 0.84	**92.61 ± 0.15**
French → German	68.60 ± 2.26	85.05 ± 1.19	89.21 ± 0.51	84.77 ± 1.25	90.94 ± 0.22	**92.09 ± 0.19**	77.04 ± 1.69	78.21 ± 2.22	86.45 ± 0.77	84.41 ± 1.37	91.50 ± 0.23	**92.66 ± 0.11**
Spanish → German	75.05 ± 1.80	72.37 ± 1.91	89.18 ± 0.37	84.80 ± 1.61	90.13 ± 0.20	**91.02 ± 0.25**	73.15 ± 1.71	61.44 ± 1.38	85.65 ± 1.75	81.00 ± 2.33	85.69 ± 1.51	**92.14 ± 0.42**
Italian → German	72.24 ± 1.94	74.77 ± 2.33	88.32 ± 0.43	82.63 ± 1.58	90.66 ± 0.17	**91.17 ± 0.26**	73.27 ± 2.09	82.51 ± 1.17	87.88 ± 0.74	81.68 ± 2.38	90.43 ± 1.33	**92.25 ± 0.42**
English → French	67.30 ± 1.79	85.43 ± 2.04	93.52 ± 0.76	85.20 ± 1.76	95.97 ± 0.13	**96.12 ± 0.14**	68.29 ± 2.30	81.10 ± 2.22	91.52 ± 0.55	82.96 ± 2.00	93.25 ± 0.29	**94.96 ± 0.18**
German → French	66.51 ± 2.56	88.34 ± 1.54	93.52 ± 0.71	87.07 ± 1.17	95.92 ± 0.16	**96.22 ± 0.18**	67.99 ± 2.31	84.23 ± 1.77	91.93 ± 0.48	83.59 ± 1.43	93.35 ± 0.37	**94.98 ± 0.20**
Spanish → French	69.46 ± 2.66	71.40 ± 1.92	93.62 ± 0.75	86.76 ± 1.59	93.90 ± 0.29	**95.54 ± 0.30**	64.89 ± 1.88	76.82 ± 2.41	92.34 ± 0.35	80.83 ± 1.95	88.25 ± 0.80	**94.95 ± 0.26**
Italian → French	70.08 ± 2.75	86.43 ± 2.05	93.52 ± 0.55	86.30 ± 0.86	95.69 ± 0.17	**95.99 ± 0.19**	64.13 ± 2.23	79.56 ± 2.22	90.37 ± 2.04	80.52 ± 2.27	92.71 ± 0.79	**94.80 ± 0.22**
English → Spanish	70.66 ± 1.45	73.67 ± 1.50	87.22 ± 0.85	77.60 ± 1.83	88.67 ± 0.54	**90.89 ± 0.28**	69.50 ± 1.95	69.64 ± 1.25	**91.64 ± 0.60**	77.75 ± 2.07	89.57 ± 0.44	91.24 ± 0.34
German → Spanish	65.66 ± 1.70	76.94 ± 1.43	85.33 ± 1.11	74.44 ± 2.13	86.81 ± 0.60	**88.65 ± 0.60**	65.04 ± 1.62	59.75 ± 2.81	**90.73 ± 0.52**	75.21 ± 1.63	88.81 ± 0.86	89.19 ± 1.05
French → Spanish	69.23 ± 2.02	73.57 ± 1.74	86.20 ± 1.42	78.85 ± 1.85	86.17 ± 0.68	**89.59 ± 0.33**	71.56 ± 2.38	69.94 ± 2.61	90.19 ± 1.36	81.87 ± 1.57	88.51 ± 0.52	**91.44 ± 0.58**
Italian → Spanish	67.91 ± 1.92	78.16 ± 1.32	87.27 ± 0.63	80.33 ± 1.23	87.07 ± 0.49	**89.59 ± 0.24**	70.79 ± 2.39	65.30 ± 2.21	91.31 ± 1.71	80.65 ± 2.03	90.74 ± 0.64	**91.61 ± 0.67**
English → Italian	70.48 ± 1.63	81.45 ± 1.60	87.58 ± 0.94	77.12 ± 1.55	89.01 ± 0.63	**91.25 ± 0.59**	67.98 ± 1.83	75.10 ± 2.22	85.22 ± 1.44	75.74 ± 1.56	88.58 ± 0.52	**91.24 ± 0.54**
German → Italian	69.18 ± 1.69	85.20 ± 1.50	86.35 ± 0.77	78.29 ± 1.29	**90.48 ± 0.54**	89.69 ± 0.65	68.37 ± 1.67	76.72 ± 2.63	84.74 ± 1.38	77.10 ± 1.48	86.23 ± 0.98	**88.53 ± 0.95**
French → Italian	67.17 ± 1.94	81.86 ± 1.69	86.02 ± 1.32	77.73 ± 1.79	88.93 ± 0.70	**90.51 ± 0.74**	69.56 ± 1.92	83.06 ± 2.53	85.20 ± 1.66	77.71 ± 1.29	88.37 ± 0.63	**90.90 ± 0.59**
Spanish → Italian	70.84 ± 2.03	71.89 ± 1.33	86.25 ± 0.92	77.47 ± 1.62	87.78 ± 0.71	**90.66 ± 0.81**	69.30 ± 1.84	71.76 ± 1.84	87.64 ± 0.90	79.32 ± 1.44	88.79 ± 0.70	**91.17 ± 0.49**

TABLE 8
Means and Standard Deviations of Classification Accuracies (%) on the Reuters3 and Reuters4

	Reuters3 (1200 samples/domain, 6 classes)						Reuters4 (2400 samples/domain, 6 classes				
Data	SVMT	HeMap	DAMA	HFA	DACoM	KDACoM	SVMT	HeMap	DAMA	HFA	KDACoM
German → English	32.80 ± 1.35	56.55 ± 0.90	64.25 ± 0.83	66.20 ± 0.39	67.17 ± 0.39	**70.93 ± 0.45**	31.95 ± 0.90	51.20 ± 0.68	53.11 ± 0.59	65.65 ± 0.35	**70.33 ± 0.52**
French → English	32.80 ± 1.35	62.88 ± 0.79	64.32 ± 0.81	66.11 ± 0.41	68.57 ± 0.50	**69.02 ± 0.54**	31.95 ± 0.90	52.91 ± 1.01	52.73 ± 0.72	65.63 ± 0.35	**70.56 ± 0.43**
Spanish → English	32.80 ± 1.35	54.32 ± 0.74	62.47 ± 0.97	66.19 ± 0.40	68.71 ± 0.47	**70.12 ± 0.55**	31.95 ± 0.90	52.58 ± 0.83	54.36 ± 0.53	65.59 ± 0.38	**71.00 ± 0.37**
Italian → English	32.80 ± 1.35	55.64 ± 0.93	64.44 ± 0.78	66.18 ± 0.36	67.97 ± 0.44	**68.67 ± 0.48**	31.95 ± 0.90	61.85 ± 0.50	54.38 ± 0.51	65.56 ± 0.34	**70.04 ± 0.47**
English → German	32.21 ± 0.98	64.09 ± 0.92	69.39 ± 0.60	66.49 ± 0.53	70.32 ± 0.44	**72.87 ± 0.43**	32.35 ± 1.01	61.47 ± 0.88	65.62 ± 0.49	65.17 ± 0.67	**72.19 ± 0.50**
French → German	32.21 ± 0.98	64.39 ± 0.66	68.84 ± 0.55	66.20 ± 0.62	70.35 ± 0.41	**72.66 ± 0.44**	32.35 ± 1.01	63.76 ± 0.68	64.30 ± 0.62	65.20 ± 0.67	**72.96 ± 0.57**
Spanish → German	32.21 ± 0.98	56.94 ± 0.41	68.73 ± 0.72	66.53 ± 0.49	70.24 ± 0.46	**72.07 ± 0.46**	32.35 ± 1.01	57.26 ± 0.19	64.19 ± 0.60	65.04 ± 0.67	**73.71 ± 0.61**
Italian → German	32.21 ± 0.98	67.38 ± 0.53	68.61 ± 0.52	66.46 ± 0.54	68.57 ± 0.46	**72.77 ± 0.53**	32.35 ± 1.01	57.02 ± 0.63	65.59 ± 0.42	65.11 ± 0.67	**71.93 ± 0.61**
English → French	32.75 ± 0.85	62.25 ± 0.49	66.33 ± 0.51	65.08 ± 0.73	68.87 ± 0.35	**70.06 ± 0.42**	32.32 ± 1.25	55.43 ± 0.54	64.48 ± 0.55	66.38 ± 0.44	**72.39 ± 0.43**
German → French	32.75 ± 0.85	60.75 ± 0.69	66.27 ± 0.49	65.05 ± 0.73	66.41 ± 0.53	**70.25 ± 0.46**	32.32 ± 1.25	64.51 ± 0.35	63.57 ± 0.55	66.37 ± 0.43	**72.49 ± 0.45**
Spanish → French	32.75 ± 0.85	56.25 ± 0.81	64.98 ± 0.47	64.94 ± 0.73	68.46 ± 0.49	**70.26 ± 0.42**	32.32 ± 1.25	61.24 ± 0.39	64.30 ± 0.56	66.31 ± 0.46	**73.53 ± 0.51**
Italian → French	32.75 ± 0.85	63.18 ± 0.51	66.16 ± 0.49	65.05 ± 0.72	67.77 ± 0.37	**69.53 ± 0.34**	32.32 ± 1.25	62.86 ± 0.33	64.38 ± 0.49	66.35 ± 0.43	**72.78 ± 0.47**
English → Spanish	45.32 ± 1.31	62.91 ± 0.72	72.23 ± 0.59	69.83 ± 0.77	70.77 ± 0.44	**72.91 ± 0.43**	40.59 ± 0.93	63.12 ± 0.28	71.15 ± 0.54	69.50 ± 0.45	**73.74 ± 0.47**
German → Spanish	31.67 ± 0.97	60.38 ± 1.38	63.49 ± 0.90	64.56 ± 0.47	70.22 ± 0.40	**73.24 ± 0.47**	40.59 ± 0.93	61.75 ± 0.47	70.56 ± 0.58	69.47 ± 0.48	**74.95 ± 0.45**
French → Spanish	45.32 ± 1.31	59.42 ± 0.49	**72.49 ± 0.57**	69.77 ± 0.75	70.86 ± 0.46	**72.49 ± 0.54**	40.59 ± 0.93	61.36 ± 0.59	70.21 ± 0.59	69.48 ± 0.44	**75.13 ± 0.50**
Italian → Spanish	45.32 ± 1.31	61.37 ± 1.30	72.05 ± 0.71	69.70 ± 0.78	70.34 ± 0.44	**72.79 ± 0.49**	40.59 ± 0.93	59.68 ± 0.54	70.48 ± 0.58	69.40 ± 0.48	**74.66 ± 0.40**
English → Italian	31.88 ± 1.64	50.09 ± 0.56	68.10 ± 0.40	63.63 ± 0.61	68.49 ± 0.51	**69.95 ± 0.53**	31.28 ± 1.53	59.27 ± 0.56	61.38 ± 0.44	62.83 ± 0.44	**67.78 ± 0.55**
German → Italian	31.88 ± 1.64	60.96 ± 0.66	68.11 ± 0.49	63.61 ± 0.60	65.56 ± 0.64	**69.87 ± 0.57**	31.28 ± 1.53	50.12 ± 0.47	60.56 ± 0.51	62.77 ± 0.44	**69.70 ± 0.60**
French → Italian	31.88 ± 1.64	61.59 ± 0.87	68.09 ± 0.49	63.47 ± 0.61	68.61 ± 0.54	**68.71 ± 0.50**	31.28 ± 1.53	60.62 ± 0.54	60.38 ± 0.45	63.02 ± 0.41	**70.02 ± 0.56**
Spanish → Italian	31.88 ± 1.64	46.25 ± 0.92	67.53 ± 0.44	63.53 ± 0.59	69.38 ± 0.54	**71.15 ± 0.50**	31.28 ± 1.53	47.81 ± 0.69	61.16 ± 0.52	62.78 ± 0.46	**70.90 ± 0.57**

kinds of samplings for 100 times, and report the mean and the standard error of the accuracies.

Fig. 3 plots curves of the clustering accuracy of DACoM/KDACoM with two sampling strategies with respect to the number n_l of landmark points. The two samplings perform similarly when n_l is relatively large. If n_l is small, we suggest the QR sampling since it is a more stable than the random sampling. In this example, using the landmark strategy, DACoM performs better than KDACoM if n_l is not small, while KDACoM works better for small n_l. Compared with accuracy 70 percent of DACoM or KDACoM on the whole data set, shown in Table 7, the landmark strategy does not decrease the accuracy of DACoM/KDACoM if we use $n_l = 600$ landmark points.

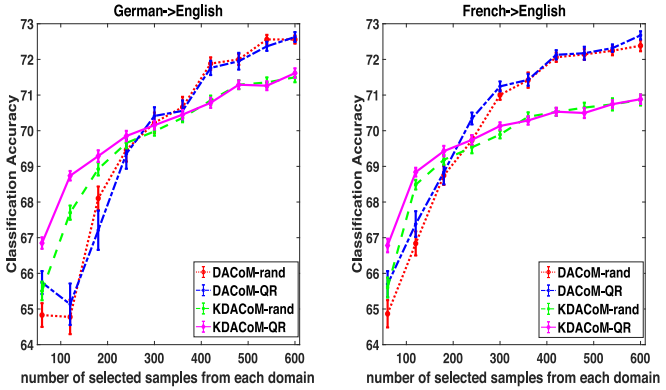

Fig. 3. Comparison of two sampling strategies over different number of landmark points.

7.7 Performance of DACoM on Large-Scale Data

We show the performance of DACoM/KDACoM using the landmark strategy on data sets in large scale on the three kinds of data sets: (1) synthetic data sets with variable feature dimension and number of samples for checking the dependence of DACoM/KDACoM on the data scale, (2) the real world data set Reuters5 that has about 55,000 samples totally and the similar scale of features, and (3) the data set Digit that has 81,000 samples with small number of features, on which we can compare the DACoM/KDACoM with or without the landmark strategy.

The synthetic data are constructed as follows. We first generate a two-dimensional dataset using the same setting for the gauss data in Section 7.1. Then combine them with some noise features to enlarge the feature dimension. The feature numbers are fixed as $p_s = p_t = 5{,}000$, while the number of samples $n_s = n_t$ are various from 1000 to 10000. We choose $n_l = 100$ landmark samples from each domain by random sampling or QR sampling, as the training data. The average accuracies of SVMT, DACoM and KDACoM are listed in Table 9. Since a small number of landmark samples are trained in DACoM/KDACoM, the computational load is very low—the training time is less than 1 second. Using the landmark strategy, DACoM or KDACoM with linear kernel can also achieve at a high accuracy around 97 percent. DACoM/KDACoM is vary stable with respect to the number of landmark points in this experiment.

For the large-scale set Reuters5, we use 600 landmark points in linear KDACoM. The classification accuracy is 71.51 percent for the task French → German. The accuracy

TABLE 9
The Classification Accuracies for (K)DACoM with Two Sampling Strategies for the Simulation Datasets

$n_s(=n_t)$	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
SVMT	95.16	91.68	95.11	94.79	94.08	93.82	94.39	94.98	92.55	93.91
DACoM-rand	97.55	96.80	97.37	97.22	97.64	96.97	97.43	97.15	97.42	96.97
DACoM-QR	97.67	96.95	97.41	97.42	97.59	97.14	97.38	97.40	97.46	97.37
KDACoM-rand	97.67	97.28	97.41	97.24	97.73	97.23	97.55	97.43	97.10	97.42
KDACoM-QR	97.74	97.27	97.39	97.45	97.70	97.23	97.55	97.40	97.57	97.38

increase to 73.01 percent for the task German → French. Because of the large scale, we cannot compare it with DACoM/KDACoM without landmark strategy. However, considering that the accuracy of DACoM/KDACoM on the relatively large Reuters4 is slightly hight than that on the relatively small Reuters3 as shown in Table 8, we guess DACoM/KDACoM on the original Reuters5 may achieve an accuracy slightly higher than that on Reuters4. If it is true, we may conclude that the landmark strategy also works well for DACoM/KDACoM on this data set.

A direct comparison can be done on the set Digit. Although Digit has huge number of samples, the total number of features is relatively small: $p = p_s + p_t = 1040$. Hence, we can run DACoM on the whole data set directly, without the landmark strategy. The results are shown in Table 10. In the example, DACoM can achieve around 80 percent accuracies for both the two tasks, though only a few labeled samples can be used in the target domain. To check the performance of the landmark strategy on this database, we also run DACoM with random sampling of $n_l = 500$ landmark points. The landmark strategy gives almost the same accuracy as that of DACoM on the original whole data set, just slightly decreasing about 1.5 and 0.35 percent accuracy for the tasks USPS→MNIST and MNIST→USPS, respectively.

7.8 Convergence Analysis

We have theoretically showed the convergence property of the DACoM algorithm in Section 5.4. Now we further study the convergence property of our DACoM algorithm empirically on three tasks of WebKB datasets. We use the same experimental setting as before, and report the value of the objective function and the classification accuracy with respect to the number of iterations in Fig. 4. We can see that for all the three tasks, the objective function values of DACoM converge decreasingly very fast, and the classification accuracies converge increasingly.

7.9 Parameters Sensitivity Analysis

The DACoM model is very robust on the parameter α—almost no changes in classification accuracy even if we set $\alpha = 0$. So, only the parameter β should be tuned to get a good accuracy—we always set $\gamma = 1$ and it works well in all the experiments reported before. However, if we do not use the last term, i.e., set $\gamma = 0$, the clustering accuracy will decrease. To further show the contribution of the each term in DACoM, we check the performance of DACoM by deleting one of the four terms. So, we have five versions totally: original DACoM with all terms, DACoM without covariance matching term, marked as $\theta = 0$, DACoM without latent spectral structure

TABLE 10
Means and Standard Deviations of Classification Accuracies (%) of SVMT, DACoM-Sampling, and DACoM on Digit

Domains	$\tilde{n}_t$	DACoM-sampling	DACoM
USPS → MNIST	20	76.03 ± 0.25	77.54 ± 0.36
	100	82.14 ± 0.17	83.63 ± 0.27
MNIST → USPS	20	77.16 ± 0.32	77.50 ± 0.28
	100	81.79 ± 0.18	82.15 ± 0.17

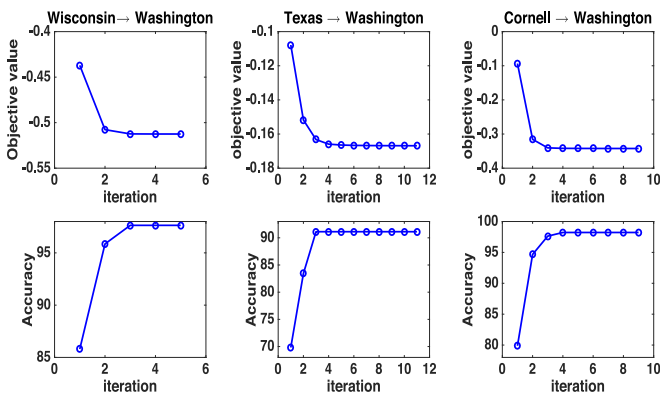

Fig. 4. Illustration of the convergence of DACoM by Webkb datasets.

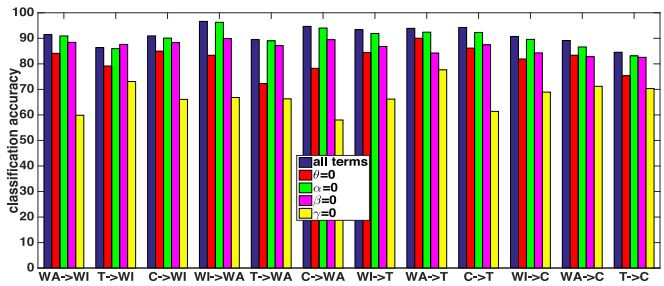

Fig. 5. Performances of DACoM on the Webkb datasets with all the terms or without covariance matching ($\theta = 0$), latent spectral structure ($\alpha = 0$), within-class discrimination ($\beta = 0$), or between-class discrimination ($\gamma = 0$).

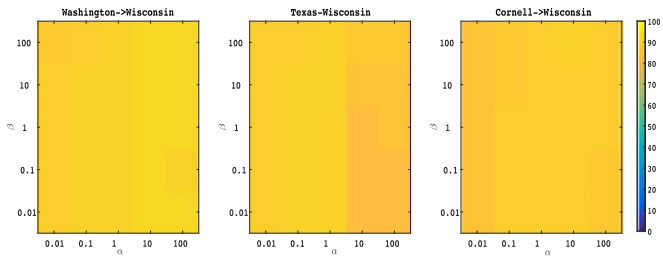

Fig. 6. Performances of our DACoM using different parameters of α and β on the Reuters1 datasets. The color represents the classification accuracy.

TABLE 11
CPU time (s) of the Compared Algorithms

Data	Domains	TCA	GFK	ARTL	HeMap	DAMA	HFA	KDACoM
Warnat	Bullinger→valk	42.77	0.23	1.76	3.63	386.40	0.51	0.16
	valk→Bullinger	39.18	0.18	3.52	3.40	355.13	0.31	0.21
WebKB	Washington→Wisconsin	1.90	0.22	2.34	3.92	5.27	0.96	0.24
	Texas→Wisconsin	1.26	0.18	1.41	3.34	5.17	0.52	0.18
	Cornell→Wisconsin	1.95	0.18	1.24	3.39	5.69	0.17	0.16
Face-pose	P2→P1	0.99	0.08	0.21	3.18	0.89	0.21	0.07
Reuters4	German →English	-	-	-	2.70	11.96	176.66	0.91
	French →English	-	-	-	2.13	11.62	171.04	0.86
	Spanish→English	-	-	-	2.26	8.96	157.82	0.90
	Italian→English	-	-	-	2.41	10.59	174.10	0.61

term ($\alpha = 0$), DACoM without the within-class discriminative term ($\beta = 0$), and DACoM without the between-class discriminative term ($\gamma = 0$). In Fig. 5, we report the classification accuracies of the generating five DACoM models on 12 adaptation tasks in WebKB datasets, where WI, WA, T, and C mean Wisconsin, Washington, Texas, and Cornell, respectively. In these datasets, the between-class discriminative term plays the first important role. The covariance matching is the second role in the importance. In most of the adaptation tasks, the covariance matching can significantly increase the classification accuracy.

We also apply DACoM on the WebKB datasets to check the parameters sensitivity. We change α and β from $\{0.01, 0.1, 1, 10, 100\}$, and compute the classification accuracies. Fig. 6 shows the classification accuracies for every possible choice of (α, β) pair. We can see that for these five datasets, the accuracies obtained by DACoM are stable in a relatively large region. The results show the robustness of DACoM in parameters.

7.10 Complexity and Time Analysis

To solve the DACoM model (6), our eigen-updating algorithm needs to solve a sequence of eigenvalue decomposition problem, with the total computation complexity around $O(n^2)$, based on the analysis in Sections 5.2 and 5.3. When $n = n_s + n_t$ is large, the complexity can be reduced to $O(n_l^2)$ by landmark strategies, where n_l is the number of landmark samples from each domain. We perform the experiments to show the computation efficiency of linear KDACoM. For each dataset listed in Table 11, we report the training time (in seconds) for each method. Note that for a fair comparison, we report the training time with the optimal parameters corresponding to the highest accuracy for each method. For Reuters4 dataset, we report the time by KDACoM with random sampling strategy. The table shows that DACoM is the most efficient in almost all the cases.

8 Conclusion

We have proposed a new domain adaptation method called DACoM by our new strategy of covariance matching. In DACoM, we embed the original data into a common space such that the distance of the mapped data distributions from two domains are minimized and the local geometric structure and discriminative information are best preserved. We propose to measure the distance of the mapped data distributions using their covariances. We aim to find new data representations of the two domains such that their covariance matrices are similar and the local geometric structure and discriminative information are best preserved. With these new representations, a classifier can be learnt using the training data from both domains and then used for predict the labels of target test samples. The DACoM optimization problem is quartic and thus difficult to solve. We prove that the KKT conditions could at least ensure local optimality under a certain condition. The KKT conditions require to solve a nonlinear eigenvalue problem, and thus we propose an eigen-updating algorithm. Theoretical analysis is given to show the convergence property of the algorithm. To deal with large scale problems with large number of features, we further propose several strategies to reduce the computational load. Comprehensive experiments were conducted to show that our DACoM outperforms other existing methods. In the future, we will investigate how to deal with the more difficult unsupervised domain adaptation by covariance matching.

Acknowledgments

We would like to thank Prof. Karsten Borgwardt from ETH Zurich for valuable discussion, and Ms Hongsha Guo for kindly collecting part of the real datasets. The work was supported by NSFC projects 11471256, 11571312, and 91730303, and National Basic Research Program of China (973 Program) 2015CB352503.

References

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[2] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, *Adapting Visual Category Models to New Domains*, Berlin, Germany: Springer, 2010, pp. 213–226.
[3] 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. Vis. Pattern Recognit.*, Jun. 2011, pp. 1785–1792.
[4] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang, "Heterogeneous transfer learning for image classification," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 1304–1309.
[5] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2006, pp. 120–128.
[6] J. Blitzer, D. Foster, and S. Kakade, "Domain adaptation with coupled subspaces," in *Proc. Conf. Artif. Intell. Statist.*, 2011, pp. 173–181.

[7] L. Duan, D. Xu, and I. W. Tsang, "Learning with augmented features for heterogeneous domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2012, pp. 711–718.
[8] P. Prettenhofer and B. Stein, "Cross-language text classification using structural correspondence learning," in *Proc. 48th Annu. Meet. Assoc. Comput. Linguistics*, Jul. 2010, pp. 1118–1127.
[9] W. Dai, Y. Chen, G. R. Xue, Q. Yang, and Y. Yu, "Translated learning: Transfer learning across different feature spaces," in *Proc. Advances Neural Inf. Process. Syst.* 2008, pp. 353–360.
[10] S. J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, "Adaptive localization in a dynamic WiFi environment through multi-view learning," in *Proc. 22nd Nat. Conf. Artif. Intell.*, 2007, pp. 1108–1113.
[11] V. W. Zheng, S. J. Pan, Q. Yang, and J. J. Pan, "Transferring multi-device localization models using latent multi-task learning," in *Proc. 23rd Nat. Conf. Artif. Intell.*, 2008, pp. 1427–1432.
[12] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguistics*, 2007, pp. 440–447.
[13] A. Kumar, A. Saha, and H. Daume, "Co-regularization based semi-supervised domain adaptation," in *Proc. Advances Neural Inf. Process. Syst.*, 2010, pp. 478–486.
[14] M. Chen, K. Q. Weinberger, and J. C. Blitzer, "Co-training for domain adaptation," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 2456–2464.
[15] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Mach. Learn.*, Jun. 2011, pp. 513–520.
[16] D. Bollegala, T. Mu, and J. Y. Goulermas, "Cross-domain sentiment classification using sentiment sensitive embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 398–410, Feb. 2016.
[17] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Proc. Advances Neural Inf. Process. Syst.*, 2008, pp. 1433–1440.
[18] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Statistical Planning Inference*, vol. 90, no. 2, pp. 227–244, 2000.
[19] 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.
[20] M. Xiao and Y. Guo, "Semi-supervised kernel matching for domain adaptation," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 1183–1189.
[21] M. Long, J. Wang, G. Ding, S. J. Pan, and Philip S. Yu, "Adaptation regularization: A general framework for transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1076–1089, May 2014.
[22] M. Long, J. Wang, G. Ding, D. Shen, and Q. Yang, "Transfer learning with graph co-regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 7, pp. 1–1, Jul. 2014.
[23] C. Wang and S. Mahadevan, "Heterogeneous domain adaptation using manifold alignment," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1541–1546.
[24] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proc. 23rd Nat. Conf. Artif. Intell.*, 2008, pp. 677–682.
[25] S. J. Pan, Ivor W. Tsang, James T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.
[26] S. Si, D. Tao, and B. Geng, "Bregman divergence-based regularization for transfer subspace learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 7, pp. 929–942, Jul. 2010.
[27] X. Shi, Q. Liu, W. Fan, P. S. Yu, and R. Zhu, "Transfer learning on heterogenous feature spaces via spectral transformation," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 1049–1054.
[28] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 9, pp. 1853–1865, Sep. 2017.
[29] S. Ben-david, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. Advances Neural Inf. Process. Syst.*, 2006, pp. 137–144.
[30] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "Learning bounds for domain adaptation," in *Proc. Advances Neural Inf. Process. Syst.*, 2008, pp. 129–136.
[31] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinf.*, vol. 22, no. 14, pp. e49–e57, Jul. 2006.
[32] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3208–3215.
[33] H. Wang and Q. Yang, "Transfer learning by structural analogy," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 513–518.
[34] P. Warnat, R. Eils, and B. Brors, "Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes," *BMC Bioinf.*, vol. 6, 2005, Art. no. 265.
[35] L. Bullinger, K. Dohner, E. Bair, S. Frohling, R. F. Schlenk, R. Tibshirani, H. Dohner, and J. R. Pollack, "Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia," *N. Engl. J. Med.*, vol. 350, pp. 1605–1616, Apr. 2004.
[36] P. J. Valk, et al., "Prognostically useful gene-expression profiles in acute myeloid leukemia," *N. Engl. J. Med.*, vol. 350, pp. 1617–1628, 2004.
[37] M. Amini, N. Usunier, and C. Goutte, "Learning from multiple partially observed views - an application to multilingual text categorization," in *Proc. Advances Neural Inf. Process. Syst.*, 2009, pp. 28–36.
[38] K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2066–2073.

Limin Li received the bachelor's and master's degrees from Zhejiang University, in 2004 and 2006, respectively, and the PhD degree in mathematics from the University of Hong Kong, in 2010. She then worked as a postdoctoral fellow with Max Planck Institute of Intelligent System. She is currently an associate professor with the School of Mathematics and Statistics in Xi'an Jiaotong University, Xi'an, China. Her main research area is machine learning and the applications in bioinformatics.

Zhenyue Zhang received the BS degree in mathematics from Fudan University, Shanghai, China, in 1982, and the PhD degree in scientific computing from Fudan University, in 1989. He was an assistant professor with the Department of Mathematics, Fudan University from 1982 to 1985, and is a full professor of the Department of Mathematics, Zhejiang University since 1998. His current research interests include machine learning and its applications, numerical linear algebra, and data science.