Domain Adaption Based on Symmetric Matrices Space Bi-Subspace Learning and Source Linear Discriminant Analysis Regularization

QIAN LI1, ZHENGMING MA1, SHUYU LIU2, AND YANLI PEI1

1School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China
2Public Experimental Teaching Center, Sun Yat-sen University, Guangzhou 510006, China

Corresponding author: Zhengming Ma (issmzm@mail.sysu.edu.cn)

This work was supported in part by the Science and Technology Program of Guangzhou, China, under Grant 68000-42050001, and in part by the Natural Science Foundation of China under Grant 61773022.

ABSTRACT At present, Symmetric Positive Definite (SPD) matrix data is the most common non-Euclidean data in machine learning. Because SPD data don’t form a linear space, most machine learning algorithms can not be carried out directly on SPD data. The first purpose of this paper is to propose a new framework of SPD data machine learning, in which SPD data are transformed into the tangent spaces of Riemannian manifold, rather than a Reproducing Kernel Hilbert Space (RKHS) as usual. Domain adaption learning is a kind of machine learning. The second purpose of this paper is to apply the proposed framework to domain adaption learning (DAL), in which the architecture of bi-subspace learning is adopted. Compared with the commonly-used one subspace learning architecture, the proposed architecture provides a broader optimization space to meet the domain adaption criterion. At last, in order to further improve the classification accuracy, Linear Discriminant Analysis (LDA) regularization of source domain data is added. The experimental results on five real-world datasets demonstrate the out-performance of the proposed algorithm over other five related state-of-the-art algorithms.

INDEX TERMS Domain adaptation learning, Riemannian manifold, bi-subspace learning, symmetric positive definite, source linear discriminant analysis regularization.

I. INTRODUCTION

Traditional machine learning methods usually work well on image recognition tasks such as face recognition and object recognition only under a common assumption: the marginal probability distributions of training data and test data are the same or similar [1]. However, in practical applications, changes of some factors in images such as illumination conditions, background, visual angles, posture, and image resolution will mismatch the distributions of training data and test data. DAL algorithms aim to reduce the difference of distributions between the training data and the test data [2].

DAL tasks usually consist of two datasets. One is called source domain data, which contains a large amount of labeled data. The other dataset is called target domain data, and its marginal probability distribution is often different from the source domain data which means that the representations of features are different. Target domain data are often unlabeled, or only a part of them have labels. Therefore, there are usually two types of DAL tasks, one is called semi-supervised DAL [2]–[4], which means that a part of the target domain data is labeled and we can use the source domain data and this part of the labeled target domain data to learn the intrinsic properties of data during the domain shift. The other DAL task we called unsupervised DAL [5]–[9], which means that all of the target domain data are unlabeled and we can only exploit the target domain data without label information to achieve DA [10]–[12]. This paper only focuses on the unsupervised DAL problem. Because in the real world, unlabeled data can be obtained in large quantities in a variety of ways, but it is difficult to identify their labels. In this sense, the unsupervised DAL algorithms are more applicable for the realistic object recognition application scenarios [7]–[9].

Unsupervised DAL algorithms usually focus on looking for a new feature space of data. In this space, the source
domain and the target domain data share similar feature representations, which means that the distributions of two domains in this space are matched. Thus, in this space, the classifier trained by the source domain data can still work well on the target domain. At present, the most commonly used space is the Reproducing Kernel Hilbert Space (RKHS).

It is worth noting that most of the advanced DAL algorithms still concentrate on dealing with Euclidean data. The original data that those algorithms processed are feature vectors. However, nowadays, visual data such as images and videos are increasingly complex and varied. Euclidean feature vectors are no longer able to represent those massive amounts of information precisely and efficiently. Recent researches suggest that a wide range of computer vision problems can be addressed more appropriately by considering non-Euclidean geometry, such as Riemannian manifolds. After some kinds of feature extractions, the original visual data can be transformed onto Riemannian manifolds. The data on the Riemannian manifolds can maintain the Riemannian geometric structure of the original visual data. Many practical visual recognition tasks such as image set classification [13], [14], video face recognition [15], [16], and visual tracking [17] also show that the Riemannian data are more discriminative than the Euclidean data. Therefore, Riemannian manifolds, especially Symmetric Positive Definite (SPD) manifolds, have been applied widely in computer vision tasks. SPD data (usually represented as SPD matrices) embedded on SPD manifold can encode regional image features and more and more researchers propose algorithms to replace the Euclidean data with SPD data as the data descriptors. Riemannian manifolds already have good performance in some machine learning researches such as dictionary learning [18]–[20], kernel learning [21], [22], metric learning [23], discriminative analysis [24], [25], dimensionality reduction of data [26], [27] and so on.

However, in the field of DAL, there are still very few relevant researches on such Riemannian data. The reason is that although Riemannian manifolds such as SPD manifold and Grassmann manifold are metric spaces, they are not linear spaces, which means that linear operations cannot be directly performed on them. Therefore, traditional machine learning algorithms, which require not only metrics but also linear operations, cannot be directly used on Riemannian manifolds [28], [29], and it is also the reason why the current DAL algorithms still deal with the Euclidean data. Whereas, different from those algorithms, the DAL algorithm proposed in this paper directly processes the Riemannian data (SPD data), which means that we extend DAL algorithm from Euclidean space to Riemannian manifold (SPD manifold).

The biggest challenge of SPD data is the challenge of non-Euclidean data structure, in the past many researchers applied traditional Euclidean learning methods directly to SPD data, resulting in the adverse effects of tensor diffusion expansion effect [30], [31], which makes the algorithm performance worse. Affine Invariant Riemannian Metric (AIRM) [30], the logarithmic Euclidean metric [31], the projection metric (PM) [32] are classic metrics. In order to overcome the limitations of European representation, the research on the application of these metric methods to SPD data has also been developed recently, effectively realizing the extension of the traditional Euclidean data learning algorithm to the manifold of learning different SPD data. SPD data are usually processed by machine learning algorithms in two ways to solve the problems of the linear operations, one is to locally flatten them via tangent spaces [23], [33]. And the other is to map them into the RKHS [20], [22] using the kernel trick, which means that the SPD matrix is transformed into a function defined on the whole SPD manifold and this is the most common method. Instead of adopting the RKHS as the workspace to perform DAL on SPD manifold, we explore a new space, called Symmetric Matrices Space (SMS). The data we transformed into SMS are symmetric positive semi-definite matrices rather than functions as those in the RKHS and the transformation is almost an identical transformation, which means that the original form of the data can be preserved in SMS. SMS is an inner product space (a linear space equipped with an inner product) and supports linear operations and distance measurements. Therefore, from the perspective of solving the problems of linear operations on SPD manifold, SMS is the least linear expansion from SPD manifold and seems like a more natural and suitable workspace than the widely-used RKHS. SMS has been used for the machine learning problems on the Grassmann manifold [34], [35], but seems never for SPD manifold. The algorithm we proposed may be the first attempt to exploit SMS as the workspace figuring out the DAL problems on SPD manifold.

The contributions of this paper are summarized as follows:

1. Non-Euclidean data are becoming more and more common in machine learning. Non-Euclidean datasets do not form a linear space and therefore do not support linear operations. However, most machine learning algorithms involve a large number of linear operations. Therefore, these machine learning algorithms cannot be carried out directly on non-Euclidean datasets. At present, the commonly-used methods are to transform the non-Euclidean dataset to RKHS, and then carry out machine learning in RKHS. This is the framework of the so-called kernel tricks in machine learning. The transformation from non-Euclidean dataset to RKHS is a nonlinear transformation, in which a non-Euclidean datum is transformed into a function defined on the entire non-Euclidean dataset. Under such transformation, the non-Euclidean datum has undergone significant changes in form and nature. Machine learning on the transformed dataset may not be the original intention. Although a non-Euclidean dataset does not form a linear space, it can often form a Riemannian manifold and the tangent space of Riemannian manifold is a finite-dimensional Hilbert space, that is, Euclidean space. Therefore, this paper proposes a new
framework for machine learning of non-Euclidean datasets, in which, with the help of Riemannian manifold, a non-Euclidean dataset can be transformed into a tangent space of its Riemannian manifold, that is, Euclidean space, and then machine learning can be transferred from the non-Euclidean dataset to Euclidean space. This framework is different from the framework of RKHS methods and has universality (Note: RKHS is an infinite-dimensional Hilbert space and therefore not a Euclidean space).

(2) SPD datasets are typical non-Euclidean datasets and they can become Riemannian manifolds under certain topology and Riemannian metrics. In particular, the tangent space of SPD manifold is a symmetric matrix space and the symmetric matrices include SPD matrices. Therefore, the tangent space of SPD manifold is actually the minimum linear expansion of SPD manifold, that is, the minimum linear space containing SPD manifold. The algorithm proposed in this paper uses log transformation to transform SPD matrix into the symmetric matrix, in which the transformation is realized only through the logarithm of eigenvalues in the eigendecomposition formula of SPD matrix. Such transformation keeps the symmetry property of SPD matrix unchanged and the change to SPD matrix as minimal as possible.

(3) In this paper, the proposed framework of non-Euclidean data machine learning based on Riemannian manifold tangent space is applied to domain adaption learning, in which the SPD source and target domain data are first transformed into the tangent space of SPD manifold, and then two subspaces in the tangent space are learned so as to make the probability distributions of source and target domain data as close as possible when they are projected into these two subspaces. The advantage of the bi-subspace setting is to expand the optimization space and avoid possible local extremum traps. More importantly, different regularization terms can be added according to different characteristics of source and target domain data. In this paper, the LDA regularization term of source domain subspace is added.

The rest of this paper is organized as follows. In Chapter II, we briefly introduce the related concepts and mathematical properties. In Chapter III, some related works are reviewed. In Chapter IV, we propose SMSBL-DA. In Chapter V, we compare five advanced DAL algorithms in theory. In Chapter VI, we describe the five classification experiments on five real-world datasets. Finally, we conclude our work in Chapter VII.

II. PRELIMINARIES

A. RIEMANNIAN MANIFOLD

The Riemannian metric can be defined on the differential manifold so that the differential manifold becomes a Riemannian manifold. The so-called Riemannian metric is a symmetric, positive definite, and smooth second-order tensor field on the differential manifold. If we define an inner product for the tangent space of each point on the differential manifold, that is, asymmetric, positive definite second-order tensor field (inner product field) is defined on the differential manifold. Therefore, the difference between the inner product field and the Riemannian metric lies in whether the inner product field is smooth on the differential manifold.

The differential manifold that defines the Riemann metric is called the Riemann manifold. Using Riemannian metric, we can define the length of any curve on the Riemannian manifold, and then you can define the distance between any two points on the Riemannian manifold, which is called the geodesic distance. The geodesic distance satisfies the distance of three kilometers, therefore, the Riemannian manifold is a metric (distance) space.

Since the tangent space of each point on the differential manifold is a finite-dimensional linear space, once the inner product is defined, it becomes a finite-dimensional inner product space, and the finite-dimensional inner product space (Hilbert space) is complete, so, the tangent space of each point on the Riemannian manifold is a finite-dimensional Hilbert space. The finite-dimensional Hilbert space is isomorphic with the same-dimensional European space. Therefore, the tangent space at each point of Riemannian manifold is essentially European space.

B. PROJECTION THEOREM IN HILBERT SPACES

Define \( (H, \langle \cdot, \cdot \rangle) \) is a Hilbert space, here \( \langle \cdot, \cdot \rangle \) is \( H \) inner product, define \( G \subseteq H \) is a \( d \)-dimensional subspace of \( H \), \( \{e_1, \cdots, e_d\} \subseteq H \) is the standard orthogonal basis of \( G \), \( x \in H \), then the projection \( z \) of \( x \) in the subspace \( G \) is:

\[
    z = \sum_{i=1}^{d} \langle x, e_i \rangle \in G
\]

and \( y = \begin{bmatrix} \langle x, e_1 \rangle \\ \vdots \\ \langle x, e_d \rangle \end{bmatrix} \in \mathbb{R}^d \) is the coordinates of the standard orthogonal basis of projection \( z \) in the subspace \( G \).

C. LOG TRANSFORM OF SPD MATRICES

Define \( X \in \mathbb{R}^{D \times D} \) is a symmetric positive definite matrix, then can be expressed as:

\[
    X = U \Sigma U^T
\]

where \( U \in \mathbb{R}^{D \times D} \), \( UU^T = I_D \), \( \Sigma = \text{diag}(\lambda_1, \cdots, \lambda_D) \), \( \lambda_i > 0 \) and \( i = 1, \cdots, D \). Define

\[
    \log(X) = U \text{diag}(\log(\lambda_1), \cdots, \log(\lambda_D)) U^T \in \mathbb{R}^{D \times D}
\]

Obviously, \( \log(X) \) is a symmetric matrix.

D. FROBENIUS INNER PRODUCT AND NORM OF MATRICES

Define \( K = \{A | A \in \mathbb{R}^{n \times m}\} \), obviously, \( K \) is a linear space in the real number domain. Further, define the inner product: for any \( A, B \in K \),

\[
    \langle A, B \rangle_F = \text{tr}(AB^T) = \text{tr}(A^T B)
\]
Here $tr(\bullet)$ represents the trace of matrix. Obviously, $(\bullet, \bullet)_F$ satisfies the condition of inner product, namely symmetric, positive definite and bilinear. $(\bullet, \bullet)_F$ is called the Frobenius inner product of matrices, and norm defined by Frobenius inner product is called Frobenius norm of matrices, denoted as $\|\bullet\|_F = \sqrt{(\bullet, \bullet)_F}$.

Remark 1: $(A, B)_F$ essentially means that the corresponding elements of the two matrices are multiplied and then added. Therefore, if two matrices $A$ and $B$ are expressed as vectors, then $(A, B)_F = A^T B$.

### III. RELATED WORKS

#### A. DOMAIN ADAPTATION BASED ON SUBSPACE

Existing DAL algorithms for visual recognition tasks can be roughly divided into two types, where the first type is based on subspace [36]–[41], and the second type is based on sample reweighting [42], [43]. DAL algorithms based on sample reweighting aim to compare the distributions of two domains in the original feature space. But the feature space may not be appropriate for the specific task targeted, because the features extracted from images may be distorted during the domain shift. And some features may only be related to a specific domain and not applicable to other domains [44]. Therefore, the DAL algorithms based on subspace are more widely studied and achieved greater success. Our algorithm is also a kind of subspace-based algorithm. Thus, we will introduce some subspace-based DAL algorithms in the following paragraphs.

In 2008, Pan et al. [45] first applied Maximum Mean Difference (MMD) criterion to DAL, proposing Maximum Mean Discrepancy Embedding (MMDE) dimensionality reduction domain adaption algorithm. MMD is a common criterion for measuring the probability distribution differences between the source and target domain data, and the MMDE algorithm uses kernel functions to map data to the RKHS and then uses MMD criterion in the RKHS to build a learning model to learn a kernel space that can reduce the difference in data distribution between the source and target domains while maintaining the variance of the data. MMDE learns kernel functions from data, resulting in the disadvantage that their computational costs are too high. Thus, based on MMDE, Pan et al. [46] proposed a new learning method for domain adaptation in 2011, i.e., Transfer Component Analysis (TCA). In order to solve the problem that TCA is only applicable to unsupervised learning, Pan et al. [46] proposed the semi-supervised learning method, i.e., Semi-Supervised Transfer Component Analysis (SSTCA) in 2011. More specifically, in terms of optimizing TCA, they choose to minimize probability distribution differences between source and target domain data as small as possible, while retaining the local geometry of the labeled source domain data. At the same time, the local geometric structure of the source domain data with labels should be preserved, and it is expected that the problems in the target domain can be solved with the knowledge obtained in the source domain. They introduced SSTCA-based category information to minimize in-class distance. In this sense, IGLDA can be seen as an extension of SSTCA, but there are essential differences between the two algorithms. SSTCA needs to make rich use of the dependencies between labels and samples, so it takes advantage of the covariance between labels and mapped data. However, IGLDA does not need to formulate this constraint, and the dependency relationship between labels and data can be improved well by bringing similar samples close together and better distinguishing samples in public space. Furthermore, in 2019, Li et al. [48] proposed a common framework for unsupervised heterogeneous domain adaptation, called Transfer Independently Together (ITT). They theoretically studied two strategies, namely distribution matching and knowledge adaptation. The distribution matching section is used to better handle heterogeneous data, and the knowledge adaptation section is used to address the issue of transferring as much knowledge as possible in an unsupervised domain adaption learning approach. In 2020, Zhang et al. [49] proposed an unsupervised domain adaptive algorithm based on subspace learning, called Guide Subspace Learning (GSL). The main idea of GSL is to use two different projection matrices to project the source domain data and the target domain data into their respective subspaces.

#### B. MACHINE LEARNING BASED ON MANIFOLD

Manifold-based machine learning methods [50], [51] have been successfully studied in recent years, and the basic idea is to apply principal component analysis (PCA) to domain adaption to obtain the geodesic line distance between the two points on the Grassman manifold. Then, randomly collecting samples along the geodesic line and constructing constraints on subspaces. Finally, mapping the data in the source domain to the subspace and learning the sample in the subspace. This method is the basis of many studies, based on which researchers proposed several effective manifold-based methods for domain adaptation. Fernando et al. [52] in 2013 proposed a method called subspace alignment, where source and...
target domain data are represented in subspaces created by feature vectors, which is similar to PCA. Ho and Gopalan [53] deduced a potential domain (or subspace) in 2014 by using points to represent objects. More specifically, based on the study of the tensor space, they characterize the potential domain with a product of Grassman manifold and successfully apply it to video and image classification tasks. In 2014, Cui et al. [54] presented a method that differs from the literature in which the representation of the domain is expressed directly in a covariance matrix, looking at the geodesic distance between the source domain data and the target domain data on the Riemannian manifold. The learning sample is then mapped to the middle area of the geodesic line so that the domain invariant features can be extracted.

Various machine learning methods based on Riemannian manifold have achieved great success in recent years, and the success of classification applications based on Riemannian manifolds is largely due to the introduction of metric learning. To extend metric learning study of SPD matrices, Zhou et al. [23] proposed the CERML framework to learn this sub-manifold with the bilinear sub-manifold learning (BSML) algorithm. In order to solve the problem of measuring the distance between two heterogeneous spaces, Huang et al. [21] proposed the General Learning Vector Method (GLVQ), where the characteristic transformation, that is, the geodetic distance between two heterogeneous spaces, they characterize the potential domain (or subspace) in 2014 by using the usual approach is to transform SPD data into a linear space. Currently, the most commonly used linear space is the tangent space of SPD manifold. We use transformation log : Sym_++^D → Sym^D. For any X ∈ Sym_++^D, because X = U diag(λ_1, . . . , λ_D) UT, where λ_1 ≥ · · · ≥ λ_D > 0 is the eigenvalue of X, and U is an orthogonal matrix composed of the normal orthogonal eigenvectors of X, so it is defined:

\[ \log(X) = U \text{diag}(\log(\lambda_1), \ldots, \log(\lambda_D)) U^T \in \text{Sym}^D \] (5)

Some documents also use log transformation, but log transformation has the following two characteristics that seem to be unmentioned:

1. Under the AIRM [30], log transformation is an equidistant transformation, that is, the geodetic distance between X and the identity matrix is equal to the Euclidean distance between log(X) and the origin of the tangent space. The AIRM of SPD manifold is as follows: Let X ∈ Sym_++^D, T_X(Sym_++^D) be the tangent space of X, then for any \( \Phi, \Theta \in T_X(Sym_++^D) \),

\[ \langle \Phi, \Theta \rangle_X = \left\langle X^{-1/2} \Phi X^{-1/2}, X^{-1/2} \Theta X^{-1/2} \right\rangle_F \] (6)

So we have:

\[ \text{geoD}(X, Y) = \left\| \log \left( Y^{-1/2} XY^{-1/2} \right) \right\|_F \] (9)

If X = I_D (identity matrix), then the inner product and the Euclidean distance in T_{I_D}(Sym_++^D) are as follows:

\[ \langle \Phi, \Theta \rangle_{I_D} = \langle \Phi, \Theta \rangle_F \] (7)

\[ \| \Phi - \Theta \|_{I_D}^2 = \langle \Phi - \Theta, \Phi - \Theta \rangle_F = \| \Phi - \Theta \|_F^2 \] (8)

The geodesic distance induced by the AIRM is:

\[ \text{geoD}(X, I_D) = \| \log(X) \|_F = \| \log(X) \|_{I_D} \] (10)

(2) Since SPD data does not constitute a linear space, the usual approach is to transform SPD data into a linear space. Currently, the most commonly used linear space is the

IV. DOMAIN ADAPTATION BASED ON SYMMETRIC MATRICES SPACE BI-SUBSPACE LEARNING AND SOURCE LINEAR DISCRIMINANT ANALYSIS REGULARIZATION

A. THE ISOMETRY TRANSFORM BETWEEN SPD MANIFOLD AND ITS TANGENT SPACE

Differential manifold is just a topological space, there is no distance metric between elements, let alone linear operation. Although the tangent space of differential manifold supports linear operation, there is no distance metric between tangent vectors. In order to support machine learning, a Riemannian metric must be assigned to the differential manifold, making it a Riemannian manifold. SPD manifold is one of the most commonly used Riemannian manifolds in machine learning. The space composed of D × D SPD matrices is called SPD manifold, denoted as Sym_++^D and whole D × D symmetric matrix set is expressed as Sym^D. It can be proved that for any X ∈ Sym_++^D, the tangent space of the SPD manifold and SMS are linearly isomorphic [29], that is T_X(Sym_++^D) = Sym^D. SPD manifold, from the perspective of Riemannian manifold, the tangent space of each point is the inner product space, and the finite-dimensional inner product space is complete, that is, the tangent space of each point on the Riemannian manifold is a finite-dimensional Hilbert space. The finite-dimensional Hilbert spaces are all isomorphic with the same-dimensional European spaces. Therefore, the tangent space of each point on the Riemannian manifold is essentially a European space.

To transfer machine learning from SPD manifold to its tangent space, we must first transform the learning samples of SPD data to the tangent space of SPD manifold. We use transformation log : Sym_++^D → Sym^D. For any X ∈ Sym_++^D, because X = U diag(λ_1, . . . , λ_D) UT, where λ_1 ≥ · · · ≥ λ_D > 0 is the eigenvalue of X, and U is an orthogonal matrix composed of the normal orthogonal eigenvectors of X, so it is defined:

\[ \log(X) = U \text{diag}(\log(\lambda_1), \ldots, \log(\lambda_D)) U^T \in \text{Sym}^D \] (5)

Some documents also use log transformation, but log transformation has the following two characteristics that seem to be unmentioned:

1. Under the AIRM [30], log transformation is an equidistant transformation, that is, the geodetic distance between X and the identity matrix is equal to the Euclidean distance between log(X) and the origin of the tangent space. The AIRM of SPD manifold is as follows: Let X ∈ Sym_++^D, T_X(Sym_++^D) be the tangent space of X, then for any \( \Phi, \Theta \in T_X(Sym_++^D) \),

\[ \langle \Phi, \Theta \rangle_X = \left\langle X^{-1/2} \Phi X^{-1/2}, X^{-1/2} \Theta X^{-1/2} \right\rangle_F \] (6)

If X = I_D (identity matrix), then the inner product and the Euclidean distance in T_{I_D}(Sym_++^D) are as follows:

\[ \langle \Phi, \Theta \rangle_{I_D} = \langle \Phi, \Theta \rangle_F \] (7)

\[ \| \Phi - \Theta \|_{I_D}^2 = \langle \Phi - \Theta, \Phi - \Theta \rangle_F = \| \Phi - \Theta \|_F^2 \] (8)

The geodesic distance induced by the AIRM is:

\[ \text{geoD}(X, Y) = \left\| \log \left( Y^{-1/2} XY^{-1/2} \right) \right\|_F \] (9)

So we have:

\[ \text{geoD}(X, I_D) = \| \log(X) \|_F = \| \log(X) \|_{I_D} \] (10)

(2) Since SPD data does not constitute a linear space, the usual approach is to transform SPD data into a linear space. Currently, the most commonly used linear space is the

IV. DOMAIN ADAPTATION BASED ON SYMMETRIC MATRICES SPACE BI-SUBSPACE LEARNING AND SOURCE LINEAR DISCRIMINANT ANALYSIS REGULARIZATION

A. THE ISOMETRY TRANSFORM BETWEEN SPD MANIFOLD AND ITS TANGENT SPACE

Differential manifold is just a topological space, there is no distance metric between elements, let alone linear operation. Although the tangent space of differential manifold supports linear operation, there is no distance metric between tangent vectors. In order to support machine learning, a Riemannian metric must be assigned to the differential manifold, making it a Riemannian manifold. SPD manifold is one of the most commonly used Riemannian manifolds in machine learning. The space composed of D × D SPD matrices is called SPD manifold, denoted as Sym_++^D and whole D × D symmetric matrix set is expressed as Sym^D. It can be proved that for any X ∈ Sym_++^D, the tangent space of the SPD manifold and SMS are linearly isomorphic [29], that is T_X(Sym_++^D) = Sym^D. SPD manifold, from the perspective of Riemannian manifold, the tangent space of each point is the inner product space, and the finite-dimensional inner product space is complete, that is, the tangent space of each point on the Riemannian manifold is a finite-dimensional Hilbert space. The finite-dimensional Hilbert spaces are all isomorphic with the same-dimensional European spaces. Therefore, the tangent space of each point on the Riemannian manifold is essentially a European space.

To transfer machine learning from SPD manifold to its tangent space, we must first transform the learning samples of SPD data to the tangent space of SPD manifold. We use transformation log : Sym_++^D → Sym^D. For any X ∈ Sym_++^D, because X = U diag(λ_1, . . . , λ_D) UT, where λ_1 ≥ · · · ≥ λ_D > 0 is the eigenvalue of X, and U is an orthogonal matrix composed of the normal orthogonal eigenvectors of X, so it is defined:

\[ \log(X) = U \text{diag}(\log(\lambda_1), \ldots, \log(\lambda_D)) U^T \in \text{Sym}^D \] (5)

Some documents also use log transformation, but log transformation has the following two characteristics that seem to be unmentioned:

1. Under the AIRM [30], log transformation is an equidistant transformation, that is, the geodetic distance between X and the identity matrix is equal to the Euclidean distance between log(X) and the origin of the tangent space. The AIRM of SPD manifold is as follows: Let X ∈ Sym_++^D, T_X(Sym_++^D) be the tangent space of X, then for any \( \Phi, \Theta \in T_X(Sym_++^D) \),

\[ \langle \Phi, \Theta \rangle_X = \left\langle X^{-1/2} \Phi X^{-1/2}, X^{-1/2} \Theta X^{-1/2} \right\rangle_F \] (6)

If X = I_D (identity matrix), then the inner product and the Euclidean distance in T_{I_D}(Sym_++^D) are as follows:

\[ \langle \Phi, \Theta \rangle_{I_D} = \langle \Phi, \Theta \rangle_F \] (7)

\[ \| \Phi - \Theta \|_{I_D}^2 = \langle \Phi - \Theta, \Phi - \Theta \rangle_F = \| \Phi - \Theta \|_F^2 \] (8)

The geodesic distance induced by the AIRM is:

\[ \text{geoD}(X, Y) = \left\| \log \left( Y^{-1/2} XY^{-1/2} \right) \right\|_F \] (9)

So we have:

\[ \text{geoD}(X, I_D) = \| \log(X) \|_F = \| \log(X) \|_{I_D} \] (10)

(2) Since SPD data does not constitute a linear space, the usual approach is to transform SPD data into a linear space. Currently, the most commonly used linear space is the
RKHS. However, in this transformation, an SPD matrix is transformed into a function defined on SPD manifold, and the matrix form and properties of the SPD matrix are transformed. Using such data for machine learning may no longer be the original intention. Looking back at log transformation, log transformation just performs log transformation on the eigenvalues in the SPD matrix feature decomposition, and log function is a monotonically increasing function which maintains the order of the eigenvalues in the SPD matrix feature decomposition. Therefore, log transformation is the transformation that changes the SPD matrix the least in terms of form and nature.

B. THE FRAMEWORK OF BI-SUBSPACE LEARNING OF SPD MANIFOLD UNIT MATRIX TANGENT SPACE

Non-European data (not data in linear space) is becoming more and more common in machine learning, but now machine learning algorithms involve a large number of linear operations, so non-European data cannot be processed directly. The current common approach is to map non-European data to Hilbert space, which means transfer DAL to Hilbert space to learn. We can transfer machine learning from SPD manifold to SMS under certain conditions. This paper studies the domain adaptation of SPD data on SMS, so the given learning sample are SPD data samples. From the perspective of SMS, these SPD data lie in one corner, and we have expanded the domain adaptation of SPD data to the domain adaptation of symmetric matrix data. Therefore, in this paper, we use log transformation mentioned in the previous section to transform the data into Riemannian manifold tangent space, not to the RKHS, Riemannian manifold tangent space is finite-dimensional Hilbert space, which is also Euclidean space, while the RKHS is infinite-dimensional Hilbert space, not Euclidean space. At present, most machine learning algorithms are developed in European space and can be directly applied to Riemannian manifold tangent space. Here, we first give the framework of bi-subspace learning of SPD manifold unit matrix tangent space.

1) THE CONSTRUCTION AND CONSTRAINT OF THE BI-SUBSPACE OF SPD MANIFOLD UNIT MATRIX TANGENT SPACE

Given the two sets of learning samples $\{X_1, \ldots, X_N\} \subseteq \text{Sym}^m_{++}$, $\{X_{N_1}, \ldots, X_{N_N}\} \subseteq \text{Sym}^m_{++}$ on SPD manifold, after log transformation, they are transformed into the tangent space $T_{Id} \left( \text{Sym}^m_{++} \right)$ of the identity matrix $I_d$:

$$\{ \log (X_1), \ldots, \log (X_N) \} \subseteq T_{Id} \left( \text{Sym}^m_{++} \right)$$ (11)

$$\{ \log (X_{N_1}), \ldots, \log (X_{N_N}) \} \subseteq T_{Id} \left( \text{Sym}^m_{++} \right)$$ (12)

We take the linear combination of $\{ \log (X_1), \ldots, \log (X_N) \}$ to construct the orthonormal basis $\{ \theta_1, \ldots, \theta_d \}$ of a subspace in the bi-subspace as an example. The construction of the orthonormal basis $\{ \theta_1, \ldots, \theta_d \}$ of the other subspace is the same:

$$\theta_{si} = \sum_{j=1}^{N_s} w_{s(j)i} \log \left( X_j \right) \in T_{Id} \left( \text{Sym}^m_{++} \right), \quad i = 1, \ldots, d$$ (13)

That is to say:

$$\begin{bmatrix} \langle \theta_{s1}, \theta_{s1} \rangle_F & \cdots & \langle \theta_{s1}, \theta_{sd} \rangle_F \\ \vdots & \ddots & \vdots \\ \langle \theta_{sd}, \theta_{s1} \rangle_F & \cdots & \langle \theta_{sd}, \theta_{sd} \rangle_F \end{bmatrix} = I_d$$ (14)

Because

$$\begin{bmatrix} \langle \theta_{si}, \theta_{sj} \rangle_F \\ \vdots \\ \langle \theta_{sd}, \theta_{sd} \rangle_F \end{bmatrix} = \begin{bmatrix} \sum_{p=1}^{N_s} w_{s(pi)} \log(X_p) - \sum_{q=1}^{N_s} w_{s(qj)} \log(X_q) \end{bmatrix} \in R^{d \times d}$$ (15)

Here

$$P_s = \begin{bmatrix} \langle \log(X_1), \log(X_1) \rangle_F & \cdots & \langle \log(X_1), \log(X_{N_1}) \rangle_F \\ \vdots & \ddots & \vdots \\ \langle \log(X_{N_1}), \log(X_1) \rangle_F & \cdots & \langle \log(X_{N_1}), \log(X_{N_1}) \rangle_F \end{bmatrix} \in R^{N_1 \times N_1}$$ (16)

$$W_s = \begin{bmatrix} w_{s(11)} & \cdots & w_{s(1d)} \\ \vdots & \ddots & \vdots \\ w_{s(N_1,1)} & \cdots & w_{s(N_1,d)} \end{bmatrix} = \begin{bmatrix} w_{s(1Col)} & \cdots & w_{s(dCol)} \end{bmatrix} \in R^{N_1 \times d}$$ (17)

where $W_{s(dCol)}$ is the $d^{th}$ row vector of $W_s$, $i = 1, \ldots, d$. So we have

$$\begin{bmatrix} \langle \theta_{s1}, \theta_{s1} \rangle_F & \cdots & \langle \theta_{s1}, \theta_{sd} \rangle_F \\ \vdots & \ddots & \vdots \\ \langle \theta_{sd}, \theta_{s1} \rangle_F & \cdots & \langle \theta_{sd}, \theta_{sd} \rangle_F \end{bmatrix} = \begin{bmatrix} W_{s(dCol)}^T P_s W_{s(dCol)} & \cdots & W_{s(dCol)}^T P_s W_{s(dCol)} \\ \vdots & \ddots & \vdots \\ W_{s(dCol)}^T P_s W_{s(dCol)} & \cdots & W_{s(dCol)}^T P_s W_{s(dCol)} \end{bmatrix}$$ (18)

Obviously, after the learning sample $\{X_1, \ldots, X_N\}, \{X_{N_1}, \ldots, X_{N_N}\}$ is given, the bi-subspace is determined by the coefficient matrices $W_s$ and $W_d$. Therefore, in this paper, we use $W_s$ and $W_d$ to represent the bi-subspace. Both the bi-subspaces $W_s$ and $W_d$ are $d$-dimensional subspaces which must satisfy the above constraints.

Remark 2: Suppose $\log(X_i) = [\log(X_{i1}), \ldots, \log(X_{iN})]$, $\log(X_j) = [\log(X_{j1}), \ldots, \log(X_{jN})]$, here $\log(X_i)$ and $\log(X_j)$ are both expressed in the form of a column vector, $i = 1, \ldots, N_s, j = 1, \ldots, N_t$, then

$$P_s = \begin{bmatrix} \langle \log(X_1), \log(X_1) \rangle_F & \cdots & \langle \log(X_1), \log(X_{N_1}) \rangle_F \\ \vdots & \ddots & \vdots \\ \langle \log(X_{N_1}), \log(X_1) \rangle_F & \cdots & \langle \log(X_{N_1}), \log(X_{N_1}) \rangle_F \end{bmatrix} \in R^{N_1 \times N_1}$$ (19)
According to the projection theorem, if \( X_t \) is shown in Fig.1.

\[
P_t = \begin{bmatrix} \langle \log(X_t), \log(X_t) \rangle_F & \cdots & \langle \log(X_t), \log(X_{N_t}) \rangle_F \\ \vdots & \ddots & \vdots \\ \langle \log(X_{N_t}), \log(X_t) \rangle_F & \cdots & \langle \log(X_{N_t}), \log(X_{N_t}) \rangle_F \end{bmatrix} = \log^T(X_t)log(X_t) \in R^{N_t \times N_t} \tag{20}
\]

This means that the matrices \( P_s \) and \( P_t \) are at least a symmetric positive semi-definite matrix. For most learning samples, the matrices \( P_s \) and \( P_t \) are a symmetric positive definite matrix. A brief schematic diagram of the construction of the bi-subspace of SPD manifold unit matrix tangent space is shown in Fig.1.

2) THE EXPRESSION OF SPD DATA IN THE BI-SUBSPACE OF SPD MANIFOLD UNIT MATRIX TANGENT SPACE

For all \( X_t \in \text{Sym}^D_{++}, X_i \in \text{Sym}^D_{++}, \), \( \log(X_t) \in T_{I_d}(\text{Sym}^D_{++}), \)

\( \log(X_i) \in T_{I_d}(\text{Sym}^D_{++}), \)

then project \( \log(X_t) \) and \( \log(X_i) \) into bi-subspace \( W_s \) and \( W_t \) of \( T_{I_d}(\text{Sym}^D_{++}) \) respectively. According to the projection theorem, if \( \{\theta_1, \ldots, \theta_d\} \) are the orthonormal basis of bi-subspace \( W_s \) and \( W_t \) respectively, then the coordinates of the projections of \( \log(X_t) \) and \( \log(X_i) \) on bi-subspace \( W_s \) and \( W_t \) are:

\[
y_{si} = \begin{bmatrix} \langle \log(X_t), \theta_1 \rangle_F \\ \vdots \\ \langle \log(X_t), \theta_d \rangle_F \end{bmatrix} = \begin{bmatrix} \langle \log(X_t), \sum_{j=1}^{N_t} w_{ij1} \log(X_j) \rangle_F \\ \vdots \\ \langle \log(X_t), \sum_{j=1}^{N_t} w_{ijd} \log(X_j) \rangle_F \end{bmatrix} = W_s^T \begin{bmatrix} \langle \log(X_t), \log(X_{1t}) \rangle_F \\ \vdots \\ \langle \log(X_t), \log(X_{N_t}) \rangle_F \end{bmatrix} \in R^d \tag{21}
\]

\[
y_{ij} = \begin{bmatrix} \langle \log(X_t), \theta_1 \rangle_F \\ \vdots \\ \langle \log(X_t), \theta_d \rangle_F \end{bmatrix} = \begin{bmatrix} \langle \log(X_t), \sum_{j=1}^{N_t} w_{ij1} \log(X_j) \rangle_F \\ \vdots \\ \langle \log(X_t), \sum_{j=1}^{N_t} w_{ijd} \log(X_j) \rangle_F \end{bmatrix} = W_t^T \begin{bmatrix} \langle \log(X_t), \log(X_{1t}) \rangle_F \\ \vdots \\ \langle \log(X_t), \log(X_{N_t}) \rangle_F \end{bmatrix} \in R^d \tag{22}
\]

In particular, for a given learning sample \( \{X_1, \ldots, X_{N_t}\} \) and \( \{X_1, \ldots, X_{N_s}\} \), their corresponding expressions in the bi-subspace \( W_s \) and \( W_t \) are respectively:

\[
y_{si} = \begin{bmatrix} \langle \log(X_t), \theta_1 \rangle_F \\ \vdots \\ \langle \log(X_t), \theta_d \rangle_F \end{bmatrix} = \begin{bmatrix} \langle \log(X_t), \sum_{j=1}^{N_t} w_{ij1} \log(X_j) \rangle_F \\ \vdots \\ \langle \log(X_t), \sum_{j=1}^{N_t} w_{ijd} \log(X_j) \rangle_F \end{bmatrix} = W_s^T P_{s(iCol)} \in R^d \tag{23}
\]

\[
y_{ij} = \begin{bmatrix} \langle \log(X_t), \theta_1 \rangle_F \\ \vdots \\ \langle \log(X_t), \theta_d \rangle_F \end{bmatrix} = \begin{bmatrix} \langle \log(X_t), \sum_{j=1}^{N_t} w_{ij1} \log(X_j) \rangle_F \\ \vdots \\ \langle \log(X_t), \sum_{j=1}^{N_t} w_{ijd} \log(X_j) \rangle_F \end{bmatrix} = W_t^T P_{t(jCol)} \in R^d \tag{24}
\]

where \( P_{s(iCol)} \) is the \( i \)-th column vector of \( P_s, i = 1, \ldots, d; \)

\( P_{t(jCol)} \) is the \( j \)-th column vector of \( P_t, j = 1, \ldots, d. \)

In this way, we finally realize the embedding from the original SPD manifold to the target Hilbert space (the inner product space of finite dimension is a Hilbert space) bi-subspace, this target space is isomorphic with the Euclidean space, we can perform linear operations on sample points, Euclidean metrics between sample and sample, etc., in this embedding space. Although this domain adaptation framework is clear,
how to determine the coefficient matrix is a problem worthy of discussion, because according to different constraint criterion, we can get different domain adaptation algorithms, and of course, different criterion consider different issues. For example, it can be supervised or unsupervised learning.

### C. Domain Adaption Based on Symmetric Matrices Space Bi-subspace Learning (SMS-BL-DA)

Machine learning based on subspace learning is widely used. This paper applies SPD manifold tangent space bi-subspace learning to domain adaptation. Assume

\[ X_s = \{X_{s1}, \ldots, X_{sN_s}\} \subseteq \text{Sym}_+^D, \]
\[ X_t = \{X_{t1}, \ldots, X_{tN_t}\} \subseteq \text{Sym}_+^D \]  

(25)

where \(X_s\) is the labeled source domain dataset, and \(X_t\) is the unlabeled target domain dataset. The source domain data is labeled data, and the target domain data is unlabeled data. The source domain dataset and the target domain dataset have different distributions. We hope to use the label of the source domain data to identify the label of the target domain data.

Domain adaptation based on bi-subspace learning is to transform the source domain dataset \(X_s\) and the target domain dataset \(X_t\) into the bi-subspace respectively, so that their distribution in the bi-subspace is as the same as possible, and then discriminate the label of the target domain data in the bi-subspace.

We transform the datasets \(X_s\) and \(X_t\) into the bi-subspaces \(W_s\) and \(W_t\) of SPD manifold unit matrix tangent space \(T_{I_0}(\text{Sym}_+^D)\) respectively. According to the framework deduced in part B of section IV, the data form of the source domain dataset \(X_s\) and the target domain dataset \(X_t\) transformed into the bi-subspace and are:

\[
y_{si} = \begin{bmatrix} \langle \log (X_{si}) \cdot \theta_{si} \rangle_F \\ \vdots \\ \langle \log (X_{si}) \cdot \theta_{si} \rangle_F \end{bmatrix} = W_s^T \begin{bmatrix} \langle \log (X_{si}) \cdot \log (X_{si}) \rangle_F \\ \vdots \\ \langle \log (X_{si}) \cdot \log (X_{si}) \rangle_F \end{bmatrix} = W_s^T P_{si(Col)}, i = 1, \ldots, N_s \tag{26} \]

\[
y_{tj} = \begin{bmatrix} \langle \log (X_{tj}) \cdot \theta_{tj} \rangle_F \\ \vdots \\ \langle \log (X_{tj}) \cdot \theta_{tj} \rangle_F \end{bmatrix} = W_t^T \begin{bmatrix} \langle \log (X_{tj}) \cdot \log (X_{tj}) \rangle_F \\ \vdots \\ \langle \log (X_{tj}) \cdot \log (X_{tj}) \rangle_F \end{bmatrix} = W_t^T P_{tj(Col)}, j = 1, \ldots, N_t \tag{27} \]

In part B of section IV, we mentioned that the constraint criterion of DAL are open, and different algorithms can be obtained by using different criterion. In this paper, MMD is used as the criterion for the selection of domain adaption bi-subspace \(W_s\) and \(W_t\):

\[
\begin{align*}
\frac{1}{N_s} \sum_{i=1}^{N_s} y_{si} - \frac{1}{N_t} \sum_{j=1}^{N_t} y_{tj} \\
= \frac{1}{N_s} \sum_{i=1}^{N_s} W_i^T P_{si(Col)} - \frac{1}{N_t} \sum_{j=1}^{N_t} W_t^T P_{tj(Col)} \\
= \text{tr} \left( W^T P M P W^T \right) \left( \frac{1}{N_s} \sum_{i=1}^{N_s} y_{si} - \frac{1}{N_t} \sum_{j=1}^{N_t} y_{tj} \right) \\
= \min \left( N_s \right) \min \left( N_t \right) \end{align*} \tag{28} \]

where

\[
\Gamma_s = \left[ \begin{array}{c} 1/N_s \\ \vdots \\ 1/N_s \\ 0 \\ \vdots \\ 0 \end{array} \right] \in \mathbb{R} ^{N_s+N_t} \quad \Gamma_t = \left[ \begin{array}{c} 1/N_t \\ \vdots \\ 1/N_t \\ 0 \\ \vdots \\ 0 \end{array} \right] \in \mathbb{R} ^{N_s+N_t} \tag{29} \]

\[
\Gamma_N = \Gamma_s - \Gamma_t \quad M = I_N - \frac{1}{N} \Gamma_N \Gamma_N^T \quad N = N_s + N_t \tag{30} \]

\[
W = [W_s; W_t] \quad P = \left( \begin{array}{cc} P_s & 0_{N_s \times N_t} \\ 0_{N_t \times N_s} & P_t \end{array} \right) \tag{31} \]

The constraint \(W^T PW = I_d\) is to avoid trivial solutions. The physical meaning of this model is very clear, that is, by mapping SPD data to SMS so that it can perform linear operations and inner product operations, and then project the data from SMS to SMS subspace to achieve the domain adaptation effect. This is an unsupervised method, which does not require label information and saves the cost of manual labeling.

### D. Linear Discriminant Analysis Regularization of Source Domain Data Subspace Learning

The algorithm we proposed uses bi-subspace learning for DAL, which respectively correspond to the source domain data subspace learning and the target domain data subspace learning. Since the source domain data \(X_s\) has labels, the target domain data \(X_t\) has no labels, and the purpose of unsupervised DAL is to use the labeled source domain data to identify the unlabeled target domain data, so here we choose to add LDA regularization term to the source domain data subspace learning, which also reflects the benefits of bi-subspace learning we propose. According to the previous section, the data form of the source domain data \(X_s\) projected into the subspace \(W_s\) is:

\[
\{\gamma_1^s, \ldots, \gamma_{N_s}^s\} = \{\gamma_{11}^s, \ldots, \gamma_{1N_s}^s, \gamma_{C1}^s, \ldots, \gamma_{CNC}^s\} \]

where $N_S = \sum_{c=1}^{C} N_c$, the mean value of the $c^{th}$ category $(\bar{y}_c, \ldots, \bar{y}_{N_N})$ is:

$$
\bar{y}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} y_{ci} = \frac{1}{N_c} \sum_{c=1}^{N_c} W_s^T P_{c(i)Col} \\
= W_s^T \left( \frac{1}{N_c} \sum_{c=1}^{N_c} P_{c(i)Col} \right) \\
= W_s^T \bar{P}_{Col}
$$

(32)

The mean of all data is:

$$
\bar{y} = \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} y_{ci} = \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} W_s^T P_{c(i)Col} \\
= W_s^T \left( \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} P_{c(i)Col} \right) \\
= W_s^T \bar{P}_{Col}
$$

(33)

where

$$
\bar{P}_{Col} = \frac{1}{N_c} \sum_{c=1}^{N_c} P_{c(i)Col}
$$

(34)

$$
\bar{P}_{Col} = \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} W_s^T P_{c(i)Col}
$$

(35)

Let the inter-class divergence be:

$$
S_b = \frac{1}{N_S} \sum_{c=1}^{C} \frac{N_c}{N_S} \| \bar{y}_c - \bar{y} \|^2 \\
= \frac{1}{N_S} \sum_{c=1}^{C} \frac{N_c}{N_S} (\bar{y}_c - \bar{y})^T (\bar{y}_c - \bar{y}) \\
= tr \left( W_s^T \left( \sum_{c=1}^{C} \frac{N_c}{N_S} (\bar{P}_{Col} - \bar{P}_{Col}) (\bar{P}_{Col} - \bar{P}_{Col})^T \right) W_s \right) \\
= tr \left( W_s^T \bar{P}_{Col} \bar{P}_{Col} W_s \right)
$$

(36)

where

$$
\bar{P}_{Col} = \frac{1}{N_S} \sum_{c=1}^{C} \frac{N_c}{N_S} (\bar{P}_{Col} - \bar{P}_{Col}) (\bar{P}_{Col} - \bar{P}_{Col})^T
$$

(37)

Let the within-class divergence be:

$$
S_w = \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} \| y_{ci} - \bar{y}_c \|^2 \\
= \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} (y_{ci} - \bar{y}_c)^T (y_{ci} - \bar{y}_c) \\
= tr \left( W_s^T \left( \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} (P_{c(i)Col} - \bar{P}_{Col}) (P_{c(i)Col} - \bar{P}_{Col})^T \right) W_s \right) \\
= tr \left( W_s^T \Phi W_s \right)
$$

(38)

where

$$
\Phi = \frac{1}{N_S} \sum_{c=1}^{C} \sum_{i=1}^{N_c} (P_{c(i)Col} - \bar{P}_{Col}) (P_{c(i)Col} - \bar{P}_{Col})^T
$$

(39)

Therefore, in our algorithm, the model for LDA regularization term to source domain data subspace learning is as follows:

$$
tr \left( W_s^T (\Phi - \Psi) W_s \right) = tr \left( W_s^T (\Phi - \Psi) W_s \right) \\
= \frac{w^T P M P^T W}{w^T P W = I_d} \min
$$

(40)

where $D$ is a diagonal matrix, if $x_i \in X_s$, $D_{ii} = 1$, otherwise $D_{ii} = 0$.

**E. SOLUTION TO SMSBL-DA**

In this way, the overall objective function of SMSBL-DA is as follows:

$$
tr \left( W_s^T (PM^T + \lambda D) (\Phi - \Psi) W \right) \\
= tr \left( W_s^T (PM^T + \lambda D) \min \right)
$$

(41)

Let $L = PM^T + \lambda D$, then the above objective function can be rewritten as follows:

$$
tr \left( W_s^T L W \right) \min
$$

(42)

Because the matrix $P$ is a symmetric positive definite matrix, it can be expressed as

$$
P = U \Sigma U^T = U \Sigma^{\frac{1}{2}} \Sigma^{\frac{1}{2}} U^T
$$

(43)

where $U U^T = I$, $\Sigma^{\frac{1}{2}} = \text{diag} (\sqrt{\sigma_1}, \ldots, \sqrt{\sigma_N})$, $\sigma$ is the eigenvalue of $P$, $\sigma = 1, \ldots, N$, and orthonormal eigenvectors of $P$ constitute the column vector of $U$. Based on the above, the constraints of SMSBL-DA can be rewritten as $W_s^T PW = W_s^T U \Sigma^{\frac{1}{2}} \Sigma^{\frac{1}{2}} U^T W = V^T V = I_d$, where $V = \Sigma^{\frac{1}{2}} U^T W$. The objective function of SMSBL-DA can be written as follows:

$$
tr \left( W_s^T L W \right) \\
= tr \left( W_s^T U \Sigma^{\frac{1}{2}} \Sigma^{\frac{1}{2}} U^T W \right) \\
= tr \left( V^T NV \right)
$$

(44)

where $N = \Sigma^{\frac{1}{2}} U^T L U \Sigma^{\frac{1}{2}}$. Obviously, $N$ is at least a symmetric positive semi-definite matrix. Here we use the properties of generalized Rayleigh entropy to solve the above problem. $V$ is the $d$-dimensional column vector, which is composed of the orthonormal eigenvectors corresponding to the first $d$ smallest eigenvalues of the matrix $N$. After the matrix $V$ is obtained, the matrix $W$ is:

$$
W = U \Sigma^{\frac{1}{2}} V
$$

(45)

The solution process of SMSBL-DA is displayed in Algorithm 1.
Algorithm 1 SMSBL-DA

**Input:** source domain data sample $X_s$, target domain data sample $X_t$, the label information of $X_s$, parameter $\lambda$.

**Output:** the label information of $X_t$.

1. Map source domain data $X_s$ and target domain data $X_t$ to SMS.
2. Calculate matrices $P, M, \Phi, \Psi, D, N$.
3. Perform eigenvalue decomposition on $N$, and take the orthonormal eigenvectors corresponding to the first $d$ smallest eigenvalues of $N$ as the column vector of $V$ to obtain matrix $V$, and get the subspace coefficient matrix $W$ from $W = U \Sigma^{-\frac{1}{2}} V$.
4. Project $X_s$ and $X_t$ to their respective subspaces to obtain data, that is $y^l_i = W^T P_d(iCol)$ and $y^l_j = W^T P_t(jCol)$. Classify $y^l_i$ in the subspace by $1$-KNN, and $y^l_j$ is used as reference.

**F. COMPLEXITY ANALYSIS**

The time complexity of the SMSBL-DA is mainly composed of four main parts: mapping the source domain data $X_s$ and target domain data $X_t$ into SMS, calculating the complexity of matrices $P, M$ and $D$, calculating the complexity of matrices $\Phi$ and $\Psi$, and optimization of feature problems. Here we discuss the time complexity of these main steps. The time complexity of the mapping from the original SPD manifold to SMS is $O(N^3 \log(n))$; the time complexity of calculating inner product matrix $P$ is $O(N^2 n^3)$, the time complexity of calculating matrices $M$ and $D$ is $O(N^4) + O(N^4) = O(2N^4)$, the time complexity of calculating matrices $\Phi$ and $\Psi$ is $O(N^4) + O(N^4) = O(2N^4)$; the time complexity of eigenvalue decomposition is $O(dn^3)$, and the total time complexity is: $O(N^3 \log(n) + N^2 n^3 + 2N^4 + 2N^4 + dN^3)$.

**V. COMPARISON TO OTHER RELATED ADVANCED ALGORITHMS**

**A. COMPARE WITH TCA**

The essence of TCA [46] is to learn the subspace that minimizes the distribution difference between the source domain data and the target domain data and maintains the data characteristics. TCA first maps the original data to the RKHS through the kernel function, and then maps the feature data of the infinite-dimensional kernel space to an $m$-dimensional low-dimensional RKHS subspace by determining a subspace projection matrix $W \in \mathbb{R}^{(N_s+N_t) \times m}$. The model of TCA is:

$$
\min_W \frac{1}{N_s} \sum_{i=1}^{N_s} W^T P_d(iCol) - \frac{1}{N_t} \sum_{j=1}^{N_t} W^T P_t(N_s+j)Col \quad \text{s.t. } W^T PHPW = I_d 
$$

The constraints $W^T KHKW = I_d$ can maintain the variance of the data and the linear independence of the projection matrix, $tr(W^T W)$ can control the complexity of the projection matrix $W \in \mathbb{R}^{(N_s+N_t) \times m}$. Both SMSBL-DA and TCA use MMD, but SMSBL-DA proposes to add LDA regularization term for the source domain data subspace learning, which also reflects the benefits of the bi-subspace learning in SMSBL-DA. In addition, TCA is based on the RKHS subspace, while SMSBL-DA is based on SMS bi-subspace.

**B. COMPARE WITH SSTCA**

SSTCA [46] is a semi-supervised learning method proposed based on TCA. SSTCA chooses to minimize the probability distribution difference between the source domain data and the target domain data in the embedding space. It introduces HSIC to more accurately estimate the dependency between the label and the data and maintains the local data geometric structure by introducing a popular regularization term. The model of SSTCA is:

$$
\min_W \left\| \frac{1}{N_s} \sum_{i=1}^{N_s} W^T P_d(iCol) - \frac{1}{N_t} \sum_{j=1}^{N_t} W^T P_t(N_s+j)Col \right\|^2 + \frac{\lambda}{N_s N_t} \sum_{i=1}^{N_s} \sum_{j=1}^{N_t} W^T P_d(iCol) - W^T P_t(N_s+j)Col \quad \text{s.t. } W^T PHPW = I_d 
$$

SSTCA, like TCA, is based on the RKHS subspace learning, and both adopt the MMD. The main difference between SMSBL-DA and SSTCA is: SSTCA is to perform subspace learning on the RKHS, while SMSBL-DA uses SMS as a working space to perform DAL on SPD manifolds. By mapping SPD data to SMS, it can perform linear operations and inner product operations.

**C. COMPARE WITH IGLDA**

IGLDA [47] proposed a new mapping function to make the distribution difference between the source domain and target domain data as small as possible, and through the kernel function, the source domain data and target domain data are mapped to a latent space. IGL refers to the idea of integrating global and local metrics. IGLDA aimed to retain the local geometric structure of the labeled source domain data, and its purpose is to use the knowledge acquired in the source domain to solve problems in the target domain. The model of IGLDA is:

$$
\min_W \left\| \frac{1}{N_s} \sum_{i=1}^{N_s} W^T P_d(iCol) - \frac{1}{N_t} \sum_{j=1}^{N_t} W^T P_t(N_s+j)Col \right\|^2 + \mu tr(W^T W) + \frac{c}{N_t(N_t-1)} \sum_{i=1}^{N_t} \sum_{j=i+1}^{N_t} \left\| W^T P_d(iCol) - W^T P_t(jCol) \right\|^2 \quad \text{s.t. } W^T PHPW = I_d 
$$
where \(c\) is the number of categories, and \(N_j\) is the number of data in each category. The main difference between our algorithm and IGLDA is that we transfer the data from the SPD manifold to the SMS through \(\log\) transformation. \(\log\) transformation has the smallest change to the form of the data, it only performs a logarithmic transformation on the eigenvalues of the matrix, and hardly changes the form of the matrix. However, IGLDA performs DAL in the RKHS and transforms the data into functions defined in the original space, which leads to relatively big changes in the form and nature of the data.

### D. COMPARE WITH TIT

TIT [48] theoretically studied two strategies, namely distribution matching and knowledge adaptation. TIT minimizes MMD of the infinite-dimensional RKHS, but it is worth noting that the distribution matching part still uses the original ecological version of MMD. The disadvantage of this method is that the probability distribution of the data is not known, and the mean of the maximum mean difference is used instead of the approximate distribution, so MMD obtained is replaced by the mean. In the knowledge adaptation part, manifold regularization term and row sparse regularization term are introduced. The model of TIT is:

\[
\min_{W} \left\{ \frac{1}{N_t} \sum_{i=1}^{N_t} W^T P_iCol - \frac{1}{N_I} \sum_{j=1}^{N_I} W^T P_{(N_t+j)Col} \right\}^2 \\
+ \frac{1}{2} \sum_{j,l=1}^{N} \left( W^T P_jCol - W^T P_lCol \right)^2 P_{jl} \\
+ \beta \left( P_t - W_t W_t^T P_t \right)_F^2 + \mu \|W\|_{2,1} \\
s.t. W^T PHPW = I_d
\]  

(49)

The differences between SMSBL-DA and TIT are as follows: first, TIT is based on the RKHS subspace learning, while SMSBL-DA maps the data from the original SPD manifold to SMS through \(\log\) transformation. Secondly, TIT transfers data to a subspace, and our algorithm projects the data into their respective subspaces. Finally, TIT uses a popular regularization term similar to PCA, while SMSBL-DA uses source domain data subspace learning LDA regularization term.

### E. COMPARE WITH GSL

GSL [49] is also an unsupervised domain adaption algorithm, Guided Learning refers to a learning method motivated by feedback information. GSL does not use MMD criterion, but simply minimizes the Bregman divergence between two subspaces \(W_s\) and \(W_t\) as the basic model. GSL obtains more inherent information of the data by reconstructing the inter-domain data to guide the learning of the target domain subspace \(W_t\). The model of GSL is:

\[
\min_{W_t, W_s, M, Z} \beta \|W_t - W_s\|_F^2 + \|W_t^T X_t - W_s^T X_s Z\|_F^2 + \lambda \|Z\|_\infty + \|W_t^T X - \hat{Y} \circ M\|_F^2 \\
s.t. M \geq 0
\]  

(50)

where \(\|W_t^T X - \hat{Y} \circ M\|_F^2\) is a classifier. The differences between SMSBL-DA and GSL lie in: first, GSL finds two subspaces in the original space, while SMSBL-DA finds two subspaces in SMS. Second, in SMSBL-DA, SMS is the smallest linear extension of SPD manifold, and SPD manifold is a submanifold of SMS. Given the original SPD data, from the perspective of SMS, these SPD data are only in one corner, and we diffuse SPD manifold to the entire space, that is, expand to SMS.

### VI. EXPERIMENTS

In this section, in order to verify the validation and effectiveness of the DAL algorithm SMSBL-DA we proposed in this paper, we perform SMSBL-DA to object recognition tasks and face recognition tasks on 5 real-world datasets. For object recognition tasks, we use COIL [57], and MSRC+VOC2007 [61]. For face recognition task, we use AR dataset [62], ORL dataset [63], and YALE dataset [60]. And SMSBL-DA is compared with five domain adaptation algorithms mentioned in this paper, including TCA [46], SSTCA [46], IGLDA [47], TIT [48], and GSL [49].

### A. DATA PROCESSING AND CLASSIFICATION STRATEGY

Data processing: to convert the original image to the SPD matrices, the area covariance matrices need to be calculated. For a given rectangular area \(R \subset F\), let \(\{f_i\}, i = 1, \ldots, n\) be the \(d\)-dimensional feature vector of \(n\) pixels in the rectangular area. Then for the area \(R\), the \(d \times d\)-dimensional covariance matrix of the area can be calculated:

\[
C_R = \frac{1}{n-1} \sum_{i=1}^{n} (f_i - \mu)(f_i - \mu)^T
\]  

(51)

where \(\mu\) is the mean of \(\{f_i\}, i = 1, \ldots, n\).

Classification strategy: for each classification task, the classification training set is all source domain data, and the test set is all target domain data. In each recognition task, our algorithm and comparison algorithms adopt SPD feature matrices or feature vectors as the data of two domains. But the final feature representations obtained by all these algorithms are all vectors, such as \(y_i^j, i = 1, \ldots, N_s\) and \(y_j^j, j = 1, \ldots, N_t\) for our algorithm, and so on. For the classification results, i.e., the labels, of the 16 original SPD data (or feature vectors) generated by a single image of the target domain, the one label which is accounted for the largest proportion among the 16 results will be taken as the final classification result of the image.
B. COIL DATASET OBJECT RECOGNITION TASK

EXPERIMENTAL RESULTS AND ANALYSIS

COIL dataset [57] contains 1440 grayscale images of 20 objects with different angles. The camera takes a picture of each object every 5 degrees, and each object has 72 pictures, and the size of image is uniformly 128 x 128. We divide COIL dataset into 4 subsets named COIL1, COIL2, COIL3, and COIL4, which contains all images taken in the directions of [0,85], [90,175], [180,265], and [270,355] respectively. COIL dataset is divided into regions. For COIL dataset, images are divided into regions. For COIL dataset, we adopted a “rich feature extraction” strategy to extract SPD manifold feature matrices data from images. First, images are divided into regions. For COIL dataset, each domain can be regarded as the source domain or the target domain, so we have 12 different tasks as shown in the first column of Table 1.

In this experiment, based on the regional covariance descriptor [58], we adopted a “rich feature extraction” strategy to extract SPD manifold feature matrices data from images. First, images are divided into regions. For COIL dataset, images are still the original 128 x 128, the region window size is set to 47 x 47, the horizontal and vertical window sliding steps are both 27, and an image will produce 16 overlapping regions. Then, for each pixel (x, y) of the image, we define a 17-dimensional feature vector:

\[ f_{x,y}^{COIL} = [x, I(x,y), I_x, I_y, I_{xx}, I_{yy}, \sqrt{I_x^2 + I_y^2}, \arctan\left(\frac{I_y}{I_x}\right), G_1(x,y), \ldots, G_8(x,y)] \]  

(52)

where \( I(x,y) \) is the gray value of the pixel (x, y), \( I_x \) and \( I_y \) are the first step of the image in the horizontal and vertical directions respectively, \( I_{xx} \) and \( I_{yy} \) are the second step of the image in the horizontal and vertical directions respectively, and \( G_i(x,y), i = 1, \ldots, 8 \) are influence value of a set of 8 DOOG filters on \( I(x,y) \) [59]. Finally, we calculate the regional covariance matrix for each regional block, obtain a 16-dimensional SPD matrix from the data processed by the five comparison algorithms TCA, SSTCA, IGLDA, TIT, and GSL are all European eigenvector data, in order to compare fairness, we adopt the method in [60] to vectorize the upper (or lower) triangular matrix including diagonal elements in the regional covariance matrix, so the dimension of the feature vectors using by the comparison algorithms on COIL dataset is 17 x (17 + 1) / 2 = 153. The experimental settings and parameter selection of our algorithm and the comparison algorithms are as follows: we choose the linear kernel uniformly for the kernel function of the comparison algorithms, the two subspace dimensions of our algorithm and the subspace dimension of the comparison algorithms are both set to 1000 dimensions, the classifier adopts 1-NN classifier, and repeat each task 10 times.

SMSBL-DA: \( \lambda = 1; \) TCA: \( \mu = 1; \) SSTCA: \( \mu = 1, \lambda = 5 \times 10^3; \) IGLDA: \( \mu = 1, \lambda = 100; \) TIT: \( \mu = 10^{-14}, \lambda = 10^{-10}, \beta = 1; \) GSL: \( \lambda = 0.1, \beta = 4 \times 10^3. \)

The experimental results are shown in Table 1, the bold data in the table mark the best results in each task, and the 12 tasks of COIL dataset are numbered 1-12 in Fig. 4 following the order in the first column of Table 1. According to the results of the table, in the 12 tasks of COIL dataset, the performance of the six algorithms is quite excellent, but SMSBL-DA is still better than the other five comparison algorithms, and the average classification accuracy rate reaches the highest 94.56%. Compared with TCA, SSTCA, IGLDA, TIT, and GSL, SMSBL-DA has 7.80%, 16.87%, 9.00%, 5.24%, and 5.88% improvement in average classification accuracy respectively. Among them, SMSBL-DA has the most significant improvement in classification accuracy for SSTCA, reaching 17.42%, and it is still about 3.17% higher than the best-performing TIT in the comparison algorithms. This shows that SMSBL-DA has good robustness and stability.

C. MSRC+VOC2007 DATASET OBJECT RECOGNITION TASK

EXPERIMENTAL RESULTS AND ANALYSIS

The MSRC+VOC2007 dataset [61] is often used in machine learning and contains various pictures of objects, animals, plants, and so on. MSRC (M) provided by Microsoft Research...
to a 300 VOC2007 dataset, we uniformly resized each image
+ MSRC from images. Firstly, images are divided into regions, for
strategy to extract SPD manifold feature matrices data
descriptor [58], we adopted a “rich feature extraction” 
M. V and V two tasks in this experiment, namely M → V and V → M.

In this experiment, based on the regional covariance 
descriptor [58], we adopted a “rich feature extraction” 
strategy to extract SPD manifold feature matrices data 
from images. Firstly, images are divided into regions, for 
MSRC+VOC2007 dataset, we uniformly resized each image 
to a 300 × 300 image, and the region window size is set 
to 120 × 120, the horizontal and vertical window sliding 
steps are both 60, so an image will produce 16 non-overlapping 
area blocks. Then, for each pixel (x, y) of the image, we define 
a 17-dimensional feature vector:

\[
\begin{align*}
J_{x,y}^{M,V} &= [x, y, I(x, y), I_x, I_y, I_{xx}, I_{yy}, \sqrt{I_x^2 + I_y^2}, \\
& \arctan\left(\frac{|I_x|}{|I_y|}\right), G_1(x, y), \cdots, G_8(x, y)] 
\end{align*}
\]  

(53)

where \(I(x, y)\) is the gray value of the pixel \((x, y)\), \(I_x\) and \(I_y\) 
are the first step of the image in the horizontal and vertical 
directions respectively, \(I_{xx}\) and \(I_{yy}\) are the second step of the 
image in the horizontal and vertical directions respectively, 
and \(G_i(x, y), i = 1, \cdots, 8\) are influence value of a set 
of 8 DOOG filters on \(I(x, y)\) [59]. Finally, we calculate the 
regional covariance matrix for each regional block, obtain a 
16-dimensional SPD matrix from a picture. Since the data 
processed by the five comparison algorithms TCA, SSTCA, 
IGLDA, TIT, and GSL are all European eigenvector data,
in order to compare fairness, we adopt the method in [60] 
to vectorize the upper (or lower) triangular matrix including 
diagonal elements in the regional covariance matrix, so the 
dimension of the feature vectors using by the comparison 
algorithms on MSRC+VOC2007 dataset is \(17 \times (17 + 1) \div 2 = 153\). The experimental settings and parameter selection 
of our algorithm and the comparison algorithms are as follows: 
we choose the linear kernel uniformly for the kernel function 
of the comparison algorithms, the two subspace 
dimensions of our algorithm and the subspace dimension 
of the comparison algorithms are both set to 1000 dimen-
sions, the classifier adopts 1-NN classifier, and repeat each 
task 10 times. SMSBL-DA: \(\lambda = 1\); TCA: \(\mu = 10^{-10}\); 
SSTCA: \(\mu = 10^{-10}, \lambda = 10^{-10}\); IGLDA: \(\mu = 1, \lambda = 100\); 
TIT: \(\mu = 10^{-14}, \lambda = 10^{-10}, \beta = 1\); GSL: \(\lambda = 1, \beta = 10^3\).

The experimental results are shown in Table 2, the bolded 
data in the table mark the best results in each task and 
the 2 tasks of MSRC+VOC2007 dataset are numbered 1-2 
in Fig. 6 following the order in the first column of Table 2. 
Among the two tasks in MSRC+VOC2007 dataset, SMSBL-
DA has the best performance on one task, and TIT has the best 
performance on one task. However, the average classification 
accuracy of SMSBL-DA is still the highest, reaching 59.20%, 
which is 0.30% higher than the best TIT in the compari-
on algorithms. Compared with TCA, SSTCA, IGLDA, TIT 
and GSL, SMSBL-DA has 5.55%, 7.00%, 13.80%, 0.30% 
and 8.40% improvement in average classification accuracy 
respectively. It can be seen that the average classification 
accuracy of SMSBL-DA compared to IGLDA has the largest 
increase, reaching 13.80%. Therefore, SMSBL-DA proposed 
in this paper has certain stability and robustness under 
MSRC+VOC2007 dataset.

![FIGURE 4. The 12 tasks of COIL dataset are numbered 1-12 in Fig. 4 following the order in the first column of Table 1.](image)

![FIGURE 5. Example images of six categories in MSRC and VOC2007.](image)

| Task | SMSBL-DA | TCA | SSTCA | IGLDA | TIT | GSL |
|------|----------|-----|-------|-------|-----|-----|
| M→ V | 48.80    | 48.00 | 47.60 | 44.80 | **54.20** | 47.20 |
| V→ M | **69.60** | 58.80 | 56.80 | 46.00 | 63.60 | **54.40** |
| average | **59.20** | 53.65 | 52.20 | 45.40 | 58.90 | 50.80 |

**TABLE 2. Classification accuracy (%) of each algorithm in MSRC+VOC2007 dataset, the bold is the best result, and the classifier adopts 1-NN classifier.**

**D. AR DATASET FACE RECOGNITION TASK EXPERIMENTAL RESULTS AND ANALYSIS**

AR dataset [62] is a face recognition dataset widely used 
in various algorithms which contains more than 4000 face
photos of 126 people. This experiment selected a most common data subset, this subset contains the grayscale images of 100 people’s faces, 26 pictures per person, a total of 2600 pictures of 165 × 120 size. The 26 pictures of each person differ in expression, lighting, obstructions (sunglasses or scarf), and shooting date (taken in two weeks), so it meets our domain adaption task requirements. Fig.7 shows 26 pictures of one person from AR dataset, 26 pictures are divided into 2 rows and 13 columns, the 13 pictures in the first row are the first group, and the 13 pictures in the second row are the other group, which were taken two weeks later. In our experiment, we merge the data of columns 1-4, columns 5-7, columns 8-10, and columns 11-13 according to the naked face, the naked face that changes according to the light, the face with sunglasses, and the face with a scarf numbered as (1), (2), (3), (4), and then use the data domains (1) and (2) as the source domains, and the other three domains as the target domains, so this experiment has set a total of 6 different tasks.

In this experiment, based on the regional covariance descriptor [58], we adopted a “rich feature extraction” strategy to extract SPD manifold feature matrices data from images. Firstly, images are divided into regions, for AR dataset, we uniformly resized each image to a 120 × 120 image, and the region window size is set to 30 × 30, the horizontal and vertical window sliding steps are both 30, so a picture will produce 16 non-overlapping area blocks. Then, for each pixel \((x, y)\) of the image, we define a 17-dimensional feature vector:

\[
J_{x,y}^{AR} = [x, y, I(x, y), I_x, I_y, I_{xx}, I_{yy}, \sqrt{I_x^2 + I_y^2}, \arctan\left(\frac{I_y}{I_x}\right), G_1(x, y), \cdots, G_8(x, y)]
\]

where \(I(x, y)\) is the gray value of the pixel \((x, y)\), \(I_x\) and \(I_y\) are the first step of the image in the horizontal and vertical directions respectively, \(I_{xx}\) and \(I_{yy}\) are the second step of the image in the horizontal and vertical directions respectively, and \(G_i(x, y), i = 1, \cdots, 8\) are influence value of a set of 8 DOOG filters on \(I(x, y)\) [59]. Finally, we calculate the regional covariance matrix for each regional block, obtain a 16-dimensional SPD matrix from a picture. Since the data processed by the five comparison algorithms TCA, SSTCA, IGLDA, TIT, and GSL are all European eigenvector data, in order to compare fairness, we adopt the method in [60] to vectorize the upper (or lower) triangular matrix including diagonal elements in the regional covariance matrix, so the dimension of the feature vectors using by the comparison algorithms on AR dataset is \(17 \times (17 + 1) / 2 = 153\). The experimental settings and parameter selection of our algorithm and the comparison algorithms are as follows: we choose the linear kernel uniformly for the kernel function of the comparison algorithms, the two subspace dimensions of our algorithm and the subspace dimension of the comparison algorithms are both set to 1000 dimensions, the classifier adopts 1-NN classifier, and repeat each task 10 times. SMSBL-DA: \(\lambda = 1\); TCA: \(\mu = 10^{-25}\); SSTCA: \(\mu = 10^{-15}, \lambda = 10^{-17}\); IGLDA: \(\mu = 10^{-10}, \lambda = 0.15\); TIT: \(\mu = 10^{-12}, \lambda = 10^{-12}, \beta = 1\); GSL: \(\lambda = 0.1, \beta = 4 \times 10^3\).

The experimental results are shown in Table 3, the bolded data in the table mark the best results in each task, and the 6 tasks of AR dataset are numbered 1-6 in Fig. 8 following the order in the first column of Table 3. According to the results of the table, among 6 tasks in AR dataset, SMSBL-DA is better than the other 5 comparison algorithms in 6 tasks, and the average classification accuracy of SMSBL-DA is the highest among all algorithms, reaching 89.58 %. Compared with TCA, SSTCA, IGLDA, TIT and GSL, SMSBL-DA has improved average classification accuracy of 21.44%, 23.70%, 25.9%, 16.67% and 25.58% respectively, which is still about 17.02% higher than the best performing TIT. It can be seen that in terms of average classification accuracy, SMSBL-DA has the largest improvement over IGLDA, reaching 25.93%, which is a significant improvement. It shows that our algorithm has a certain degree of stability and robustness.

E. ORL DATASET FACE RECOGNITION TASK

EXPERIMENTAL RESULTS AND ANALYSIS

ORL dataset [63] is also often used in face classification experiments in the field of face recognition, the face images of forty volunteers make up the dataset. The 40 volunteers have different races, genders, and ages, each volunteer collects
TABLE 3. Classification accuracy (%) of each algorithm in AR dataset, the bold is the best result, and the classifier adopts 1-NN classifier.

| Task          | SMSBL-DA | TCA   | SSTCA | IGLDA | TIT   | GSL   |
|---------------|----------|-------|-------|-------|-------|-------|
| (1) → (2)     | 95.00    | 77.00 | 73.83 | 67.00 | 83.83 | 81.33 |
| (1) → (3)     | 82.33    | 51.83 | 34.83 | 53.83 | 65.50 | 78.00 |
| (1) → (4)     | 87.17    | 53.33 | 66.17 | 62.83 | 67.67 | 49.17 |
| (2) → (1)     | 90.00    | 83.00 | 85.25 | 76.88 | 80.00 | 57.33 |
| (2) → (3)     | 90.67    | 75.33 | 61.17 | 53.67 | 79.83 | 64.00 |
| (2) → (4)     | 92.33    | 68.33 | 74.00 | 67.67 | 60.67 | 54.17 |
| average       | 89.53    | 68.14 | 65.88 | 63.65 | 72.91 | 64.00 |

FIGURE 8. The 6 tasks of AR dataset are numbered 1-6 in Fig. 8 following the order in the first column of Table 3.

TABLE 4. Classification accuracy (%) of each algorithm in ORL dataset, the bold is the best result, and the classifier adopts 1-NN classifier.

| Task          | SMSBL-DA | TCA | SSTCA | IGLDA | TIT | GSL |
|---------------|----------|-----|-------|-------|-----|-----|
| (A) → (c)     | 80.00    | 77.50 | 87.50 | 85.00 | 82.50 | 80.00 |
| (A) → (d)     | 72.50    | 55.00 | 72.50 | 65.00 | 55.00 | 45.00 |
| (A) → (e)     | 67.50    | 47.50 | 57.50 | 60.00 | 52.50 | 55.00 |
| (A) → (f)     | 67.50    | 50.00 | 47.50 | 70.00 | 55.00 | 60.00 |
| (A) → (g)     | 67.50    | 55.00 | 57.50 | 62.50 | 47.50 | 52.50 |
| (A) → (h)     | 72.50    | 57.50 | 70.00 | 70.00 | 40.00 | 47.50 |
| (A) → (i)     | 72.50    | 50.00 | 62.50 | 57.50 | 47.50 | 55.00 |
| average       | 72.50    | 57.50 | 64.69 | 66.25 | 54.38 | 55.94 |

FIGURE 9. Example of ORL dataset.

In this experiment, based on the regional covariance descriptor [58], we adopted a “rich feature extraction” strategy to extract SPD manifold feature matrices data from images. Firstly, images are divided into regions, for ORL dataset, we uniformly resized each image to a 30 × 30 image, and the region window size is set to 8 × 8, the horizontal and vertical window sliding steps are both 8, so an image will produce 16 non-overlapping area blocks. Then, for each pixel \((x, y)\) of the image, we define a 17-dimensional feature vector:

\[
f^\text{ORL}_{x, y} = [x, y, I(x, y), I_x, I_y, I_{xx}, I_{yy}, \sqrt{I_x^2 + I_y^2}, \arctan\left(\frac{|I_x|}{|I_y|}\right), G_1(x, y), \cdots, G_8(x, y)]
\]

where \(I(x, y)\) is the gray value of the pixel \((x, y)\), \(I_x\) and \(I_y\) are the first step of the image in the horizontal and vertical directions respectively, \(I_{xx}\) and \(I_{yy}\) are the second step of the image in the horizontal and vertical directions respectively, and \(G_i(x, y), i = 1, \cdots, 8\) are influence value of a set of 8 DOOG filters on \(I(x, y)\) [59]. Finally, we calculate the regional covariance matrix for each regional block, obtain a 16-dimensional SPD matrix from a picture. Since the data processed by the five comparison algorithms TCA, SSTCA, IGLDA, TIT, and GSL are all European eigenvector data, in order to compare fairness, we adopt the method in [60] to vectorize the upper (or lower) triangular matrix including diagonal elements in the regional covariance matrix, so the dimension of the feature vectors using by the comparison algorithms on ORL dataset is \(17 \times (17 + 1) \div 2 = 153\).

The experimental settings and parameter selection of our algorithm and the comparison algorithms are as follows: we choose the linear kernel uniformly for the kernel function of the comparison algorithms, the two subspace dimensions of our algorithm and the subspace dimension of the comparison algorithms are both set to 400 dimensions, the classifier adopts 1-NN classifier, and repeat each task 10 times. SMSBL-DA: \(\lambda = 1\); TCA: \(\mu = 10^{-26}\); SSTCA: \(\mu = 10^{-25}, \lambda = 10^{-7}\); IGLDA: \(\mu = 10^{-10}, \lambda = 2\); TIT: \(\mu = 10^{-6}, \lambda = 10^{-12}, \beta = 1\); GSL: \(\lambda = 0.1, \beta = 5 \times 10^3\).

The experimental results are shown in Table 4, the bolded data in the table mark the best results in each task, and the 8 tasks of ORL dataset are numbered 1-8 in Fig. 10 following the order in the first column of Table 4. According
to the results of the table, among the 8 tasks in ORL dataset, SMSBL-DA has the best performance in 6 tasks, IGLDA has the best performance in 2 tasks, and SSTCA has the best performance in 1 task. The optimal proportion of SMSBL-DA is still the highest, and the average classification accuracy rate reaches the highest 72.50%. Compared with TCA, SSTCA, IGLDA, TIT and GSL, SMSBL-DA has 15.00%, 7.81%, 6.25%, 18.12% and 16.56% improvement in average classification accuracy, respectively. Specifically, for IGLDA, which performed the best in the comparison algorithms, it also increased by 6.25%. It shows that the classification effect of SMSBL-DA is stable, and it has certain robustness.

F. YALE DATASET FACE RECOGNITION TASK
EXPERIMENTAL RESULTS AND ANALYSIS

YALE dataset [60] was created by Yale University and is often used in face classification experiments in the field of face recognition. YALE dataset collects 165 face images of 15 volunteers, and images were collected for each volunteer in 11 situations where there were differences in facial expressions, postures, movements, and light directions. In the experiment, the pictures of 11 situations are represented by letters (a)-(k), (a) and (b) are combined into a domain named (A). We use the data domain (A) as the source domain, (c)-(k) are respectively regarded as 9 target domains, so a total of 9 recognition tasks have been completed. Fig. 11 shows examples in 10 domains of face samples from YALE dataset.

In this experiment, based on the regional covariance descriptor [58], we adopted a “rich feature extraction” strategy to extract SPD manifold feature matrices data from images. Firstly, images are divided into regions, for ORL dataset, we uniformly resized each image to a $30 \times 30$ image, and the region window size is set to $70 \times 51$, the horizontal and vertical window sliding steps are both 8, so an image will produce 16 non-overlapping area blocks. Then, for each pixel $(x, y)$ of the image, we define a 17-dimensional feature vector:

$$j_{x,y}^{YALE} = [x, y, I(x, y), I_x, I_y, I_{xx}, I_{yy}, \sqrt{I_x^2 + I_y^2}, \arctan(I_x/I_y), G_1(x, y), \cdots, G_8(x, y)]$$  \hspace{1cm} (56)

where $I(x, y)$ is the gray value of the pixel $(x, y)$, $I_x$ and $I_y$ are the first step of the image in the horizontal and vertical directions respectively, $I_{xx}$ and $I_{yy}$ are the second step of the image in the horizontal and vertical directions respectively, and $G_i(x, y), i = 1, \cdots, 8$ are influence value of a set of 8 DOOG filters on $I(x, y)$ [59]. Finally, we calculate the regional covariance matrix for each regional block, obtain a 16-dimensional SPD matrix from a picture. Since the data processed by the five comparison algorithms TCA, SSTCA, IGLDA, TIT, and GSL are all European eigenvector data, in order to compare fairness, we adopt the method in [60] to vectorize the upper (or lower) triangular matrix including diagonal elements in the regional covariance matrix, so the dimension of the feature vectors using by the comparison algorithms on YALE dataset is $17 \times (17 + 1) / 2 = 153$. The experimental settings and parameter selection of our algorithm and the comparison algorithms are as follows: we choose the linear kernel uniformly for the kernel function of the comparison algorithms, the two subspace dimensions of our algorithm and the subspace dimension of the comparison algorithms are both set to 400 dimensions, the classifier adopts 1-NN classifier, and repeat each task 10 times. SMSBL-DA: $\lambda = 1$; TCA: $\mu = 10^{-10}$; SSTCA: $\mu = 10^{-10}, \lambda = 10^{-2}$; IGLDA: $\mu = 5 \times 10^{-11}, \lambda = 10^{-7}$; TIT: $\mu = 10^{-9}, \lambda = 1.1 \times 10^{-7}, \beta = 1$; GSL: $\lambda = 0.1, \beta = 5 \times 10^3$.

The experimental results are shown in Table 5, the bolded data in the table mark the best results in each task, and the 9 tasks of YALE dataset are numbered 1-9 in Fig. 12 following the order in the first column of Table 5. Among the 9 tasks in YALE dataset, SMSBL-DA has the best performance in 7 tasks, IGLDA has the best performance in 4 tasks, and GSL has the best performance in 1 task. The optimal proportion of SMSBL-DA is still the highest, and the average classification accuracy of SMSBL-DA reached 91.85%, the highest among all algorithms. Compared with TCA, SSTCA, IGLDA, TIT, and GSL, SMSBL-DA has improved the average classification accuracy of 21.48%, 30.00%, 15.55%, 22.22%, and 25.18% respectively. It can be seen that on YALE dataset, SMSBL-DA has significantly improved the average classification accuracy compared to SSTCA and GSL. At the
same time, the classification accuracy of SMSBL-DA is relatively stable, between 80.00%-100.00%. It shows that the classification effect of SMSBL-DA is stable and has certain robustness.

### Table 5: Classification accuracy (%) of each algorithm in YALE dataset, the bold is the best result, and the classifier adopts 1-NN classifier.

| Task   | SMSBL-DA | TCA | STSCA | IGGLDA | TTT | GSL |
|--------|----------|-----|-------|--------|-----|-----|
| (A)→(c)| 100.00   | 86.67| 76.67 | 93.33  | 80.00| 73.33|
| (A)→(d)| 80.00    | 60.00| 66.67 | 46.67  | 33.33| 33.33|
| (A)→(e)| 86.67    | 46.67| 40.00 | 40.00  | 46.67| 33.33|
| (A)→(f)| 80.00    | 40.00| 26.67 | 20.00  | 33.33| 40.00|
| (A)→(g)| 100.00   | 86.67| 73.33 | 100.00 | 80.00| 86.67|
| (A)→(h)| 93.33    | 73.33| 63.33 | 100.00 | 86.67| 80.00|
| (A)→(i)| 93.33    | 93.33| 73.33 | 93.33  | 93.33| 93.33|
| (A)→(j)| 100.00   | 73.33| 60.00 | 93.33  | 86.67| 73.33|
| (A)→(k)| 93.33    | 73.33| 73.33 | 100.00 | 86.67| 86.67|
| average | 91.85    | 70.37| 61.85 | 76.30  | 69.63| 66.67|

**FIGURE 12.** The 9 tasks of YALE dataset are numbered 1-9 in Fig. 12 following the order in the first column of Table 5.

### VII. CONCLUSION

Traditional machine learning algorithms on SPD manifold usually adopt the RKHS as the workspace, and most of the advanced DAL algorithms still deal with Euclidean data. Different from all of them, the algorithm proposed in this paper explores a new space called SMS to perform the machine learning algorithm (in this paper, it is DAL) on SPD manifold. And the proposed algorithm may be the first attempt to use SMS as the workspace for DAL on SPD manifold. SMS is an inner product space, so it supports linear operations and distance measurements which are needed in machine learning algorithms. After extracting the principal components of each SPD matrix, then the SPD manifold can be transformed into SMS by the Symmetric Mapping, which is defined by the product of the principal components and their transpositions, and SMS is the least linear expansion from SPD manifold. $\log$ transformation is just to take the logarithm of the eigenvalues in the eigenvalue diagonal matrix, and $\log$ transformation of SPD data has the smallest change to the form and nature of the data. In this paper, we concentrate on using $\log$ transformation of data from SPD manifold to SMS to deal with the DAL algorithm based on bi-subspace learning, which uses the mean discrepancy minimized criterion to match the distributions of two domains. And we choose to add LDA regularization term to the subspace learning of the source domain data, which also reflects the benefits of the bi-subspace learning in our algorithm. But obviously, $\log$ transformation is also suitable for other machine learning algorithms on SPD manifold and undoubtedly the proposed method has strong extensibility and applicability.

### REFERENCES

[1] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset,” Calif. Inst. Tech., Pasadena, CA, USA, Tech. Rep., 2007.
[2] C.-X. Ren, P. Ge, P. Yang, and S. Yan, “Learning target-domain-specific classifier for partial domain adaptation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 5, pp. 1989–2001, May 2021, doi: 10.1109/TNNLS.2020.2995648.
[3] S. Chen, M. Harandi, X. Jin, and X. Yang, “Semi-supervised domain adaptation via asymmetric joint distribution matching,” IEEE Trans. Neural Netw. Learn. Syst., early access, Oct. 15, 2020, doi: 10.1109/TNNLS.2020.3027364.
[4] L. Li and Z. Zhang, “Semi-supervised domain adaptation by covariance matching,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 11, pp. 2724–2739, Nov. 2019, doi: 10.1109/TPAMI.2018.2866846.
[5] W. Zhang, C. Li, and S. Teng, “Joint discriminative distribution adaptation and manifold regularization for unsupervised domain adaptation,” in Proc. IEEE 24th Int. Conf. Comput. Supported Cooperat. Workshop Design (CSCWD), May 2021, pp. 226–231, doi: 10.1109/CSCWD49262.2021.9437655.
[6] Y. Li, L. Cheng, Y. Peng, Z. Wen, and S. Yang, “Manifold alignment and distribution adaptation for unsupervised domain adaptation,” in Proc. IEEE Int. Conf. Multimedia, Jul. 2019, pp. 688–693, doi: 10.1109/ICME.2019.00124.
[7] F. Liu, G. Zhang, and J. Lu, “Heterogeneous domain adaptation: An unsupervised approach,” IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 12, pp. 5588–5602, Dec. 2020, doi: 10.1109/TNNLS.2020.2972293.
[8] C. Gui and J. Hu, “Unsupervised domain adaptation by regularizing softmax activation,” in Proc. 24th Int. Conf. Pattern Recognit. (ICPR), Aug. 2018, pp. 397–402, doi: 10.1109/ICPR.2018.8545224.
[9] R. Gui, X. Xu, R. Yang, L. Wang, and F. Pu, “Statistical scattering-component-based subspace alignment for unsupervised cross-domain PolSAR image classification,” IEEE Trans. Geosci. Remote Sens., vol. 59, no. 7, pp. 5449–5463, Jul. 2021, doi: 10.1109/TGRS.2020.3028906.
[10] C. X. Ren, X. L. Xu, and H. Yan, “Generalized conditional domain adaptation: A causal perspective with low-rank translators,” IEEE Trans. Cybern., vol. 50, no. 2, pp. 821–834, Feb. 2020, doi: 10.1109/TCYB.2018.2874219.
[11] R. Cai, J. Li, Z. Zhang, X. Yang, and Z. Hao, “DACH: Domain adaptation without domain information,” IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 2, pp. 5055–5067, Dec. 2020, doi: 10.1109/TNNLS.2019.2906217.
[12] S. Chen, L. Han, X. Liu, Z. He, and X. Yang, “Subspace distribution adaptation frameworks for domain adaptation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 12, pp. 5204–5218, Dec. 2020, doi: 10.1109/TNNLS.2020.2964790.
[13] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, “Kernel methods on the Riemannian manifold of symmetric positive definite matrices,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2013, pp. 73–80, doi: 10.1109/CVPR.2013.17.
[14] M. T. Harandi, R. Hartley, B. Lovell, and C. Sanderson, “Sparse coding on symmetric positive definite manifolds using Bregman divergences,” IEEE Trans. Neural Netw. Learn. Syst., vol. 27, no. 6, pp. 1294–1306, Jun. 2016.
[15] J. Lu, G. Wang, and P. Moulin, “Image set classification using holistic multiple order statistics features and localized multi-kernel metric learning,” in Proc. IEEE Int. Conf. Comput. Vis., Dec. 2013, pp. 329–336, doi: 10.1109/ICCV.2013.48.
J. Zhang, L. Wang, L. Zhou, and W. Li, “Learning discriminative S. Lee and H.-S. Oh, “Robust multivariate regression on Riemannian X. Xie, Z. L. Yu, and Z. Gu, “Classification of symmetric positive definite Z. Huang, R. Wang, S. Shan, and X. Chen, “Projection metric learning X. Pennec, P. Fillard, and N. Ayache, “A Riemannian framework for tensor Z. Cui, W. Li, D. Xu, S. Shan, X. Chen, and X. Li, “Flowing R. Vemulapalli, J. K. Pillai, and R. Chellappa, “Kernel learning Q. Li T. Yao, Y. Pan, C.-W. Ngo, H. Li, and T. Mei, “Semi-supervised domain
et al. J. Hu, B. Ma, Q. Ma, and W. Liang, “Covariance based local salient descriptors for visual tracking,” in Proc. IEEE Int. Conf. Multimedia, J. Hu, L. Wang, L. Zhou, R. Zhang, and W. Li, “Covariance based learning discriminative block sparse representation,” IEEE Access, vol. 8, pp. 19058–19073, 2020, doi: 10.1108/ACCESS.2020.2968817.

[37] Y. Xu, X. Fang, J. Wu, X. Li, and D. Zhang, “Discriminative transfer subspace learning via low-rank and sparse representation,” IEEE Trans. Image Process., vol. 25, no. 2, pp. 850–863, Feb. 2016.

[38] K. Yan, L. Kou, and D. Zhang, “Learning domain-invariant subspace using domain features and independence maximization,” IEEE Trans. Cybern., vol. 48, no. 1, pp. 288–299, Jan. 2018, doi: 10.1109/TCYB.2016.2633306.

[39] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, “Transfer feature learning with joint distribution adaptation,” in Proc. IEEE ICCV, Dec. 2013, pp. 2200–2207.

[40] J. Wang, Y. Chen, S. Hao, W. Feng, and Z. Shen, “Balanced distribution adaptation for transfer learning,” in Proc. IEEE Int. Conf. Data Mining (ICDM), Nov. 2017, pp. 1129–1134.

[41] Z. Deng, M. Shao, and Y. Fu, “Latent low-rank transfer subspace learning for missing modality recognition,” in Proc. 28th AAAI Conf. Artif. Intell., San Francisco, CA, USA, 2014, pp. 1192–1198.

[42] R. Aujendi, R. Emonet, D. Muselet, and M. Sebban, “Landmarks-based kernelized subspace alignment for unsupervised domain adaptation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR). Jun. 2015, pp. 56–63.

[43] Y.-H.-H. Tsai, Y.-R. Yeh, and Y.-C.-F. Wang, “Learning cross-domain landmarks for heterogeneous domain adaptation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR). Jun. 2016, pp. 5081–5090.

[44] M. Bakhtashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann, “Unsupervised domain adaptation by domain invariant projection,” in Proc. IEEE Int. Conf. Comput. Vis., Dec. 2013, pp. 769–776.

[45] S. J. Pan, J. T. Kwock, and Q. Yang, “Transfer learning via dimensionality reduction,” in Proc. Int. Conf. Artif. Intell., 2008, pp. 677–682.

[46] S. J. Pan, I. W. Tsang, J. T. Kwock, and Q. Yang, “Domain adaptation via transfer component analysis,” IEEE Trans. Neural Netw., vol. 22, no. 2, pp. 190–210, Feb. 2011.

[47] M. Jiang, W. Huang, Z. Huang, and G. G. Yen, “Integration of global and local metrics for domain adaptation learning via dimensionality reduction,” IEEE Trans. Cybern., vol. 47, no. 1, pp. 38–51, Dec. 2017, doi: 10.1109/TCYB.2015.2502483.

[48] J. Li, K. Lu, Z. Huang, L. Zhu, and H. T. Shen, “Transfer independently together: A generalized framework for domain adaptation,” IEEE Trans. Cybern., vol. 49, no. 6, pp. 2144–2155, Jun. 2019, doi: 10.1109/TCYB.2018.2820174.

[49] L. Zhang, J. Fu, S. Wang, D. Zhang, Z. Dong, and C. L. P. Chen, “Guide subspace learning for unsupervised domain adaptation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 9, pp. 3374–3388, Sep. 2020, doi: 10.1109/TNNLS.2019.2944455.

[50] R. Gopalan, R. Li, and R. Chellappa, “Domain adaptation for object recognition: An unsupervised approach,” in Proc. Int. Conf. Comput. Vis., Barcelona, Spain, Nov. 2011, pp. 999–1006.

[51] J. Zheng, M.-Y. Liu, R. Chellappa, and P. J. Phillips, “A Grassmann manifold-based domain adaptation approach,” in Proc. 21st Int. Conf. Pattern Recognit. (ICPR), Tsukuba, Japan, 2012, pp. 2095–2099.

[52] B. Fernando, A. Habrard, M. Sebban, and T. Tuylaeels, “Unsupervised visual domain adaptation using subspace alignment,” in Proc. IEEE Int. Conf. Comput. Vis., Sydney, NSW, Australia, Dec. 2013, pp. 2960–2967.

[53] H. T. Ho and R. Gopalan, “Model-driven domain adaptation on product manifolds for unconstrained face recognition,” Int. J. Comput. Vis., vol. 109, nos. 1–2, pp. 110–125, Aug. 2014.

[54] Z. Cui, W. Li, D. Xu, S. Shan, X. Chen, and X. Li, “Flowing on Riemannian manifold: Domain adaptation by shifting covariance,” IEEE Trans. Cybern., vol. 44, no. 12, pp. 2264–2273, Dec. 2014, doi: 10.1109/TCYB.2014.2305701.

[55] X. Xie, Z. L. Yu, H. Lu, Z. Gu, and Y. Li, “Motor imagery classification based on bilinear sub-manifold learning of symmetric positive-definite matrices,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 25, no. 6, pp. 504–516, Jul. 2017, doi: 10.1109/TNSRE.2016.2587939.

[56] F. Tang, M. Fan, and P. Tino, “Generalized learning Riemannian space quantization: A case study on Riemannian manifold of SPD matrices,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 1, pp. 281–292, Jan. 2021, doi: 10.1109/TNNLS.2020.2978514.

[57] C. Rate and C. Retrieval, “Columbia object image library (COIL-20),” Columbia Univ, New York, NY, USA, Tech. Rep. CUCS-006-96, 2011.

[58] J. Jiang and C. Zhai, “Instance weighting for domain adaptation in NLP,” in Proc. Annu. Meeting Assoc. Comput. Linguistics, vol. 7, 2007, pp. 264–271.

[59] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, “Semantic segmentation with second-order pooling,” in Proc. Eur. Conf. Comput. Vis., 2012, pp. 430–443.
[60] M. T. Harandi, C. Sanderson, R. Hartley, and B. Lovell, “Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach,” in Proc. Eur. Conf. Comput. Vis., 2012, pp. 216–229.

[61] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, “Transfer joint matching for unsupervised domain adaptation,” in Proc. IEEE CVPR, Jun. 2014, pp. 1410–1417.

[62] A. M. Martínez and R. Benavente, “The AR face database,” Centre de Vis. Comput., Univ. Autónomo Barcelona, Barcelona, Spain, Tech. Rep. 24, 1998.

[63] The Olivetti & Oracle Research Laboratory Face Database of Faces. Accessed: Jun. 27, 2016. [Online]. Available: http://www.camorl.co.uk/facedatabase.html

QIAN LI received the B.Sc. degree from the School of Optoelectronic Information Science and Engineering, South China University of Technology, Guangzhou, China, in 2019. He is currently pursuing the master’s degree with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou.

ZHENGMING MA received the B.Sc. degree in radio technology and the M.Sc. degree in electronics and communication system from the South China University of Technology, Guangzhou, China, in 1982 and 1985, respectively, and the Ph.D. degree in pattern recognition and intelligent control from the Department of Automation, Tsinghua University, Beijing, China, in 1989. He is currently a Professor with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou. His current research interest includes machine learning.

SHUYU LIU received the Ph.D. degree from Sun Yat-sen University, in 2010. He is currently the Director of the Experimental Teaching Center, School of Pharmacy, Sun Yat-sen University, China. His research interests include biological image processing, biomedical engineering, and intelligent medicine.

YANLI PEI received the B.Sc. and M.Sc. degrees from Zhejiang University, in 2000 and 2003, respectively, and the Ph.D. degree from Hiroshima University, Japan, in 2006. She is currently a Professor with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China.

* * *