Learning Distribution-Matched Landmarks for Unsupervised Domain Adaptation

Mengmeng Jing, Jingjing Li(✉), Jidong Zhao, and Ke Lu

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
poxiaoge77@foxmail.com, lijin117@yeah.net, {jdzhao,kel}@uestc.edu.cn

Abstract. Domain adaptation is widely used in database applications, especially in data mining. The basic assumption of domain adaptation (DA) is that some latent factors are shared by the source domain and the target domain. Revealing these shared factors, as a result, is the core operation of many DA approaches. This paper proposes a novel approach, named Learning Distribution-Matched Landmarks (LDML), for unsupervised DA. LDML reveals the latent factors by learning a domain-invariant subspace where the two domains are well aligned at both feature level and sample level. At the feature level, the divergences of both the marginal distribution and the conditional distribution are mitigated. At the sample level, each sample is evaluated so that we can take full advantage of the pivotal samples and filter out the outliers. Extensive experiments on two standard benchmarks verify that our approach can outperform state-of-the-art methods with significant advantages.

Keywords: Domain adaptation · Transfer learning
Landmark selection

1 Introduction

In real-world multimedia databases and data mining applications, cross-domain contents are often involved, such as videos, audios, texts and so on [3,5,29]. Naturally, there exists the need for multi-domain knowledge transferring among these applications. For example, if a biologist wants to develop an application that can accurately index or retrieve an endangered breed of bird, numerous labeled images of the birds are required if he or she employs the conventional machine learning algorithms. Unfortunately, it is impossible to collect plenty of such images and consume high labor-cost to label them manually. In recent years, DA algorithms get a rapid development for their advantage on solving the cross-domain knowledge transfer problems. Domain adaptation has been widely used in computer vision and database applications, such as image classification [18, 19], object recognition [11], text analysis [29] and recommendation systems [16].

© Springer International Publishing AG, part of Springer Nature 2018
J. Pei et al. (Eds.): DASFAA 2018, LNCS 10828, pp. 491–508, 2018.
https://doi.org/10.1007/978-3-319-91458-9_30

The principle of existing DA algorithms is to learn the latent shared factors between the source and target domains. These factors are at either feature level or sample level. Most existing DA algorithms, therefore, can be roughly classified into two groups: feature-based methods and sample-based methods.

Methods in the first group aim to reduce the gaps between domains at the feature level. The idea of these methods is to learn a new feature representation, typically a latent subspace, to minimize distribution divergence between domains. For example, Gopalan et al. [11] find intermediate representations by learning subspaces along the geodesic path that connects the source subspace and the target one on the Grassmann manifold. Fernando et al. [7] align subspaces of the two domains by directly using one linear mapping matrix so that the source and target subspaces can move closer. LRDE [18] constructs a novel graph structure under the graph embedding framework so that it can preserve geometric information in both the ambient instance space and the embedding feature space. JGSA [28] learns two coupled projections that project the source domain and the target domain data onto low-dimensional subspaces where the geometric structures are preserved and the distribution divergence is reduced. For a better understanding, we show the main idea of these methods in Fig. 1(a) and (b).

Methods in the second group employ landmark selection at the sample level. Landmarks are defined as a subset of the training samples which can bridge the two domains. Figure 1(c) illustrates the importance of landmark selection. After projecting data onto a low-dimensional subspace, discrepancies among samples remain. Some cross-domain pivotal samples are close to each other and can reduce the distribution divergence between domains, while other samples are far from the ones in the other domain and can increase the gaps between domains. The process of landmark selection takes full advantage of pivotal samples that align the source and target domains well, and filters out outliers. For instance, Gong et al. [9] discover landmarks at multiple granularities and construct auxiliary tasks correspondingly to compose domain-invariant features. LSSA [1] computes a quality measure for each sample using the Gaussian kernel. If the measure is above a threshold, the corresponding sample is selected as a landmark. CDLS [15] selects landmarks that match cross-domain data distributions well by using the empirical Maximum Mean Discrepancy (MMD) [12].

However, most of the existing approaches are based on either features or samples. They consider only one of the two factors independently. We observe that features and samples are factors at two different levels, and they can adapt two domains at different granularities: coarse-grained adaptation at the sample level and fine-grained adaptation at the feature level. It is obvious that both of the factors are interrelated and can reinforce each other. Therefore, in this paper, we propose a novel approach, named Learning Distribution-Matched Landmarks (LDML), for unsupervised DA. LDML learns a domain-invariant subspace where the two domains are well aligned at both feature level and sample level. Technically, at the feature level, we use MMD as the measure of the marginal and conditional distribution divergences. Then, minimizing MMD is employed to

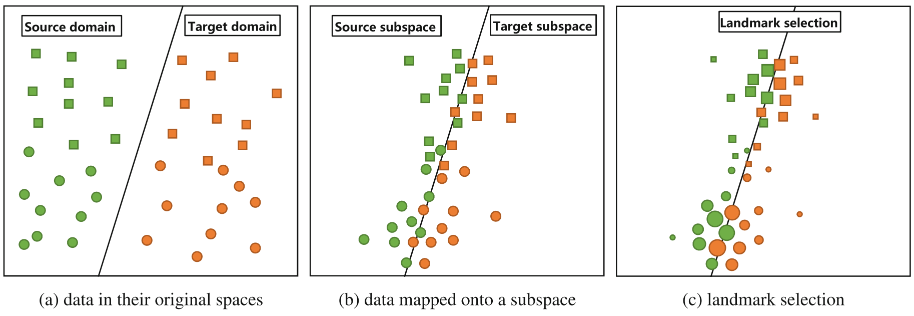

Fig. 1. The main idea of LDML. Points with the same color stand for samples that belong to the same domain and points with the same shape stand for samples that belong to the same class. The size of a point represents the weight of a sample. (a) the distribution divergence of data in their original spaces is large; (b) the distribution divergence between domains is reduced after mapping data onto a latent shared subspace; (c) select landmarks that can bridge the source and target subspaces.

align the two domains. Furthermore, we construct a graph under the graph embedding framework to preserve the locality structures in both domains. At the sample level, we evaluate each sample and put different weights on them. As a result, the pivotal samples are selected automatically in the iteration process and can be used to train a robust model. The contributions of this paper can be summarized as follows:

(1) We propose a multi-granularity DA method which considers factors at not only fine-grained feature level but also coarse-grained sample level. Therefore, our method can not only reduce the distribution divergence between domains but also be robust to outliers.
(2) Different from the conventional domain adaptation methods which learn only one projection matrix, we learn two projection matrices, one for each domain, so that the proposed method is more generalized and can be easily extended to handle the heterogeneous domain adaptation problems where the two domains may have different dimensionalities and features.
(3) Extensive experiments on two standard benchmarks with three features verify that our algorithm outperforms state-of-the-art algorithms with a significant advantage.

2 Related Work

As stated above, existing DA methods can be roughly classified as feature-based methods and sample-based methods. We now give a brief review of these methods from both categories.

Feature-based methods can be further categorized as distribution matching, e.g., subspace learning [17,20,22], and property preserving, e.g., geometric structures preserving [8,28]. Distribution matching algorithms reduce marginal or conditional distribution divergence through learning a latent subspace shared by the two domains. The MMD-based subspace learning algorithms [20,22] are the typical examples of distribution matching. For instance, TCA [22] learns some transfer components across domains in a Reproducing Kernel Hilbert Space using MMD, which mitigates the marginal distribution between domains. JDA [20] improves TCA by jointly considering the marginal and the conditional distributions and building a new feature representation to guarantee robustness. Property preserving algorithms preserve important properties when embedding data to a low-dimensional subspace. Graph-based subspace learning algorithms [4,18,26] belong to this category. For example, LRDE [18] constructs a novel graph structure under the graph embedding framework. Specifically, LRDE builds a within-class graph to encourage data in the same class to move closer, and a between-class graph to make data between classes far away. JGSA [28] is a feature matching algorithm which learns two coupled projections that project the two domains onto low-dimensional subspaces where the geometric structures are preserved and the distribution divergence is reduced.

Landmark [30] selection is a DA method at the sample level. For instance, TJM [21] employs instance reweighting by imposing the $\ell_{2,1}$-norm regularizer on the source projection matrix. CDLS [15] selects landmarks that have higher matching degree across domains by minimizing MMD. LSSA [1] uses the Gaussian kernel to compute quality measures of all samples and selects landmarks by a quality threshold.

The most related work is JGSA [28]. However, our method is significantly different from JGSA in at least two aspects:

(1) JGSA is merely a feature-based method which does not consider discrepancies among samples. When the two domains are not closely related, JGSA still enforces samples close to each other though they are far away. This transfer may degrade the performance of DA, even lead to the negative transfer [23]. LDML reweights samples to ensure that the pivotal samples are taken full advantage of and outliers are filtered out. The advantage of landmark selection is illustrated in Fig. 1.
(2) To preserve the discriminative information, JGSA employs the LDA criterion on the source domain data, which assumes the data are sampled from Gaussian distribution and makes the algorithm sensitive to outliers [27]. LDML applies the graph embedding strategy to preserve the local and global geometric information, which is more robust.

Experimental results in Sect. 4.4 show that our method can achieve better performance compared with JGSA.

3 Learning Distribution-Matched Landmarks

3.1 Problem Definition

Definition 1. A domain D is defined by a feature space χ and its probability distribution $P(X)$, where $X \in \chi$. For a specific domain, a classification task T

consists of class information y and a classifier $f(x)$, that is $T = \{y, f(x)\}$. We use subscripts s and t to indicate the source domain and the target domain, respectively. This paper focuses on the following problem:

Problem 1. Given a labeled source domain $\{X_\mathrm{s}, y_\mathrm{s}\}$ and an unlabeled target domain $\{X_\mathrm{t}, y_\mathrm{t}\}$, where X_s and X_t are the source and target domain samples, y_s and y_t are labels for the corresponding domains respectively, y_t is unknown, $P(X_\mathrm{s}) \neq P(X_\mathrm{t})$ and $P(y_\mathrm{s}|X_\mathrm{s}) \neq P(y_\mathrm{t}|X_\mathrm{t})$, project the source and target domains onto a subspace by projection matrices A and B so that the common latent features shared by involved domains are uncovered, the data manifold structure is preserved, and the domain difference is minimized.

3.2 Problem Formulation

Distribution Matching. LDML reveals the latent factors by learning a domain-invariant subspace where the two domains are well aligned at both feature level and sample level. Previous work [14,20] only learn one projection matrix, which has a major limitation when the two domains have different dimensionalities. To gain strong generalization ability, we learn two different projection matrices, one for each domain. We deploy these two matrices to project the source and target domain data onto a latent shared subspace. Specifically, let A be the projection matrix for the source domain and B for the target domain. $X_\mathrm{s} \in \mathbb{R}^{m \times n_\mathrm{s}}$ and $X_\mathrm{t} \in \mathbb{R}^{m \times n_\mathrm{t}}$ are the source and target domain data respectively, where n_s and n_t are the total numbers of the corresponding domain samples, m is the original dimensionality. $A \in \mathbb{R}^{m \times d}$ can project the source domain data onto a d-dimensional subspace, where $d \ll m$. Then, the low-dimensional data can be represented by $A^\mathrm{T} X_\mathrm{s}$. Similarly the low-dimensional representation of the target domain data X_t is $B^\mathrm{T} X_\mathrm{t}$. Considering the substantial distribution divergence, we align both the marginal and the conditional distribution between the two domains by adopting MMD. It is worth noting that the learned subspace should be shared by two domains. Therefore, we employ the Frobenius norm to minimize the distance between the two domains. Then, our objective function can be written as:

$$\min_{A,B} E_\mathrm{M}(X_\mathrm{s}, X_\mathrm{t}, A, B) + E_\mathrm{C}(X_\mathrm{s}, X_\mathrm{t}, A, B) + \lambda \left\| A - B \right\|_\mathrm{F}^2 , \tag{1}$$

where $\lambda > 0$ is the regularization parameter, E_M and E_C match the marginal and the conditional cross-domain data distributions, respectively. For simplicity, we set E_M and E_C with the same coefficient. E_M can be calculated as:

$$E_\mathrm{M} = \left\| \frac{1}{n_\mathrm{s}} \sum_{i=1}^{n_\mathrm{s}} A^\mathrm{T} x_\mathrm{s}^i - \frac{1}{n_\mathrm{t}} \sum_{j=1}^{n_\mathrm{t}} B^\mathrm{T} x_\mathrm{t}^j \right\|_\mathrm{F}^2 , \tag{2}$$

E_{C} can be calculated as:

$$E_C = \sum_{c=1}^{C} \left(\left\| \frac{1}{n_{\mathrm{s}}^c} \sum_{i=1}^{n_{\mathrm{s}}^c} A^{\mathrm{T}} x_{\mathrm{s}}^{i,c} - \frac{1}{n_{\mathrm{t}}^c} \sum_{j=1}^{n_{\mathrm{t}}^c} B^{\mathrm{T}} x_{\mathrm{t}}^{j,c} \right\|_{\mathrm{F}}^2 + \frac{1}{n_{\mathrm{s}}^c n_{\mathrm{t}}^c} \sum_{i=1}^{n_{\mathrm{s}}^c} \sum_{j=1}^{n_{\mathrm{t}}^c} \left\| A^{\mathrm{T}} x_{\mathrm{s}}^{i,c} - B^{\mathrm{T}} x_{\mathrm{t}}^{j,c} \right\|_{\mathrm{F}}^2 \right) , \tag{3}$$

where C is the number of classes, n_{s}^c and n_{t}^c denote the total numbers of the source and target domain samples in class c, respectively. Since labels in the target domain are not available, we use the pseudo labels to classify the target domain samples.

Landmark Selection. At the sample level, we learn two weight vectors α and β for the source and target domains, respectively. Each entry of the vector is a weight of the corresponding sample. Then the objective function can be rewritten as:

$$\begin{aligned} &\min_{A,B} E_{\mathrm{M}}(\alpha, \beta, X_{\mathrm{s}}, X_{\mathrm{t}}, A, B) + E_{\mathrm{C}}(\alpha, \beta, X_{\mathrm{s}}, X_{\mathrm{t}}, A, B) + \lambda \left\| A - B \right\|_{\mathrm{F}}^2 , \\ &\text{s.t. } \{\alpha_i^c, \beta_i^c\} \in [0,1], \frac{{\alpha^c}^{\mathrm{T}} \mathbf{1}_{n_{\mathrm{s}}^c}}{n_{\mathrm{s}}^c} = \frac{{\beta^c}^{\mathrm{T}} \mathbf{1}_{n_{\mathrm{t}}^c}}{n_{\mathrm{t}}^c} = \delta , \end{aligned} \tag{4}$$

where $\alpha = [\alpha^1; \cdots; \alpha^c; \cdots; \alpha^C] \in \mathbb{R}^{n_{\mathrm{s}}}$ and $\beta = [\beta^1; \cdots; \beta^c; \cdots; \beta^C] \in \mathbb{R}^{n_{\mathrm{t}}}$ are the weights of data in the source and target domains respectively, $\alpha^c = [\alpha_1^c; \cdots; \alpha_{n_{\mathrm{s}}^c}^c]$, $\beta^c = [\beta_1^c; \cdots; \beta_{n_{\mathrm{t}}^c}^c]$, $\mathbf{1}_{n_{\mathrm{s}}^c} \in \mathbb{R}^{n_{\mathrm{s}}^c}$ and $\mathbf{1}_{n_{\mathrm{t}}^c} \in \mathbb{R}^{n_{\mathrm{t}}^c}$ are column vectors with all ones. $\delta \in [0,1]$ controls the average weight of the whole source or target domain samples.

The E_M and E_C in (4) can be updated as:

$$E_{\mathrm{M}} = \left\| \frac{1}{\delta n_{\mathrm{s}}} \sum_{i=1}^{n_{\mathrm{s}}} \alpha_i A^{\mathrm{T}} x_{\mathrm{s}}^i - \frac{1}{\delta n_{\mathrm{t}}} \sum_{j=1}^{n_{\mathrm{t}}} \beta_j B^{\mathrm{T}} x_{\mathrm{t}}^j \right\|_{\mathrm{F}}^2 ,$$

$$E_{\mathrm{C}} = \sum_{c=1}^{C} \left(\left\| \frac{1}{\delta n_{\mathrm{s}}^c} \sum_{i=1}^{n_{\mathrm{s}}^c} \alpha_i A^{\mathrm{T}} x_{\mathrm{s}}^{i,c} - \frac{1}{\delta n_{\mathrm{t}}^c} \sum_{j=1}^{n_{\mathrm{t}}^c} \beta_j B^{\mathrm{T}} x_{\mathrm{t}}^{j,c} \right\|_{\mathrm{F}}^2 + \frac{1}{\delta^2 n_{\mathrm{s}}^c n_{\mathrm{t}}^c} \sum_{i=1}^{n_{\mathrm{s}}^c} \sum_{j=1}^{n_{\mathrm{t}}^c} \left\| \alpha_i A^T x_{\mathrm{s}}^{i,c} - \beta_j B^T x_{\mathrm{t}}^{j,c} \right\|^2 \right) ,$$

Equation (4) can be further transformed to its matrix form as follows:

$$\min_{A,B} \operatorname{Tr} \left(\begin{bmatrix} A^{\mathrm{T}} & B^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} M_{\mathrm{ss}} + \lambda I & M_{\mathrm{st}} - \lambda I \\ M_{\mathrm{ts}} - \lambda I & M_{\mathrm{tt}} + \lambda I \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} \right) , \tag{5}$$

where

$$\begin{aligned} M_{\mathrm{ss}} &= X_{\mathrm{s}} H_{\mathrm{ss}} X_{\mathrm{s}}^{\mathrm{T}} , \quad M_{\mathrm{tt}} = X_{\mathrm{t}} H_{\mathrm{tt}} X_{\mathrm{t}}^{\mathrm{T}} , \\ M_{\mathrm{st}} &= X_{\mathrm{s}} H_{\mathrm{st}} X_{\mathrm{t}}^{\mathrm{T}} , \quad M_{\mathrm{ts}} = M_{\mathrm{st}}^{\mathrm{T}} . \end{aligned}$$

Each entry $(H_{\mathrm{ss}})_{ij}$ in $H_{\mathrm{ss}} \in \mathbb{R}^{n_{\mathrm{s}} \times n_{\mathrm{s}}}$ denotes the coefficient associated with ${x_{\mathrm{s}}^i}^{\mathrm{T}} x_{\mathrm{s}}^j$. Similar remarks can be applied to $H_{\mathrm{tt}} \in \mathbb{R}^{n_{\mathrm{t}} \times n_{\mathrm{t}}}$ and $H_{\mathrm{st}} \in \mathbb{R}^{n_{\mathrm{s}} \times n_{\mathrm{t}}}$. Detailed derivations are similar to that in CDLS [15]. For the conciseness, we omit the regular mathematical derivations in this paper.

Locality Structure Preservation. In general, a sample tends to have the same label with its k-nearest neighbors. This locality property is crucial in many computer vision tasks [4,18]. Therefore, we propose to preserve the geometric structures in the source domain so that the discriminative information can be maximized and then be transferred to the target domain. On the other hand, the geometric structures in the target domain should be preserved as well, so that the manifold structures in the original space can be retained. So we introduce two Laplacian graph terms, one for each domain:

$$\min \frac{\mathrm{Tr}(A^{\mathrm{T}} X_{\mathrm{s}} L_{\mathrm{w}}^{\mathrm{s}} X_{\mathrm{s}}^{\mathrm{T}} A)}{\mathrm{Tr}(A^{\mathrm{T}} X_{\mathrm{s}} L_{\mathrm{b}}^{\mathrm{s}} X_{\mathrm{s}}^{\mathrm{T}} A)} = \min \frac{\mathrm{Tr}(A^{\mathrm{T}} S_{\mathrm{w}}^{\mathrm{s}} A)}{\mathrm{Tr}(A^{\mathrm{T}} S_{\mathrm{b}}^{\mathrm{s}} A)}, \tag{6}$$

$$\min \frac{\mathrm{Tr}(B^{\mathrm{T}} X_{\mathrm{t}} L_{\mathrm{w}}^{\mathrm{t}} X_{\mathrm{t}}^{\mathrm{T}} B)}{\mathrm{Tr}(B^{\mathrm{T}} X_{\mathrm{t}} L_{\mathrm{b}}^{\mathrm{t}} X_{\mathrm{t}}^{\mathrm{T}} B)} = \min \frac{\mathrm{Tr}(B^{\mathrm{T}} S_{\mathrm{w}}^{\mathrm{t}} B)}{\mathrm{Tr}(B^{\mathrm{T}} S_{\mathrm{b}}^{\mathrm{t}} B)}, \tag{7}$$

where

$$S_{\mathrm{b}}^{\mathrm{s}} = X_{\mathrm{s}} L_{\mathrm{b}}^{\mathrm{s}} X_{\mathrm{s}}^{\mathrm{T}}\ , \quad S_{\mathrm{w}}^{\mathrm{s}} = X_{\mathrm{s}} L_{\mathrm{w}}^{\mathrm{s}} X_{\mathrm{s}}^{\mathrm{T}}\ ,$$
$$S_{\mathrm{b}}^{\mathrm{t}} = X_{\mathrm{t}} L_{\mathrm{b}}^{\mathrm{t}} X_{\mathrm{t}}^{\mathrm{T}}\ , \quad S_{\mathrm{w}}^{\mathrm{t}} = X_{\mathrm{t}} L_{\mathrm{w}}^{\mathrm{t}} X_{\mathrm{t}}^{\mathrm{T}}\ ,$$

where $L = D - W$, D is a diagonal matrix and its diagonal entry is $D_{ii} = \sum_{j \neq i} W_{ij}$. $L_{\mathrm{w}}^{\mathrm{s}}$ and $L_{\mathrm{b}}^{\mathrm{s}}$ are the Laplacian matrices of the intrinsic graph and the penalty graph for the source domain. Similarly, $L_{\mathrm{w}}^{\mathrm{t}}$ and $L_{\mathrm{b}}^{\mathrm{t}}$ are the Laplacian matrices for the target domain. W_{w} and W_{b} are the weight matrices for the intrinsic graph and the penalty graph, respectively. In this paper, we follow the following two criteria to construct W_{w} and W_{b}.

(a) Construct the intrinsic weight matrix W_{w}: For each sample x, connect the nearest neighbor v with x where v has the same class information with x.
(b) Construct the penalty weight matrix W_{b}: For each domain, connect the k-nearest vertex pairs where samples in each pair belong to different classes.

By applying (a), samples from the same class can be more compact and the local manifold structure can be preserved. By deploying (b), samples from the same domain but different classes can be more separable and the global discriminative information can be retained. We apply the heat kernel method to get W_{w} and W_{b}, e.g., if two samples x_i and x_j are connected, then the weight of them is $\exp(-((||x_i - x_j||^2)/(2\sigma^2)))$, otherwise it is 0.

Overall Objective Function. Considering all the above discussions, we get the overall objective function:

$$\min_{A,B} \frac{\mathrm{Tr}\left(\begin{bmatrix} A^{\mathrm{T}} & B^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} M_{\mathrm{ss}} + \gamma S_{\mathrm{w}}^{\mathrm{s}} + \lambda I & M_{\mathrm{st}} - \lambda I \\ M_{\mathrm{ts}} - \lambda I & M_{\mathrm{tt}} + \gamma S_{\mathrm{w}}^{\mathrm{t}} + (\lambda + \mu) I \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} \right)}{\mathrm{Tr}\left(\begin{bmatrix} A^{\mathrm{T}} & B^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \gamma S_{\mathrm{b}}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \gamma S_{\mathrm{b}}^{\mathrm{t}} + \mu S_{\mathrm{h}}^{\mathrm{t}} \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} \right)}, \tag{8}$$

where

$$S_{\mathrm{h}}^{\mathrm{t}} = X_{\mathrm{t}} (I_{\mathrm{t}} - \frac{1}{n_{\mathrm{t}}} \mathbf{1}_{n_{\mathrm{t}}} \mathbf{1}_{n_{\mathrm{t}}}^{\mathrm{T}}) X_{\mathrm{t}}^{\mathrm{T}} \tag{9}$$

is the covariance matrix of the target domain, γ, μ and λ are trade-off parameters for the graph-embedding term, the target variance term and $||A-B||_{\mathrm{F}}^2$, respectively.

3.3 Problem Optimization

Optimizing A and B. To optimize (8), we rewrite$[A;B]$ as P. Thus, the objective function can be rewritten as:

$$\min_{P} \frac{\mathrm{Tr}\left(P^{\mathrm{T}}\begin{bmatrix} M_{\mathrm{ss}}+\gamma S_{\mathrm{w}}^{\mathrm{s}}+\lambda I & M_{\mathrm{st}}-\lambda I \\ M_{\mathrm{ts}}-\lambda I & M_{\mathrm{tt}}+\gamma S_{\mathrm{w}}^{\mathrm{t}}+(\lambda+\mu)I \end{bmatrix} P\right)}{\mathrm{Tr}\left(P^{\mathrm{T}}\begin{bmatrix} \gamma S_{\mathrm{b}}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \gamma S_{\mathrm{b}}^{\mathrm{t}}+\mu S_{\mathrm{h}}^{\mathrm{t}} \end{bmatrix} P\right)}. \tag{10}$$

We can reformulate (10) as:

$$\max_{P} \mathrm{Tr}\left(P^{\mathrm{T}}\begin{bmatrix} \gamma S_{\mathrm{b}}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \gamma S_{\mathrm{b}}^{\mathrm{t}}+\mu S_{\mathrm{h}}^{\mathrm{t}} \end{bmatrix} P\right), \tag{11}$$

$$\text{s.t.} \quad \mathrm{Tr}\left(P^{\mathrm{T}}\begin{bmatrix} M_{\mathrm{ss}}+\gamma S_{\mathrm{w}}^{\mathrm{s}}+\lambda I & M_{\mathrm{st}}-\lambda I \\ M_{\mathrm{ts}}-\lambda I & M_{\mathrm{tt}}+\gamma S_{\mathrm{w}}^{\mathrm{t}}+(\lambda+\mu)I \end{bmatrix} P\right)=1 \; .$$

The Lagrange function of (11) is

$$\begin{aligned} L = &\mathrm{Tr}\left(P^{\mathrm{T}}\begin{bmatrix} \gamma S_{\mathrm{b}}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \gamma S_{\mathrm{b}}^{\mathrm{t}}+\mu S_{\mathrm{h}}^{\mathrm{t}} \end{bmatrix} P\right) \\ &+\mathrm{Tr}\left(\left(P^{\mathrm{T}}\begin{bmatrix} M_{\mathrm{ss}}+\gamma S_{\mathrm{w}}^{\mathrm{s}}+\lambda I & M_{\mathrm{st}}-\lambda I \\ M_{\mathrm{ts}}-\lambda I & M_{\mathrm{tt}}+\gamma S_{\mathrm{w}}^{\mathrm{t}}+(\lambda+\mu)I \end{bmatrix} P - I\right)\varPhi\right), \end{aligned} \tag{12}$$

where $\varPhi=\mathrm{diag}(\phi_1,\cdots,\phi_d)$ and $(\phi_1,\cdots,\phi_d)$ are the d largest eigenvalues of the following eigendecomposition problem:

$$\begin{bmatrix} \gamma S_{\mathrm{b}}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \gamma S_{\mathrm{b}}^{\mathrm{t}}+\mu S_{\mathrm{h}}^{\mathrm{t}} \end{bmatrix} P = \begin{bmatrix} M_{\mathrm{ss}}+\gamma S_{\mathrm{w}}^{\mathrm{s}}+\lambda I & M_{\mathrm{st}}-\lambda I \\ M_{\mathrm{ts}}-\lambda I & M_{\mathrm{tt}}+\gamma S_{\mathrm{w}}^{\mathrm{t}}+(\lambda+\mu)I \end{bmatrix} P\varPhi. \tag{13}$$

As a result, P consists of the corresponding d eigenvectors of the above problem. Once the transformation matrix P is obtained, the subspace A and B can be obtained easily.

Optimizing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. By regarding A and B as constants, the optimization of our objective function can be written as:

$$\min_{\alpha,\beta} \frac{1}{2}\alpha^{\mathrm{T}}K_{\mathrm{ss}}\alpha-\alpha^{\mathrm{T}}K_{\mathrm{st}}\beta, \tag{14}$$

$$\text{s.t.} \; \{\alpha_i^c,\beta_i^c\}\in[0,1], \frac{{\alpha^c}^{\mathrm{T}}\mathbf{1}}{n_{\mathrm{s}}^c}=\frac{{\beta^c}^{\mathrm{T}}\mathbf{1}}{n_{\mathrm{t}}^c}=\delta \; ,$$

where $(K_{\mathrm{ss}})_{i,j}$ in $K_{\mathrm{ss}} \in \mathbb{R}^{n_{\mathrm{s}} \times n_{\mathrm{s}}}$ is the coefficient associated with $(A^{\mathrm{T}} x_{\mathrm{s}}^{i})^{\mathrm{T}} A^{\mathrm{T}} x_{\mathrm{s}}^{i}$, and $(K_{\mathrm{st}})_{i,j}$ in $K_{\mathrm{st}} \in \mathbb{R}^{n_{\mathrm{s}} \times n_{\mathrm{t}}}$ is the coefficient associated with $(A^{\mathrm{T}} x_{\mathrm{s}}^{i})^{\mathrm{T}} B^{\mathrm{T}} x_{\mathrm{t}}^{j}$. Limited by space, the detailed derivations are omitted since they are similar to that in CDLS [15].

With the above formulation, we can apply Quadratic Programming (QP) solvers to optimize the equivalent problem:

$$\min_{z_i \in [0,1],\, Z^{\mathrm{T}} V = G} \quad \frac{1}{2} Z^{\mathrm{T}} B Z, \tag{15}$$

where

$$Z = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, B = \begin{pmatrix} K_{\mathrm{ss}} & -K_{\mathrm{st}} \\ -K_{\mathrm{st}}^{\mathrm{T}} & \mathbf{0} \end{pmatrix}, G \in \mathbb{R}^{1 \times 2C} \text{ with}$$

$$(G)_c = \begin{cases} \delta n_{\mathrm{s}}^{c} & \text{if c} \le \text{C} \\ \delta n_{\mathrm{t}}^{c-C} & \text{if c} > \text{C} \end{cases}, V = \begin{bmatrix} V_{\mathrm{s}} & \mathbf{0}_{n_{\mathrm{s}} \times C} \\ \mathbf{0}_{n_{\mathrm{t}} \times C} & V_{\mathrm{t}} \end{bmatrix} \in \mathbb{R}^{(n_{\mathrm{s}} + n_{\mathrm{t}}) \times 2C} \text{ with}$$

$$(V_{\mathrm{s}})_{ij} = \begin{cases} 1 & \text{if } x_{\mathrm{s}}^{i} \in \text{class j} \\ 0 & \text{otherwise} \end{cases}, (V_{\mathrm{t}})_{ij} = \begin{cases} 1 & \text{if } x_{\mathrm{t}}^{i} \text{ predicted as class j} \\ 0 & \text{otherwise} \end{cases}.$$

Finally, we use the weight sensitive libsvm [2] to train a classifier so that the learned weights can be fully exploited. We show the procedure of LDML in Algorithm 1.

3.4 Computational Complexity

Now we analyze the computational complexity of Algorithm 1 by the big O notation. As stated above, $X_{\mathrm{s}} \in \mathbb{R}^{m \times n_{\mathrm{t}}}$, $X_{\mathrm{t}} \in \mathbb{R}^{m \times n_{\mathrm{s}}}$, $X = [X_{\mathrm{s}}\ X_{\mathrm{t}}] \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times d}$, $B \in \mathbb{R}^{m \times d}$, where m is the original dimensionality, d is the dimensionality of the subspace, $n = n_{\mathrm{s}} + n_{\mathrm{t}}$ is the number of all samples. We note the number of classes as C and the number of iterations as T. The time cost of Algorithm1 consists of the following three parts:

(1) Solving the eigendecomposition problem in step 2 is $O(Tdm^2)$.
(2) Solving the equality constrained QP problems in step 5 is $O(Tn^3)$.
(3) Computing the MMD matrices and the graph embedding matrices in step 6 are both $O(TCn^2)$.

Then, the overall computational complexity of Algorithm 1 is $O(Tdm^2 + Tn^3 + TCn^2)$. In Sect. 4.5, we show that the number of iterations T is usually set as 5, which is enough to guarantee convergence. Besides, the typical values of d are not greater than 200, so $T \ll \min(m, n)$, $d \ll \min(m, n)$. Therefore, the computational complexity of Algorithm 1 depends mainly on the number of samples n and the dimensionality m.

```
Algorithm1: Learning Distribution-Matched Landmarks
Input: source and target domain data: $X_s, X_t$; labels for source domain data: $y_s$;
         Parameters: $\delta = 0.5$, $d$, $\lambda$, $\mu$, $\gamma$
Output: Predicted labels $y_t$ for target domain data
begin
      1: Initialize pseudo labels of target domain $\hat{y}_t$ using PCA; Construct $S_h^t$, $M_{ss}$, $M_{tt}$, $M_{st}$,
         $M_{ts}$, $S_b^s$, $S_w^s$, $S_b^t$, $S_w^t$ according to (9)(5)(6)(7);
   while not converge do
      2: Solve the generalized eigendecomposition problem in (13) and select $d$ corresponding
         eigenvectors of $d$ largest eigenvalues as the transformation $P$, and obtain transformation
         $A$ and $B$;
      3: Map the original data to respective subspace to get the embeddings: $Z_s = A^T X_s$,
         $Z_t = B^T X_t$;
      4: Train a SVM classifier on $\alpha, Z_s, y_s$ to update pseudo labels in target domain $\hat{y}_t$;
      5: Update landmark weights $\alpha, \beta$ by (15);
      6: Update $M_{ss}, M_{tt}, M_{st}, M_{ts}, S_b^t, S_w^t$ by (5)(6)(7);
   end while
end
```

4 Experiments

4.1 Datasets

Office + Caltech: Office [24] consists of three different datasets: **A**mazon (images downloaded from amazon.com), **W**ebcam (low-resolution images taken by a web camera), **D**SLR (high-resolution images taken by a digital SLR camera). **Caltech** [13] is a dataset which has 256 classes and 30,607 images. Ten common classes of all four datasets are selected [10] and each dataset is regarded as a domain. As a result, we have four domains: C (Caltech), A (Amazon), W (Webcam), D (DSLR) and 12 DA evaluations by selecting two different domains as the source and target domains respectively. We consider two types of features: SURF and DeCAF$_6$ [6]. SURF features are encoded with the 800-bin histogram with codebooks trained from a subset of Amazon images. DeCAF$_6$ are activation features of the 6th fully connected layer of a convolutional network constructed by [6].

PIE: PIE [25] is a face recognition benchmark which has 68 individuals with 41,368 face images. These individuals have different poses, expressions and illuminations. Five subsets of PIE are selected to conduct the face recognition experiments: C05 (left pose), C07 (upward pose), C09 (downward pose), C27 (frontal pose) and C29 (right pose). All face images are cropped and resized to 32×32 pixels and then converted to grayscale. Each subset is regarded as a domain, and each time one subset plays the role of the source domain and the other one plays the role of the target domain. Then, we can generate 20 DA evaluations.

4.2 Compared Baselines

We compare our LDML with the following six state-of-the-art baselines: transfering with SVM, subspace alignment (SA) [7], geodesic flow kernel (GFK) [10], joint distribution analysis (JDA) [20], transfer joint matching (TJM) [21], joint geometrical and statistical alignment (JGSA) [28].

4.3 Experimental Settings

Following the previous work [10,21], we normalize all of the data to have zero mean and unit standard deviation in each dimension.

For all the baselines, we report the best results we can achieve. For LDML, we fix the average weight $\delta = 0.5$. Both of the numbers of neighbors in the intrinsic graph and the penalty graph are 5. The number of iterations T is usually set to 5 except for testing the convergence of our method. For different experimental tasks, we set different hyperparameters to gain good performance. Specifically, when evaluating our method on Office + Caltech with SURF features, we set $\lambda = 0.5$, $\mu = 0.1$, $\gamma = 0.001$, $d = 40$. On Office + Caltech with DeCAF$_6$ features, we set $\lambda = 0.05$, $\mu = 0.5$, $\gamma = 0.01$, $d = 40$. On PIE dataset, we set $\lambda = 1$, $\mu = 0.1$, $\gamma = 0.02$, $d = 120$.

To exploit the learned weights for samples, our LDML uses the weight sensitive SVM to train a classifier. For fairness, we also use SVM to train classifiers on some baselines, e.g., SA, JDA, TJM and JGSA.

In this paper, we follow the previous work [10,22] and use the following equation as the classification accuracy:

$$\frac{|x : x \in X_{\mathrm{t}} \wedge \hat{y}_{\mathrm{t}} = y_{\mathrm{t}}|}{|x : x \in X_{\mathrm{t}}|},$$

where x is a sample in target domain, y_{t} is the real label for x, and $\hat{y}_{\mathrm{t}}$ is the predicted label for x.

4.4 Experimental Results

The results of LDML and other baseline methods on all the datasets are reported in Tables 1, 2 and 3.

On Office + Caltech with SURF features, LDML achieves the best classification accuracy on 8 out of 12 evaluations. Besides, the average accuracy of LDML is 55.81%, gaining an improvement of 3.43% compared with the best baseline JGSA. JGSA is a method that can reduce not only the distribution divergence but also the geometrical divergence simultaneously when projecting data onto low-dimensional subspaces. However, since the domain difference in Office + Caltech is substantially large, there may exist samples in one domain that are irrelevant to samples in the other domain even in subspaces learned by JGSA. LDML handles limitation of JGSA by selecting pivotal samples in both of the source and target domains, which contributes to the better performance compared with JGSA.

On Office + Caltech with DeCAF$_6$ features, all of the methods achieve better performance than that on Office + Caltech with SURF features, especially our LDML. LDML achieves the best classification accuracy on 10 out of 12 evaluations. It is also observed that LDML, JGSA and JDA achieve the best three average classification accuracies. The common point of these methods is that they all reduce both the marginal and conditional distribution divergences

Table 1. Accuracy (%) on the Office + Caltech dataset with SURF features.

X_s	X_t	SVM	GFK	SA	JDA	TJM	JGSA	LDML
C	A	54.18	41.02	51.04	54.28	51.77	58.87	**59.60**
	W	44.41	40.68	40.34	47.80	44.41	54.58	**58.98**
	D	43.95	38.85	45.86	45.86	49.68	45.22	**54.14**
A	C	**45.41**	40.25	44.79	43.46	43.99	40.69	45.24
	W	36.61	38.98	38.64	46.10	45.08	**60.34**	55.25
	D	37.58	36.31	39.49	40.76	45.22	**58.60**	52.87
W	C	33.04	30.72	35.71	32.23	35.89	31.79	**37.76**
	A	34.55	29.75	36.95	38.41	38.10	40.40	**43.11**
	D	84.08	80.89	71.97	84.08	82.17	84.71	**91.72**
D	C	30.01	30.28	34.11	33.30	**35.80**	34.11	35.44
	W	32.57	32.05	35.39	35.60	37.16	38.62	**44.47**
	A	73.90	75.59	76.61	82.03	84.75	80.68	**91.19**
Avg.		45.86	42.95	45.91	48.65	49.50	52.38	**55.81**

Table 2. Accuracy (%) on the Office + Caltech dataset with DeCAF$_6$ features.

X_s	X_t	SVM	GFK	SA	JDA	TJM	JGSA	LDML
C	A	89.46	89.04	90.61	90.60	90.92	92.07	**92.90**
	W	77.28	87.80	82.37	80.68	86.10	83.05	**89.49**
	D	80.25	85.99	86.62	84.71	87.90	88.54	**89.81**
A	C	81.21	78.45	84.06	85.04	82.46	85.57	**87.71**
	W	74.58	81.02	82.71	**87.46**	86.10	86.78	87.12
	D	82.80	80.25	86.62	**89.17**	87.90	87.90	87.26
W	C	66.34	72.31	76.58	82.37	79.52	85.40	**87.00**
	A	75.68	81.84	84.45	90.61	89.04	91.34	**91.44**
	D	99.36	**100**	99.36	99.36	98.73	99.36	**100**
D	C	64.74	75.96	73.73	72.31	73.91	74.98	**88.07**
	W	76.93	77.35	84.45	89.46	88.52	88.94	**93.11**
	A	97.29	97.29	94.24	97.29	94.58	**99.66**	**99.66**
Avg.		80.49	83.94	85.48	87.42	87.14	88.63	**91.13**

simultaneously by using MMD. TJM only considers the marginal distribution divergence, which leads to its relatively poor performance.

On PIE dataset, LDML outperforms state-of-the-art methods on all of the 20 evaluations. The average classification accuracy of LDML is 82.36%, which overwhelmingly has a 19.62% improvement compared with the best baseline JGSA. On PIE dataset, the Euclidean distance between samples from one domain but different classes may be smaller than that of samples within one class but from

Table 3. Accuracy (%) on the PIE dataset.

X_s	X_t	SVM	GFK	SA	JDA	TJM	JGSA	LDML
C05	C07	33.52	45.49	35.54	54.08	34.87	52.73	**80.11**
	C09	43.69	50.31	46.38	60.54	44.36	51.84	**72.49**
	C27	61.28	65.82	63.62	85.97	60.74	73.72	**95.31**
	C29	36.46	41.97	39.58	49.33	34.93	52.39	**64.64**
C07	C05	42.05	46.91	44.24	60.41	38.75	64.26	**80.34**
	C09	41.85	56.74	44.06	55.45	49.82	58.88	**75.49**
	C27	65.64	70.86	66.45	82.43	63.41	70.71	**94.86**
	C29	34.13	41.85	35.48	47.37	35.23	49.02	**69.73**
C09	C05	49.58	48.65	51.80	58.64	43.04	64.89	**78.75**
	C07	42.91	56.23	46.41	49.60	38.74	59.91	**79.19**
	C27	67.98	74.95	68.76	67.80	65.33	72.63	**96.00**
	C29	42.40	50.61	45.22	46.69	41.12	57.72	**74.69**
C27	C05	66.96	71.43	69.57	79.26	63.39	74.73	**95.98**
	C07	62.06	81.34	64.03	77.59	60.59	76.24	**96.13**
	C09	70.71	86.34	72.06	77.45	74.39	67.89	**94.24**
	C29	54.23	59.87	55.94	58.88	50.61	63.05	**84.74**
C29	C05	46.16	39.98	48.50	51.02	37.18	63.99	**77.46**
	C07	34.81	38.80	35.30	42.79	29.28	54.02	**69.06**
	C09	47.98	48.96	49.39	41.85	42.77	59.87	**77.63**
	C27	59.12	54.73	59.51	62.51	46.77	66.39	**90.36**
Avg.		50.17	56.59	52.09	60.48	47.77	62.74	**82.36**

different domains. The illustration can be seen in Fig. 2. To handle this, LDML constructs an intrinsic graph to preserve the local manifold structure, and a penalty graph to make samples with different labels more separable. This novel graph structure enhances the discriminability of the model and leads to the best performance of LDML on PIE dataset. We also notice that TJM performs poorly on PIE dataset, even not as good as SVM. TJM imposes $\ell_{2,1}$ norm on the source projection matrix to get the "row sparsity". On PIE dataset, $\ell_{2,1}$ norm reinforces the modal information among the faces with the same pose but from different people, and weaken the classification information among the faces from one person but with different poses. This results in low performance of TJM.

Above all, our LDML obviously performs the best on almost all evaluations, especially on PIE dataset. The only two exceptions are on A $\rightarrow$ W and A $\rightarrow$ D. As stated in [10], Amazon is greatly different from Webcam and DSLR geometrically and statistically. Besides, we also notice that Amazon is the unique dataset where images usually have no backgrounds. In this case, the MMD distances of almost all cross-domain sample pairs are so far that the selected landmarks have no obvious discrepancies with other samples. Even so, LDML still

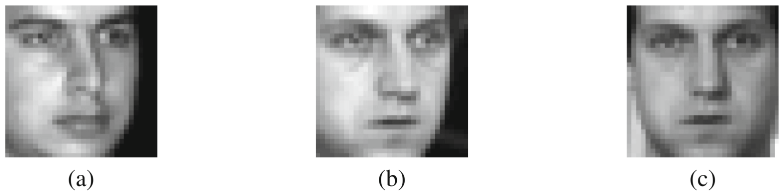

Fig. 2. Three faces are chosen from the CMU PIE dataset. Profile and frontal faces can be regarded as two domains. The faces of different people represent different classes. Figure 2(a) and (b) are two faces from the same domain but with different labels. Figure 2(b) and (c) are two faces with the same label but from different domains. We simply compute the Euclidean distance of two images. One is between the profile face of two classes in (a) and (b), the other is between two domains of one class in (b) and (c). We get that the distance of (a) and (b) is much smaller than that of (b) and (c), which is a common phenomenon in PIE dataset.

achieves the second-best performance on A $\rightarrow$ W(SURF), A $\rightarrow$ D (SURF) and A $\rightarrow$ W (DeCAF$_6$), which reveals the robustness of LDML.

4.5 Parameter Sensitivity and Convergence

To analyze the parameter sensitivity, we evaluate four hyperparameters: λ, μ, γ and d. When one of them is being evaluated, we set the others the same as parameters in Sect. 4.3. We conduct sensitivity analysis on C05 $\rightarrow$ C07 and C $\rightarrow$ A. The results are shown in Fig. 3.

In Fig. 3(a), when the dimensionality of subspace d varies from 20 to 200, the classification accuracies on Office + Caltech with both SURF and DeCAF$_6$ features fluctuate in small ranges. We observe that the classification accuracy on PIE is sensitive to d and the optimal range of d on PIE is between 100 and 120. In Fig. 3(b), it is obvious that the classification accuracy will reduce on all datasets if $\gamma > 0.01$. In Fig. 3(c), we can observe that the optimal λ for Office + Caltech is around 0.1, while the optimal λ for PIE is around 1. In Fig. 3(d), μ affects the performance on Office + Caltech dataset little when $\mu > 0.01$, while the optimal μ on PIE is around 0.1.

To show the convergence of our method, we report the values of the objective function on C $\rightarrow$ W (SURF), C $\rightarrow$ W (DeCAF$_6$) and C05 $\rightarrow$ C07 with different iterations. The results are illustrated in Fig. 4. It is obvious that the value of the objective function is monotonically decreasing when $T \leq 5$ and keeps stable when $T > 5$. This result reveals that LDML can converge in 5 iterations.

4.6 Effectiveness Analysis

To evaluate the effectiveness of landmark selection, we conduct experiments on Office + Caltech (SURF) with two settings: one is LDML with the same weight, the other is LDML with landmark selection. The results are reported

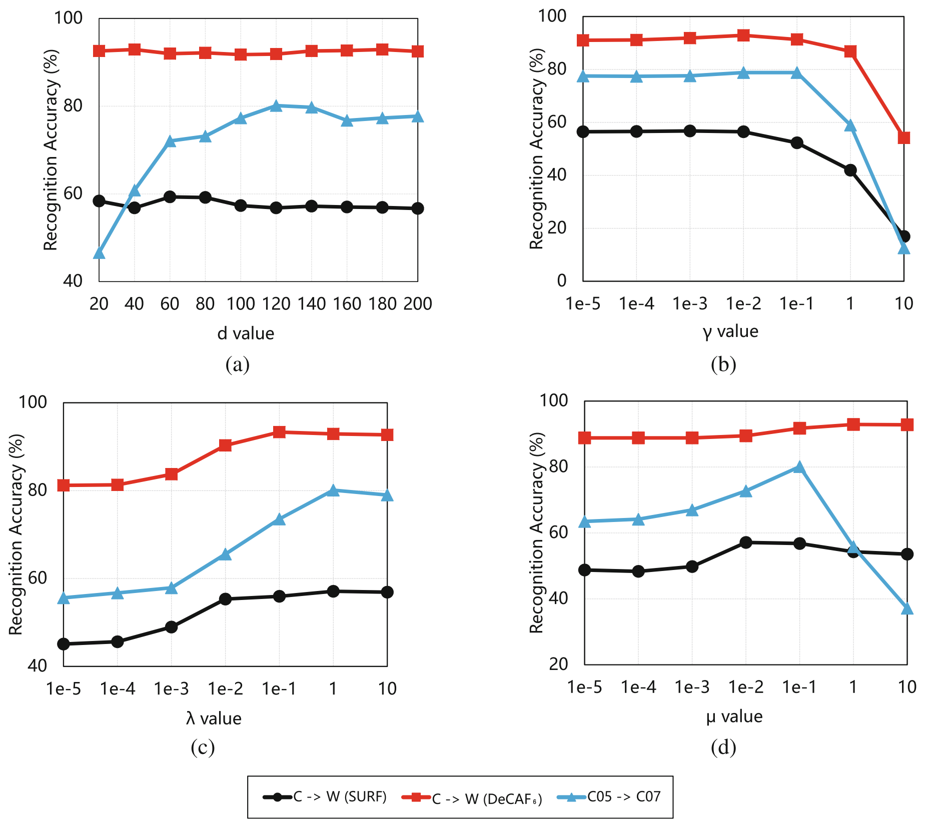

Fig. 3. Parameter sensitivity.

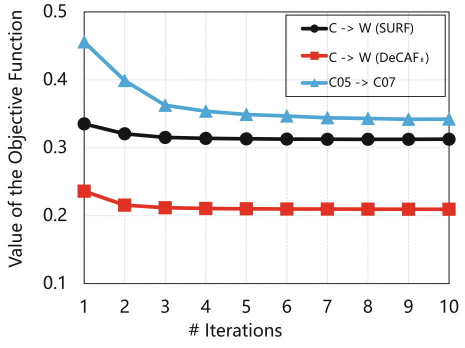

Fig. 4. Convergence analysis.

in Table 4. We observe that LDML with landmark selection achieves the better performance on 10 out of 12 evaluations. Besides, the average accuracy of LDML with landmark selection gains an improvement of 2.13% compared with that of

Table 4. Effectiveness analysis: accuracies (%) of two settings on Office + Caltech (SURF).

Method	C → A	C → W	C → D	A → C	A → W	A → D	-
No landmarks	58.35	**61.02**	**54.14**	43.01	50.51	47.77	-
With landmarks	**59.60**	58.98	**54.14**	**45.24**	**55.25**	**52.87**	-
Method	W → C	W → A	W → D	D → C	D → A	D → W	**Avg.**
No landmarks	30.81	41.02	**92.99**	31.26	42.07	**91.19**	53.68
With landmarks	**37.76**	**43.11**	91.72	**35.44**	**44.47**	**91.19**	**55.81**

LDML with the same weight. These results reveal the effectiveness of landmark selection in our method.

5 Conclusion

In this paper, we propose a novel method for unsupervised domain adaptation, named Learning Distribution-Matched Landmarks (LDML). LDML aligns the source and target domains by reducing the distribution divergence and selecting landmarks that bridge two domains. Comprehensive experiments on two real-world datasets verify the effectiveness of the proposed method. In the future, we plan to extend our LDML to be capable of handling the heterogeneous DA.

Acknowledgment. This work was supported in part by the National Postdoctoral Program for Innovative Talents under Grant BX201700045, China Postdoctoral Science Foundation under Grant 2017M623006, the Applied Basic Research Program of Sichuan Province under Grant 2015JY0124, and the Fundamental Research Funds for the Central Universities under Grant ZYGX2016J089.

References

1. Aljundi, R., Emonet, R., Muselet, D., Sebban, M.: Landmarks-based kernelized subspace alignment for unsupervised domain adaptation. In: CVPR, pp. 56–63 (2015)
2. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM TIST **2**(3), 27 (2011)
3. Chattopadhyay, R., Sun, Q., Fan, W., Davidson, I., Panchanathan, S., Ye, J.: Multisource domain adaptation and its application to early detection of fatigue. KDD **6**(4), 18 (2012)
4. Chen, H.T., Chang, H.W., Liu, T.L.: Local discriminant embedding and its variants. In: CVPR, vol. 2, pp. 846–853. IEEE (2005)
5. Ding, Z., Shao, M., Fu, Y.: Deep low-rank coding for transfer learning. In: IJCAI, pp. 3453–3459 (2015)
6. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: DeCAF: a deep convolutional activation feature for generic visual recognition. In: ICML, pp. 647–655 (2014)

7. Fernando, B., Habrard, A., Sebban, M., Tuytelaars, T.: Unsupervised visual domain adaptation using subspace alignment. In: ICCV, pp. 2960–2967 (2013)
8. Ghifary, M., Balduzzi, D., Kleijn, W.B., Zhang, M.: Scatter component analysis: a unified framework for domain adaptation and domain generalization. IEEE TPAMI **39**(7), 1414–1430 (2016)
9. Gong, B., Grauman, K., Sha, F.: Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation. In: ICML, pp. 222–230 (2013)
10. Gong, B., Shi, Y., Sha, F., Grauman, K.: Geodesic flow kernel for unsupervised domain adaptation. In: CVPR, pp. 2066–2073. IEEE (2012)
11. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: an unsupervised approach. In: ICCV, pp. 999–1006. IEEE (2011)
12. Gretton, A., Borgwardt, K.M., Rasch, M., Schölkopf, B., Smola, A.J.: A kernel method for the two-sample-problem. In: NIPS, pp. 513–520 (2007)
13. Griffin, G., Holub, A., Perona, P.: Caltech-256 Object Category Dataset (2007)
14. Gu, Q., Li, Z., Han, J.: Joint feature selection and subspace learning. In: IJCAI, vol. 22, p. 1294 (2011)
15. Hubert Tsai, Y.H., Yeh, Y.R., Frank Wang, Y.C.: Learning cross-domain landmarks for heterogeneous domain adaptation. In: CVPR, pp. 5081–5090 (2016)
16. Li, J., Lu, K., Huang, Z., Shen, H.T.: Two birds one stone: on both cold-start and long-tail recommendation. In: ACM Multimedia, pp. 898–906. ACM (2017)
17. Li, J., Lu, K., Zhu, L., Li, Z.: Locality-constrained transfer coding for heterogeneous domain adaptation. In: Huang, Z., Xiao, X., Cao, X. (eds.) ADC 2017. LNCS, vol. 10538, pp. 193–204. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68155-9_15
18. Li, J., Wu, Y., Zhao, J., Lu, K.: Low-rank discriminant embedding for multiview learning. IEEE TCYB **47**(11), 3516–3529 (2017)
19. Li, J., Zhao, J., Lu, K.: Joint feature selection and structure preservation for domain adaptation. In: IJCAI, pp. 1697–1703 (2016)
20. Long, M., Wang, J., Ding, G., Sun, J., Yu, P.S.: Transfer feature learning with joint distribution adaptation. In: ICCV, pp. 2200–2207 (2013)
21. Long, M., Wang, J., Ding, G., Sun, J., Yu, P.S.: Transfer joint matching for unsupervised domain adaptation. In: CVPR, pp. 1410–1417 (2014)
22. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE TNN **22**(2), 199–210 (2011)
23. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE TKDE **22**(10), 1345–1359 (2010)
24. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 213–226. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_16
25. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database. In: FG, pp. 53–58. IEEE (2002)
26. Sugiyama, M.: Local fisher discriminant analysis for supervised dimensionality reduction. In: ICML, pp. 905–912. ACM (2006)
27. Yan, S., Xu, D., Zhang, B., Zhang, H.J., Yang, Q., Lin, S.: Graph embedding and extensions: a general framework for dimensionality reduction. IEEE TPAMI **29**(1), 40–51 (2007)

28. Zhang, J., Li, W., Ogunbona, P.: Joint geometrical and statistical alignment for visual domain adaptation. In: CVPR (2017)
29. Zhou, M., Chang, K.C.: Unifying learning to rank and domain adaptation: enabling cross-task document scoring. In: SIGKDD, pp. 781–790. ACM (2014)
30. Zhu, L., Shen, J., Jin, H., Zheng, R., Xie, L.: Content-based visual landmark search via multimodal hypergraph learning. IEEE TCYB **45**(12), 2756–2769 (2015)