Dictionary Learning of Symmetric Positive
Definite Data Based on Riemannian Manifold
Tangent Spaces and Local Homeomorphism

Hui He, Shuyu Liu, Zhengming Ma, and Weijia Feng

1School of Electronics and Information Technology, Sun Yat-Sen University, Guangzhou 510006, China
2Public Experimental Teaching Center, Sun Yat-sen University, Guangzhou 510006, China
3China Telecom Cloud Technology Co., Ltd, Guangzhou 510000, China

Corresponding authors: Shuyu Liu (ljie@mail.sysu.edu.cn); 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 Symmetric Positive Definite (SPD) data are increasingly prevalent in dictionary learning recently. SPD data are the typical non-Euclidean data and cannot constitute a Euclidean space. Therefore, many dictionary learning algorithms cannot be directly adopted on SPD data. Reproducing Kernel Hilbert Spaces (RKHS) is now commonly used to deal with this problem. However, RKHS is an infinite-dimensional Hilbert space, rather than a Euclidean space, resulting in the inability of the dictionary learning to be directly used on SPD data. In this paper, we propose a novel dictionary learning algorithm for SPD data, which is based on the Riemannian Manifold Tangent Space (RMTS). Since RMTS is based on a finite-dimensional Hilbert space, i.e., Euclidean space, most machine learning algorithms developed on Euclidean space can be directly applied to RMTS. The introduction of RMTS provides a better mathematical platform for machine learning of non-Euclidean data. The proposed RMTS method first selects a point on a Riemannian manifold as an anchor point. It transforms the other points on the Riemannian manifold into tangent vectors of the geodesics between these points and the anchor point, with the tangent vectors at the anchor point. We set the length of the tangent vector equal to the length of the geodesic. As a result, such tangent vectors have an explicit geometric meaning, such as direction information, while the RKHS method may cause some geometric meaning to be lost in the original data during the mapping process. In addition, the proposed algorithm adds a regular term of local homomorphism between SPD data and its RMTS dictionary coding coefficients, so that the similarity of SPD data and its RMTS dictionary coding coefficients are as close as possible. Experimental results on four public datasets demonstrate that the proposed algorithm significantly outperforms five state-of-the-art algorithms. This work opens a new pathway towards SPD data dictionary learning methods.

INDEX TERMS Symmetric positive definite matrices, Riemannian manifolds, dictionary learning, sparse coding.

I. INTRODUCTION

In recent years, with the rapid development of machine learning technology, an increasing number of non-Euclidean data are generated. They are performed on the Riemannian manifolds, such as SPD manifolds, Steifel manifolds [1] and Grassmann manifolds [2]. SPD data, a typical type of non-Euclidean data, are generally obtained by extracting regional covariance descriptors from the original data. While they provide a nonlinear representation of the original data and possess rich geometric characteristics, they also offered an outstanding performance in many applications, such as 3D object recognition [3], visual surveillance [4], object recognition [5], action recognition [6], medical imaging [7], manual detection and tracking [8]. The whole SPD matrices form an SPD manifold. Moreover, the SPD manifold is a nonlinear space, instead of a Euclidean space. However, as traditional machine learning algorithms are typically developed on the Euclidean space, they cannot be directly applied to the SPD manifold.
Dictionary learning and sparse representation (DLSR) is a powerful tool to represent data, which becomes increasingly prevalent in the fields of signal processing [9] and computer vision [10]. Given a set of raw data \( x = \{x_1, \ldots, x_n\} \), using the DLSR, it can be considered that they can be effectively represented by a linear combination of a set of dictionaries \( E = \{e_1, \ldots, e_p\} \). The combination coefficient is called sparse representation, i.e., \( A = \{a_1, \ldots, a_n\} \), where the sparseness means that the fewer non-zero vectors of this combination coefficient, the better. Therefore, for a given training dataset, DLSR algorithms aim to learn a set of ideal dictionaries. Then we can use the dictionaries to perform sparse coding for each queried data while minimizing the reconstruction error. Although the research on SPD manifold-based DLSR algorithms has attracted much research attention recently, the progress is still limited. DLSR is essentially a sparse linear approximation that requires a large number of linear operations. However, the SPD manifold is not a linear space. If directly applying traditional DLSR algorithms to it, the nonlinear geometry of the original SPD will be ignored, resulting in poor sparse coding.

The nonlinear problem of the SPD manifold is the key to applying DLSR algorithms to SPD data. To solve this problem, there are mainly two approaches. The first approach is to solve the specific problem of DLSR by building a model directly on the original SPD manifold, i.e., both the original data and the learned dictionaries are SPD matrices. However, the linear combination of a set of SPD matrices cannot guarantee that it is still an SPD matrix. It requires many restrictions, which makes dictionary learning difficult. The second method is to map the original SPD data into the Hilbert space and perform specific machine learning tasks on the Hilbert space to solve the nonlinear problem of the SPD manifold. Currently, the main Hilbert spaces are RKHS and tangent spaces.

Nowadays, Riemannian manifolds are prevalent in machine learning applications. Riemannian manifolds are not Euclidean spaces, and most existing machine learning algorithms are developed in the Euclidean space. Therefore, these machine learning algorithms cannot be directly applied to the Riemannian manifold. Currently, the commonly-used processing method is the RKHS method. The RKHS method first transforms the data on the Riemannian manifold into RKHS by using the reproducing kernels, and then performs machine learning on RKHS. In this paper, we explore applying DLSR on the SPD data, and primary contributions are as follows:

1) We propose a machine learning method for non-Euclidean data based on Riemannian Manifold Tangent Spaces (RMTS). It converts data on Riemannian manifolds into RMTS and then performs machine learning on RMTS. Compared with the RKHS method, RKHS is an infinite dimensional Hilbert space instead of Euclidean space, while RMTS is a finite dimensional Hilbert space, which is isomorphic to Euclidean space. Almost all machine learning algorithms developed based on Euclidean space can be directly applied to RMTS.

2) The proposed RMTS method preserves the geometric relationship between the data on the Riemann manifold. The RKHS method converts the data on the Riemann manifold into a nonlinear function defined on the Riemann manifold, and the form and properties of the data change significantly. However, the RMTS method converts the data on the Riemann manifold into a tangent vector of reference points on the Riemann manifold. The tangent vector starts from the reference point to the other point.

3) We propose to apply the RMTS method to dictionary learning of SPD data. Sparse linear approximation is the most commonly used operation in dictionary learning. However, SPD data do not support linear operations. Therefore, the dictionary learning of SPD data cannot be performed directly on SPD data, or even expressed. Here we apply the RMTS method to dictionary learning of SPD data. We first transform the samples of SPD data dictionary learning into RMTS, and then construct the dictionary learning model on RMTS.

4) An SPD data dictionary learning algorithm based on RMTS and local homeomorphic regularization (RLH-SDL) is proposed. We add a local homeomorphic regular term between SPD data and their dictionary coding coefficients, and ensure that the similarity between SPD data should be similar to the similarity between their dictionary coding coefficients as much as possible.

The remainder of this paper is organized as follows. Section 2 introduces the notations. Section 3 reviews the representative studies of dictionary learning. Section 4 explains the proposed SPD data dictionary learning framework based on SPD Riemannian manifold tangent space and local homeomorphism (RLH-SDL). Section 5 compares the performance of five different algorithms. Section 6 discusses the experimental results on four public datasets. Finally, Section 7 concludes this paper.

II. NOTATIONS AND PRELIMINARIES

A. Notations

We first introduce the notations in this paper, as shown in Table 1.

B. Differential Manifolds

In general, the \( n \)-dimensional topological manifold is a topological space that is locally homeomorphic to the \( n \)-dimensional Euclidean space \( \mathbb{R}^n \). If \( M \) is an \( n \)-dimensional topological manifold, then \( M \) must have a local coordinate system \( \{(U_i, \phi_i)|i \in Q\} \), where \( Q \) is the index set, \( U_i \) is an open set of \( M \), \( \phi_i \) is a homeomorphic mapping between \( U_i \) and an open set of \( n \)-dimensional Euclidean space \( \mathbb{R}^n \), and
Mathematical Notes

It can be proved that ρ'(tₐ) satisfies linearity and Leibniz law, so ρ'(tₐ) is a tangent vector at point ρ(tₐ) on the differential manifold M. Conversely, it can also be proved that for any tangent vector φₚ at point p, there must be a unique differentiable curve ρ, then ρ'(tₐ) = p, ρ'(tₐ) = φₚ.

Let Tₚ(M) denote the totality of the tangent vectors at point p, it can be proved:

\[(φₚ + φₗ)(f) = φₚ(f) + φₗ(f), \quad (αφₚ)(f) = αφₚ(f)\]

Tₚ(M), which is a linear space, is called the tangent space of point p.

It can be further proved that the dimension of Tₚ(M) is the same as that of the differential manifold M. Therefore, Tₚ(M) is a finite-dimensional linear space, and all finite-dimensional linear spaces are isomorphic with the same-dimensional Euclidean space. Therefore, we call the tangent spaces of differential manifolds are all Euclidean spaces.

### C. Riemannian manifolds

There is no distance metric and linear operation between the elements of the differentiable manifolds. Although the tangent space of the differential manifold supports linear operation, there is no distance metric between the tangent vectors. In order to support machine learning algorithms, the differential manifold must be assigned a Riemannian metric, making it a Riemannian manifold.

The so-called Riemannian metric is a symmetric, positive definite, and smooth second-order tensor field on the differential manifold. Since the symmetric positive definite second-order tensor field defines the inner product for the tangent space of each point of the differential manifold, the Riemannian metric can also be regarded as a smooth inner product field on the differential manifold. The Riemannian metric can be used to define the length of the smooth curve between any two points on the Riemannian manifold. Then it can define the distance between any two points, that is, the length of the shortest smooth curve connecting the two points. Such a curve is called a geodesic, and such the distance is called the geodesic distance.

As explained above, the tangent space of each point on the differential manifold is a finite-dimensional linear space. In addition, the Riemannian metric is to define the inner product for the tangent space of each point on the differential manifolds. Therefore, the tangent space of each point on the Riemannian manifold is finite-dimensional inner product space. Furthermore, finite-dimensional inner product spaces are complete. Therefore, 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 Euclidean space endowed with an inner product (all the following Euclidean spaces refer to the Euclidean space endowed with the inner product). Therefore, the tangent space of each point on the Riemannian manifold is essentially the Euclidean space. In addition, the distance...
defined by the Riemannian metric (inner product) is the Euclidean distance.

D. Dictionary Coding and Learning
A dictionary refers to a set of basic data \( \{ E_i, \cdots, E_n \} \), and the function of dictionary coding is to make the data \( X \) approximated by a sparse linear combination of this set of dictionaries. The objective function of dictionary coding is:

\[
X - \sum_{i=1}^{d} \alpha_i E_i + \lambda \text{sparse}(\alpha) \rightarrow \min
\]  

(3)

where \( \text{sparse}(\alpha) \) is the sparse regular term of dictionary coding. It represents that the optimal linear approximation requires the least non-zero component of the approximation coefficient \( \alpha \). We usually use the 1-norm to represent the sparse regular term [9]. The \( \text{sparse}(\alpha) \) can be expressed as follows:

\[
\text{sparse}(\alpha) = \| \alpha \|_1 = \sum_{i=1}^{D} |\alpha_i|
\]  

(4)

It is should be noted that the sparse regular term \( \text{sparse}(\alpha) \) is the feature of dictionary coding. Otherwise, dictionary coding learns the subspace. Therefore, we aim to find a data \( Z \) in the linear subspace \( \text{span}\{E_i, \cdots, E_D\} \), which is formed by the basic data \( \{E_i, \cdots, E_n\} \), and make it the closest to the given data \( X \) as follows:

\[
X - \sum_{i=1}^{D} \alpha_i E_i \rightarrow \min \quad \text{subject to } Z \in \text{span}\{E_i, \cdots, E_D\}
\]  

(5)

The prerequisite of dictionary coding is suitable dictionaries. The dictionary can be learned from the given samples. Based on different applications, we collect a certain number of fully representative samples, and use these samples for dictionary learning [10].

Let \( \{X_1, \cdots, X_n\} \) represent the learning samples, and let \( \{E_1, \cdots, E_n\} \) denote the dictionaries to be learned. Then, the objective function of dictionary learning is as follows:

\[
\sum_{i=1}^{n} \left[ X_i - \sum_{j=1}^{D} \alpha_{ij} E_j \right] + \lambda \text{sparse}(A_{row}) \rightarrow \min
\]  

(6)

Here \( A = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1D} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nD} \end{bmatrix} \) is the dictionary coding matrix. In the above objective function, the dictionary coding matrix \( A \) is a by-product, which is not necessarily required, the dictionary \( \{E_1, \cdots, E_n\} \) is needed.

III. Related Work
With the successful application of SPD data in various fields of machine learning, the related study of SPD data DLSR has attracted much attention. Conventional DLSR algorithms are typically developed on Euclidean space. However, with the continuous development and in-depth research of machine learning, more and more data belong to non-Euclidean data. In recent years, the research of DLSR algorithms based on SPD manifolds has attracted much attention. For the nonlinear problem of SPD manifolds, there are two main solutions: 1) to build the DLSR model directly on the SPD manifold, and 2) to map the original SPD data to Hilbert space.

A. DLSR Modeled Directly on SPD Manifold
DLSR is essentially a sparse linear approximation, which requires a large number of linear operations. However, the SPD manifold is not a linear space. If traditional DLSR algorithms are directly performed on the SPD manifolds, it ignores the nonlinear geometric structure of the original SPD data and also leads to poor sparse coding.

Therefore, appropriate linear decomposition and reconstruction error metrics for SPD matrices should be introduced. In general, if we want to employ the linear combination of atomic matrices to represent SPD matrices, we need to choose the appropriate metrics to measure the reconstruction errors, such as Logdet divergence and Frobenius norm. Sivalingam [13] proposed a tensor sparse coding (TSC) algorithm for positive definite matrices. This algorithm strictly limits each coefficient vector to be nonnegative, and ensures that the sparse coding result is also an SPD matrix. Finally, it uses the defined Logdet divergence to measure the coding errors. In addition, based on TSC, Sivalingam further proposed a new tensor dictionary learning (TDL) algorithm [14]. Like TSC, Sivalingam used the Logdet divergence to measure the coding errors in the dictionary learning process.

Although this approach avoids the extra calculation caused by mapping SPD data into other linear spaces. Modeling on SPD manifolds would strictly limit each coding coefficient to be non-negative in the dictionary learning process. Therefore, it would impose a significant burden on dictionary learning and greatly increase the computational complexity. In addition, the Logdet divergence is not the Riemannian geodesic distance, and it cannot accurately measure the distance between two SPD matrices. Therefore, it is not accurate to use Logdet divergence to measure coding errors.

In addition, there have been many attempts to use other metrics to measure the reconstruction error. Cherian et al. [15] used the method of Riemannian geometry to learn the dictionary. They ensured that the dictionary was composed of SPD matrices, and used affine invariant Riemannian metric (AIRM) to measure the coding errors. AIRM can be a good measure of the distance between two SPD matrices. However, their objective function of dictionary learning is a non-convex function that leads to coding errors. Sra et al. [16] used the Frobenius norm to measure the similarity between SPD matrices in the generalized dictionary learning (GDL) algorithm. They ex-pressed each SPD matrix as a linear combination of the atomic matrices with rank 1. However, it is not a good metric since the Frobenius norm ignores the Riemannian structure of the SPD manifolds.

B. SPD Manifold DLSR Methods Mapped to Hilbert spaces
Recently, when applying DLSR algorithms on SPD data, most of the algorithms map the original SPD data into the
Hilbert space to solve the nonlinear problem of SPD manifolds. At present, the main Hilbert spaces are RKHS and tangent spaces.

It is well known that RKHS is a complete inner product space. We can define a kernel function to generate a unique RKHS. Harandi et al. [9] derive Stein kernel function and Jeffery kernel function. They further mapped the SPD data into RKHS by these two kernel functions and directly performed DLSR on RKHS. Li et al. [17] defined the special matrix logarithmic multiplication and scalar logarithmic multiplication on the original SPD manifolds. Then, they derived three kernel functions for DLSR. Asha et al. [18] applied DLSR of the SPD data to a new field of breast tumor grading. They mapped the SPD data into RKHS by the Gaussian kernel function based on the Log-Euclidean metric, Stein divergence, and Jeffery divergence respectively for DLSR. This is the most sophisticated breast cancer-grading algorithm in terms of both quantitative and qualitative analysis.

However, neither the kernel function defined by Harandi nor the kernel function defined by Li, they do not contain data dependencies. They have no parameters to learn and have little correlation with the original SPD data. Once the kernel function is selected, the mapped RKHS will not change with the change of SPD data. Therefore, the connection between the original data and RKHS is not close, and it cannot achieve the best coding results.

Some studies preserve the original SPD data geometry [19-22]. Li et al. [19] proposed a semantic and neighborhood preservation dictionary learning method. They added a regular term to the objective function based on the Log-Euclidean kernel function, and applied the Graph Laplace smoothing operator to it. Although this method fails to generate a kernel function containing data dependencies, the coding coefficient vector retains the similarity between the original SPD data. Wu et al. [20] used the Belkin kernel function framework defined in [21] and the kernel function defined by Li as the basic kernel function, to propose two new kernel functions containing data dependencies by the Graph Laplace smoothing operator. Zhuang et al. [22] used a Gaussian kernel function as the basic kernel function based on the Log-Euclidean metric. Then they used a symmetric semi-positive definite matrix as a data dependency. However, the kernel function proposed in [21, 22] is too complex and the learning of the data dependency will greatly increase the computational complexity of the algorithm.

Many approaches have been proposed for tangent spaces. Zhang et al. [23] used the Log-Euclidean of SPD matrices to obtain vectorized covariance features for sparse representation. Guo et al. [24] trans-formed the SPD Riemannian manifold into the tangent space by the method of matrix logarithmic mapping. Each mapped SPD matrix can then be represented by a linear combination of other mappings in the tangent space. Yuan [25] et al. mapped the SPD manifold into the tangent space to calculate the sparse representation of the SPD data and applied it to the action recognition. Since the tangent space of the SPD manifold is linear, mapping the SPD matrix data into it can also effectively solve the nonlinearity problem of SPD manifolds.

However, the mapping functions of the above three studies will not only bring additional computation, but also ignore the form of the mapped data. In this way, their tangent spaces only retain the local geometrical structure of the original SPD matrix data, while the characteristics of the SPD matrix data are fundamentally lost. Moreover, they have no further measures in the process of DLSR, resulting in unsatisfactory sparse coding results. The tangent space of the SPD Riemannian manifold proposed in this paper is a symmetric matrix space. Therefore, the data properties of the original SPD data can be retained to the utmost extent while effectively solving the nonlinearity problem of the SPD manifold.

IV. Dictionary Learning of Symmetric Positive Definite Data Based on Riemannian Manifold Tangent Spaces and Local Homeomorphism (RLH-SDL)

SPD data is the most popular non-Euclidean data in machine learning today. Since the whole SPD data cannot constitute a linear space, and dictionary learning is formulated by linear operations. The dictionary learning cannot be directly performed on the SPD data. At present, the common method is to transform SPD data into RKHS, and perform dictionary learning on RKHS. However, the SPD matrix becomes a function defined on the SPD data after SPD data transforming into RKHS. The form and nature of the data have changed significantly.

Although the entire SPD data cannot form a linear space, it can form a Riemannian manifold (hereinafter referred to as the SPD manifold). In addition, the tangent space of the Riemannian manifold is a finite-dimensional Hilbert space. In particular, the tangent space of the SPD manifold is a symmetric matrix space. Since the symmetric matrix includes the symmetric positive definite matrix, the SPD manifold can be regarded as a sub-manifold of the symmetric matrix space. The symmetric matrix space can be also regarded as the smallest linear space containing the SPD manifold. We propose to first transform the SPD data into a symmetric matrix space, and then perform dictionary learning on the symmetric matrix space. Subsequently, the dictionary is also applied on the symmetric matrix space.

A. The Transform from the Riemannian Manifold to its Tangent Spaces

Let $\text{Sym}^n_+$ denote the SPD manifold. $\text{Sym}^n$ denotes the symmetric matrix space, i.e., the tangent space of $\text{Sym}^n_+$. We define the transformation: $\text{log} : \text{Sym}^n_+ \rightarrow \text{Sym}^n$. For any $X \in \text{Sym}^n_+$, then $X = U \text{diag}(\lambda_1, \ldots, \lambda_n) U^T$, where $\lambda_1 \geq \cdots \geq \lambda_n > 0$ is the eigenvalue of $X$. In addition, $U$ is an orthogonal matrix composed of the orthonormal eigenvectors of $X$. The $\text{log}$ is defined as follows:
\[
\log (X) = U \text{diag}(\log(\lambda_1), \ldots, \log(\lambda_n))U^T \in \text{Sym}^n \tag{7}
\]

Some studies also use \( \log \) transformation, but it seems that they did not mention the following two characteristics:

1. The \( \log \) transformation is an isometric transformation under the affine invariant Riemannian metric. That is, the geodesic matrix between \( X \) and the identity matrix is equal to the Euclidean matrix between \( \log(X) \) and the origin of the tangent space \( \text{Sym}^n \). The affine invariant Riemannian metric on the SPD manifold is defined as follows: \( X \in \text{Sym}^n \), and \( T_x(\text{Sym}^n) \) are the tangent space of the SPD manifold \( \text{Sym}^n \) in \( X \), for any \( \langle \Phi, \theta \rangle_x \in T_x(\text{Sym}^n) \),

\[
\partial \Phi(x) = \{X^{-1} \Phi X^{-1}, X^{-1} \Phi X^{-1}\}_{f}
\tag{8}
\]

If \( X = I_n \) (identity matrix), the inner product and Euclidean distance in \( T_x(\text{Sym}^n) \) are as follows:

\[
\langle \Phi, \Phi \rangle_x = \langle \Phi, \Phi \rangle_f, \quad ||\Phi - \Phi||_f = ||\Phi - \Phi||_x
\tag{9}
\]

The geodetic distance derived from the affine invariant Riemannian metric is as follows:

\[
\text{geoD}(X,Y) = \log(Y^{-1}XY^{-1})_f
\tag{10}
\]

Then we have:

\[
\text{geoD}(X,I_n) = ||\log(X)||_f = ||\log(X)||_x
\tag{11}
\]

\( \log(X) \) is the Euclidean distance between the origin and \( \log(X) \) in \( T_x(\text{Sym}^n) \).

2. The \( \log \) transforms the SPD matrix into a symmetric matrix. It keeps the form of the symmetric matrix of the data, and only performs \( \log \) transformation on the eigenvalues of the SPD matrix. Since \( \log \) is a monotonically increasing function, it thus maintains the order of the eigenvalues of the SPD matrix. Furthermore, the \( \log \) changes the SPD matrices the least in form and nature.

**B. Dictionary Learning of SPD Data Based on Riemannian Manifold Tangent Spaces**

Given a set of dictionary learning samples \( \{X_1, \ldots, X_s\} \subseteq \text{Sym}^n \) on the SPD manifold. We first transform these samples into the tangent space as: \( \{\log(X_1), \ldots, \log(X_s)\} \subseteq \text{Sym}^n \). Then we use the linear combination of these tangent vectors to construct the dictionary in the tangent space of the SPD manifold as follows:

\[
E_s = \sum b_i \log(X_i) \in \text{Sym}^n, \quad i = 1, \ldots, M
\tag{12}
\]

where \( M \) is the number of dictionaries. Then we derive the following expression of the dictionary generated matrix \( B \):

\[
B = \begin{bmatrix}
b_{11} & \cdots & b_{1N} \\
b_{21} & \cdots & b_{2N} \\
\vdots & \ddots & \vdots \\
b_{M1} & \cdots & b_{MN}
\end{bmatrix}
\tag{13}
\]

Apparently, the dictionary \( \{E_1, \ldots, E_s\} \) is completely determined by the dictionary generated matrix \( B \) in the tangent space \( \text{Sym}^n \).

Therefore, the model of dictionary learning based on SPD manifold tangent space is as follows:

\[
\sum_{i=1}^{M} \|\log(X_i) - \sum_{j=1}^{N} a_{ij} E_j\|_{X} = \sum_{i=1}^{M} + \lambda_D \sum_{i=1}^{M} \|A_{Row}\|_{x} \rightarrow \min
\tag{14}
\]

where

\[
A = \begin{bmatrix}
a_{11} & \cdots & a_{1M} \\
a_{21} & \cdots & a_{2M} \\
\vdots & \ddots & \vdots \\
a_{N1} & \cdots & a_{NM}
\end{bmatrix}
\tag{15}
\]

is the dictionary coding matrix, and \( A_{Row} \) is the row vector of \( A \), representing the dictionary coding of \( \log(X_i) \), where \( i = 1, \ldots, N \).

**C. Local Homeomorphism between SPD Data and Their Dictionary Coding**

According to the mathematical definition of manifolds, the manifold is a topological space that is locally homeomorphic with the Euclidean space. In the practical applications, the so-called manifold refers to the SPD manifold, and the Euclidean space locally homeomorphic with the SPD manifold is the tangent space of each SPD matrix. Therefore, if two SPD matrices \( X_i \) and \( X_j \) are sufficiently close, their tangent vectors \( \log(X_i) \) and \( \log(X_j) \) will also be sufficiently close. However, the dictionary coding of \( \log(X_i) \) and \( \log(X_j) \) is the global coding. Although there is the limitation of sparse coding, it cannot guarantee that \( \sum_{j=1}^{N} a_{ij} E_j \) and \( \sum_{j=1}^{N} a_{ji} E_j \) are local homeomorphic to \( X_i \) and \( X_j \) after dictionary coding. Therefore, a local isomorphic regular term is added here to remedy this deficiency.

Since the dictionary coding of the tangent vectors \( \log(X_i) \) and \( \log(X_j) \) is completely determined by the row vectors \( A_{Row} \) of the dictionary coding matrix, the local homeomorphism model is as follows:

\[
\sum_{i=1}^{M} \sum_{j=1}^{N} e^{-\beta \text{geoD}(X_i, X_j)} \|A_{Row} - A_{Row}\|_{x} \rightarrow \min
\tag{16}
\]

Here \( \beta > 0 \), \( \text{geoD}(X_i, X_j) \) is the geodesic distance between the two SPD matrices \( X_i \) and \( X_j \), and we can derive that:

\[
\sum_{i=1}^{M} \sum_{j=1}^{N} e^{-\beta \text{geoD}(X_i, X_j)} \|A_{Row} - A_{Row}\|_{x} = \text{tr}(A^T L A) \rightarrow \min
\tag{17}
\]

Where

\[
L = D - W, \quad W = \begin{bmatrix}
w_{11} & \cdots & w_{1N} \\
w_{21} & \cdots & w_{2N} \\
\vdots & \ddots & \vdots \\
w_{N1} & \cdots & w_{NM}
\end{bmatrix}, \quad D = \text{diag}(\sum_{i=1}^{N} w_{i1}, \ldots, \sum_{i=1}^{N} w_{IN})
\tag{18}
\]

\[
w_{ij} = e^{-\beta \text{geoD}(X_i, X_j)}, \quad i, j = 1, \ldots, N
\]

**D. Dictionary Learning of Symmetric Positive Definite Data Based on the Riemannian Manifold Tangent Space and Local Homeomorphism (RLH-SDL)**

1. **MODEL**

Based on the model described in Section 4.2 and Section 4.3, we propose a SPD data dictionary learning model based on
the Riemannian manifold tangent space and local homeomorphism (RLH-SLD) as follows:

\[ \sum_{i=1}^{N} \log (X_i) = \ni \sum_{\iota=1}^{N} (a_{\iota} E_{\iota}) + \lambda_1 \sum_{\iota=1}^{N} |A_{\iota}|, \]

where \( \lambda_1 > 0 \), \( \lambda_2 > 0 \). In the RLH-SLD, there are two optimization variables: dictionary coding matrix \( A \) and dictionary generated matrix \( B \). The dictionary generated matrix \( B \) is necessary, and the dictionary coding matrix \( A \) is a by-product. From the dictionary learning point of view, it is not necessary. However, the dictionary encoding matrix is indispensable in order to compute the dictionary generated matrix \( B \).

2) OBJECTIVE FUNCTION

Because

\[ \{E_r, E_s\} = \left\{ \sum_{\iota=1}^{N} b_{\iota} \log (X_i), \sum_{\iota=1}^{N} b_{\iota} \log (X_i) \right\} = B_{\iota} KB_{\iota}^{T} \]

Where

\[ K = \begin{bmatrix} \log (X_1), \log (X_1) \\ \vdots \\ \log (X_N), \log (X_N) \end{bmatrix} \]

Then

\[ \{E_r, E_s\} = B_{\iota} KB_{\iota}^{T} \]

Furthermore, we have the following:

\[ \log (X_i) = \sum_{\iota=1}^{N} a_{\iota} E_{\iota} = \sum_{\iota=1}^{N} a_{\iota} E_{\iota} - \sum_{\iota=1}^{N} a_{\iota} E_{\iota} \]

So

\[ \sum_{\iota=1}^{N} \log (X_i) = \sum_{\iota=1}^{N} a_{\iota} E_{\iota} \]

Finally, the objective function of RLH-SLD is as follows:

\[ \min_{A,B} tr (A K^{T}) + tr (ABKB^{T} A^{T}) + \lambda_1 tr (A^{T} L A) + \lambda_2 |A_{\iota}| \]

\[ \text{(26)} \]

E. Solution to RLH-SLD

In the objective function of RLH-SLD, there are two optimization variables: the dictionary coding matrix \( A \) and the dictionary generated matrix \( B \). We use an alternating and iterative method to solve \( A \) and \( B \) sequentially.

First, we fix the dictionary generated matrix \( B \) and calculate the dictionary coding matrix \( A \). Then, the objective function can be simplified as follows:

\[ \min_{A} tr (ABKB^{T} A^{T}) + \lambda_1 tr (A^{T} L A) + \lambda_2 |A_{\iota}| \]

\[ \text{(26)} \]

We use Feature-Sign Search Method [26] to solve it. Then, we fix the dictionary coding matrix \( A \) and calculate the dictionary generated matrix \( B \). The objective function can be simplified as follows:

\[ f(B) = tr (ABKB^{T} A^{T}) - 2tr (ABK) \rightarrow \min \]

We take the partial derivative of Eq. (27) as follows:

\[ f(B) = 2A^{T} BK - 2AK = 0 \rightarrow B = \left(A^{T}\right)^{-1} A \]

The above procedure is repeated until the objective function is smaller than the preset threshold.

Algorithm 1 summarizes the main flow of the RLH-SLD algorithm.

Algorithm 1 RLH-SLD algorithm

Input: dictionary training data \( \mathbf{X} = \{X_1, \ldots, X_N\} \subseteq \text{Sym}_n^+ \), numbers of dictionaries \( M \), and maximum number of iterations \( T_{\text{iter}} \).

Output: dictionary generated matrix \( B \).

1. Initialization: initialized dictionary \( E \) is generated by SPD manifold clustering algorithm, calculate the Graph Laplacian matrix \( L \), and set \( t = I \); repeat
2. Update dictionary generated matrix \( A \) : update dictionary coding matrix \( A \) through Eq. (26);
3. Update dictionary generated matrix \( B \) : update dictionary generated matrix \( B \) through Eq. (28);
4. \( t + + \);
5. Up to \( T_{\text{iter}} \) times;
6. Return Dictionary generated matrix \( B \).

F. Complexity Analysis

We first transform the SPD data into the tangent space of the SPD manifold, i.e., a symmetric matrix space, and then performs dictionary learning on the symmetric matrix space. The subsequent applications of the dictionary are also carried out in the symmetric matrix space. In this subsection, we analyze the computational complexity of RLH-SLD. Given a set of dictionary learning data \( \mathbf{X} = \{X_1, \ldots, X_N\} \subseteq \text{Sym}_n^+ \), \( N \) is the number of samples, and \( \text{Sym}_n^+ \) represents SPD matrix set of \( D \times D \). We first need to compute the Graph Laplacian matrix \( L \). For all \( X, Y \in \text{Sym}_n^+ \), \( \text{geoD}(X, Y) = \log (X^{-1/2} YX^{-1/2}) \), the geodesic distance \( \text{geoD}(X, Y) \) requires matrix multiplication, inversion and decomposition operations, and the corresponding time complexity is \( o(D^{3}) \). Moreover, the time complexity of the matrix norm is \( o(D^{3}) \). Therefore, for a similarity coefficient \( w_{ij} \), its time complexity is \( o(D^{3} + D^{2}) \), then for \( N \times N \) matrix, the time complexity of \( W \) is \( o(N^{2}(D^{2} + D^{3})) \), which is also the time complexity of the Graph Laplacian matrix \( L \). Then we calculate the time complexity of the matrix \( K \). For an original data \( X_{i} \subseteq \text{Sym}_n^+ \), when it is mapped into the SPD manifold tangent space through the \( \log \) function, its computational complexity comes from the eigenvalue decomposition, which is \( o(D^{3}) \). Then, the time complexity of mapping \( N \) original data to SPD manifold tangent space is \( o(ND^{3}) \). Furthermore, the time complexity of the inner product of two matrices on the SPD
manifold tangent space is $o(D^r)$, and the time complexity of $N \times N$ such inner products in matrix $K$ is $o(N^2D^r)$. Therefore, the total time complexity of the matrix $K$ is $o(N^2D^r + ND^r)$.

After obtaining the matrix $K$ and $L$, we use the iterative method to compute the dictionary coding matrix $A$ and the dictionary generated matrix $B$. The total number of iterations is $T_{iter}$. When calculating the dictionary coding matrix $A$, we adopt Feature-Sign Search Method algorithm, and the time complexity of this algorithm is $o(D^r)$. While the updating equation of dictionary generated matrix $B$ is $B = (AA^T)^{-1} A$, and its time complexity is $o(D^r)$. The total time complexity of RLH-SDL is $O(T_{iter} \times (D^r + D^r) + N^2 (D^r + 2 \times D^r) + ND^r)$.

V. Comparison with State-of-the-Art Algorithms

In this section, we introduce five state-of-the-art algorithms as comparisons to the proposed algorithm.

Since the whole SPD matrix cannot form a linear space in the real number field. However, dictionary encoding is a sparse linear encoding of dictionaries. Therefore, dictionary learning cannot be performed directly on SPD data. Currently, the common approach is to transform the dictionary learning samples into RKHS and perform dictionary learning and application on RKHS. Nevertheless, RKHS can be generated uniquely by a kernel function, and then this kernel function is called the reproduction kernel function of this RKHS. Therefore, different kernel functions imply different RKHSs. In addition, SPD data can form Riemannian manifolds under certain topologies and Riemannian metrics. Therefore, the transformation of SPD data into RKHS can be regarded as the transformation from Riemannian manifolds to RKHS.

As for RSR [9], SNP [19], KLE-DLSC [17], and MKSR [20], they all transform dictionary learning from SPD manifolds to RKHS, but they use different Riemannian metrics and kernel functions. The kernel function used by RSR is as follows:

$$k_i(X,Y) = \exp(-\beta S(X,Y)), k_{i}(X,Y) = \exp(-\beta J(X,Y))$$  \hspace{1cm} (29)

where $S(X,Y)$ and $J(X,Y)$ represent Stein divergence and Jeff divergence respectively, and they are both constructed by Bregman divergence. The kernel function used by SNP is as follows:

$$k(X,Y) = \exp(-\alpha \|\log(X) - \log(Y)\|^2)$$  \hspace{1cm} (30)

KLE-DLSC defines multiplication and scalar multiplication operations on SPD manifolds. Then, on the two multiplication operations and their derived three inner product operations, they define three kernel functions. For $x, y \in \text{Sym}_p^+$, the three kernel functions are as follows:

$$\text{Log-E poly.ker nel} : k_p(x, y) = p_o(x, y)$$

$$\text{Log-E exp.ker nel} : k_e(x, y) = \exp(p_o(x, y))$$

$$\text{Log-E Gaus.ker nel} : k_g(x, y) = \exp(-\|x - y\|^2)$$  \hspace{1cm} (31)

where $p_o$ is the polynomial function with positive coefficients.

MKSR makes comprehensive use of the work of KLE-DLSC and the Belkin kernel framework. Given an SPD sample set $\{X_1, \ldots, X_{n}\} \subseteq \text{Sym}_p^+$, it adopts the following kernel function:

$$k(X,Y) = k(X,Y) - \mu \tilde{K}(X)(I_n + LK)^{-1} L\tilde{K}(Y)$$  \hspace{1cm} (32)

Here $L$ is the Graph Laplacian matrix of the sample set $\{X_1, \ldots, X_{n}\}$, and $k(X,Y)$ is an arbitrary kernel function,

$$K = \begin{pmatrix} k(x_1,x_1) & \cdots & k(x_1,x_{n}) \\ \vdots & \ddots & \vdots \\ k(x_{n},x_{1}) & \cdots & k(x_{n},x_{n}) \end{pmatrix} \in R^{n \times n}, \tilde{K}(X) = \begin{pmatrix} k(x_1,x_1) \\ \vdots \\ k(x_{n},x_{n}) \end{pmatrix} \in R^{n \times 1}$$

MKSR transforms $k(X,Y)$ into a sample-dependent kernel function $\tilde{k}(X,Y)$. This sample-dependent kernel function is proposed and proved by [17].

RSR, SNP, KLE-DLSC, and MKSR are all performed on RKHS. However, our algorithm first transforms the SPD data from the SPD manifold into the tangent space of the SPD manifold, i.e., the symmetric matrix space. Compared with the dictionary learning on RKHS, our algorithm retains the original form and nature of the SPD data to the greatest extent.

Because the linear combination of SPD matrices cannot be guaranteed to be an SPD matrix, the entire SPD data cannot constitute a linear space. However, if the combination coefficient is limited to non-negative or at least one combination coefficient is guaranteed to be positive, the linear combination of SPD matrix is still an SPD matrix. Riem-DLSC [15] proposed a dictionary learning algorithm on SPD manifolds under the condition of non-negative linear combination as follows:

$$\sum_{i=1}^{n} \text{geoD}(X_i, \sum_{i=1}^{n} \alpha_i E_i) + \lambda \text{exp} A_{\text{geoD}} \rightarrow \min$$  \hspace{1cm} (34)

where $E_1, \ldots, E_n \in \text{Sym}_p^+$ is the dictionary, $\{X_1, \ldots, X_{n}\} \subseteq \text{Sym}_p^+$ is the dictionary learning samples, $A \in R^{n \times n}$ is a non-negative matrix. There is at least one positive number in the same rows of $A$. geoD is the geodetic distance of the SPD manifolds. The geodesic distance is determined by the Riemannian metric, and Riem-DLSC uses the affine invariant Riemannian metric [27]-[29].

Riem-DLSC performs dictionary learning directly on the SPD datasets. It defines the non-negative linear combination operation and this increases the computational complexity.

VI. Experimental Results

Here, we present the experimental results of RLH-SDL and the five state-of-the-art algorithms presented in Section 5 on four public datasets. In the experiment, we first divided the data into two parts: A and B. A is used to learn the dictionary. In the process of dictionary learning, we also calculate the dictionary coding of A at the same time. After obtaining the dictionary, we use the obtained dictionary to encode B. Then, the dictionary coding of A is used as the learning samples of the classifier, while the dictionary coding of B is used as the
testing samples of the classifier. Thus, the obtained classification accuracy of B is used as the measure index of the performance of each algorithm. Since the focus of this paper is not classification algorithms, we only use simple classifiers: the Ridge Regression (CRR) and the K-Nearest Neighbor (KNN) classification algorithm.

A. The Experimental Results on QMUL Dataset

1) DATASET DESCRIPTION

The QMUL dataset [30] is the human head images collected by airport terminal cameras. The entire dataset contains 20005 images, and the original images is 50 × 50 pixels. The original dataset has been divided into two parts before processing. The first part is the training dataset, with a total of 11,280 images, and the second part is the testing dataset, with a total of 8725 images. From different shooting angles of human heads, the entire dataset is divided into five subsets: 1) ‘back’ represents the images taken from the back of the head, 2) ‘background’ represents the background images, 3) ‘front’ represents the images taken from the front of the head, 4) ‘left’ means the images taken from the left side of the head, and 5) ‘right’ means the images taken from the right side of the head. Some sample images are shown in Figure 1. The number of datasets in each subset is shown in Table 2.

Table 2. QMUL dataset distribution.

| Class          | Back | Background | Front | Left | Right |
|----------------|------|------------|-------|------|-------|
| Number of training dataset | 2256 | 2256 | 2256 | 2256 | 2256 |
| Number of testing dataset  | 2096 | 1107 | 1772 | 1502 | 2248 |

Each image is 50 × 50 pixels in the original QMUL dataset. In our experiment, we first generate a 13-dimensional vector for each pixel of the image as follows:

\[
[I_x(x,y), I_y(x,y), I_z(x,y), \sqrt{I_x^2 + I_y^2 + I_z^2}, G_1(x,y), \ldots, G_8(x,y)]
\]

(35)

where \(I_x(x,y)\), \(I_y(x,y)\) and \(I_z(x,y)\) are the values of CIELab color space from three channels. \(I_x\) and \(I_y\) are the gradient values of \(I_x(x,y)\) in the \(x\) and \(y\) direction respectively. \(G_i(x,y), i = 1, \ldots, 8\) is the response value of 8 DOG (Difference of Offset Gaussian) filters at \((x,y)\). Therefore, a 50×50 image generates a 2500 13-dimensional vector group, and the covariance matrix of this vector group is the SPD matrix of the image.

2) THE EFFECT OF THE NUMBER OF DICTIONARIES ON RLH-SDL

The dictionary learning selects or reselects a certain number of dictionaries from the original dataset, and uses the selected dictionary to perform the sparse linear representation of the original data. Therefore, the number of dictionaries directly affects the result. If the number of dictionaries is too small, some data information may be lost and cannot be expressed effectively. If the number of dictionaries is too large, the computational complexity will be too high.

To explore the effect of the number of dictionaries on RLH-SDL, in our experiments, we randomly select 200 images from each subset of the training dataset as the training samples for dictionary learning. Then, we randomly select 300 images from each subset of the test dataset. Where we select 200 images as training samples for the classifier and 100 images as testing samples for the classifier to calculate the classification accuracy. The training and testing of the classifier are randomly repeated 10 times, and the average value of the classification accuracy is used as the final experimental result. Here we use two classifiers to evaluate the accuracy of the algorithms, i.e., the ridge regression-based classifier (CRR) and the K-nearest neighbor (KNN) classification algorithm.

For RLH-SDL, the regular term \(\lambda_{sp}\) is set to 0.1, and the sparse term \(\lambda_{sr}\) is set to 0.5. For the number of dictionaries, we set it from 20 and increase it in each experiment by 10 until it reaches 200.

Figure 2 shows the effect of the number of dictionaries on classification accuracy of RLH-SDL. We observe that the classification accuracy of the two classifiers increases rapidly with the increase of the number of dictionaries. However, when the number of dictionaries reaches 50, the classification accuracy saturates. Figure 3 shows the effect of the number of dictionaries on the running time of RLH-SDL. We found that if the number of dictionaries is continuously increased, the computational complexity of the algorithm keeps increasing. However, the accuracy of the algorithm does not change much, so the gain is not worth the loss.
3) EXPERIMENTAL RESULTS OF RLH-SDL AND COMPARISON ALGORITHM

Table 3 shows the experimental results of the proposed RLH-SDL and other five comparison algorithms. The chosen parameters of RLH-SDL are $\lambda_{sp} = 0.1$, $\lambda_{sr} = 0.5$.

| Method                  | CRR   | KNN   |
|-------------------------|-------|-------|
| Riem-DLSC-ARM [15]      | 36.6  | 38.6  |
| RSR-S [9]               | 73.2  | 57.3  |
| RSR-J [9]               | 70.0  | 69.2  |
| KLE-DLSC-Poly [17]      | 66.7  | 54.3  |
| KLE-DLSC-Exp [17]       | 65.6  | 56.4  |
| KLE-DLSC-Gaussian [17]  | 72.4  | 54.5  |
| MKSR [20]               | 69.3  | 68.3  |
| SNP [19]                | 69.4  | 68.7  |
| RLH-S SDL               | 75.7  | 71.6  |

TABLE 3. Classification accuracy of dictionary learning algorithms based on SPD manifolds on QMUL dataset (%).

As can be seen from Table 3, the classification accuracy of Riem-DLSC, which directly models the SPD manifold, is much lower than that of the other algorithms. Among the algorithms modeled by RKHS, RSR-S with the Stein kernel function achieved the highest accuracy on the CRR classifier, while RSR-J with the Jeffery kernel function had the highest accuracy on the KNN classifier. KLE-DLSC with three kernel functions based on the LE kernel framework performs poorly on the QMUL dataset, only outperforming Riem-DLSC. The MKSR algorithm with data dependency and the SNP algorithm perform close to RSR, but still worse than RSR.

Compared with the other five algorithms, the proposed RLH-SDL algorithm achieves the highest classification accuracy on the QMUL dataset. With the CRR classifier, the accuracy of RLH-SDL is 2.5% higher than RSR-S, while it is 2.4% higher than RSR-J with the KNN classifier. This indicates that the RLH-SDL is more suitable for QMUL dataset. The local homeomorphism can preserve the similarity between the original data, so the sparse coding results can better preserve the geometric structure of the original dataset. In summary, we conclude that the RLH-SDL algorithm is effective on the QMUL dataset.

B. THE EXPERIMENTS ON ETHZ DATASET

1) DATASET DESCRIPTION

The original ETHZ dataset [31] is captured using a mobile camera, which provides a series of variations in human appearance. The dataset is divided into 3 sequences, where each sequence contains 4,857 images of 83 pedestrians, 1,936 images of 35 pedestrians, and 1,762 images of 28 pedestrians, respectively. Figure 4 shows some example images.

![Example images of the ETHZ dataset.](image)

Each image has $640 \times 480$ pixels. To convert the original image to SPD data with the regional covariance descriptor, we first sample all images into $64 \times 32$ pixels, and then generate a 17-dimensional vector for each pixel as follows:

$$\left[ I_x(x,y), I_y(x,y), I_x(x,y) \cdot I_y(x,y), I_x(x,y) \cdot I_y(x,y), I_x(x,y) \cdot I_y(x,y), \ldots, G(x,y), \ldots, G(x,y) \right]$$

where $I_x(x,y)$, $I_y(x,y)$ and $I(x,y)$ are the values of CIELab color space from three channels. $I_x$, and $I_y$ are the gradient values of $I(x,y)$ in the directions of $x$ and $y$ respectively. $G(x,y)$ is the response values of 8 DOG (Difference of Offset Gaussian) filters at $(x,y)$. Therefore, an $64 \times 32$ image produces a vector group containing $64 \times 32$ 17-dimensional vectors. The covariance matrix of the vector group is the characteristic matrix of this image, and the characteristic matrix is an SPD matrix.

2) THE EXPERIMENTAL RESULTS

In the experiment, we combine all three sequences of ETHZ into an overall dataset. It contains 146 pedestrians and 8555 pictures. For each pedestrian, we randomly select 20 pictures as learning samples of the dictionary. The rest are used as testing samples of the classifier. The regular item $\lambda_{sp}$ is set close to 0.1 and the sparse item $\lambda_{sr}$ is set to 0.5. The samples will be randomly selected for 20 times in the experiment, and the average of the results will be the final result. Table 4 shows the experimental results.

As can be seen from Table 4, the Riem-DLSC algorithm, which directly models the SPD manifold, performs the worst, while the RSR-J with Jeffery kernel function performs the best among the five compared algorithms. The KLE-DLSC-Gaussian algorithm using the LE kernel framework, the MKSR algorithm using the Belkin kernel function framework and data dependence, and the SNP algorithm using the logarithmic Euclidean kernel function and Graph Laplacian regularization perform similarly, but they all perform poorly. The difference with RSR-J is clear.
of the image, we define $I(x, y)$. We can see that RLH is
$e^{-\frac{1}{2}}$. Then, $\lambda_{\text{min}}$ is set to 0.1, and the sparse term $\delta_{\text{sp}}$ is set to 0.5.

Table 5 shows the classification accuracy of each dictionary learning algorithm. We can see that RLH-SDL achieves the best classification accuracy. However, the

accuracy of Riem-DLSC, which is directly modeled on the SPD manifolds, is inferior to the other comparison algorithms. For the methods modeled on RKHS, the accuracy of the SNP algorithm with Log-Euclidean kernel function and Graph Laplacian regularization achieved the highest accuracy on the CRR classifier. While the accuracy of the RSR-J algorithm with Jeffery kernel function reaches the highest on the KNN classifier. The RSR-S algorithm with the Stein kernel function, the KLE-DLSC-Gaussian with the LE kernel function framework, and the MKSR with the Belkin kernel function framework and data dependencies perform relatively poorly.

C. The Experimental Results on COIL Dataset

1) DATASET DESCRIPTION

The original COIL dataset [32] is a common dataset for object recognition tasks. It contains a total of 1440 grayscale images, and each image in the original dataset is 128 x 128 pixels. These images are from 20 objects. Since each object is captured on a turntable with a 360-degree rotation, and the images of each object are separated by 5 degrees. Therefore, there are 72 images for each object. Figure 5(a) shows the 20 objects and Figure 5(b) shows the 72 images of an object.

![Sample images of the COIL dataset](image)

(a) (b)

**FIGURE 5.** Sample images of the COIL dataset: (a) 20 object pictures of COIL dataset; (b) All pictures of an object in the COIL dataset.

For each pixel $(x, y)$ of the image, we define the 15-dimensional characteristic vector. The generation of it is as follows:

$$[x, y, I(x, y), \sqrt{|I_x|^2 + |I_y|^2}, \arctan\left(\frac{|I_y|}{|I_x|}\right), G_1(x, y), \ldots, G_8(x, y)]^T$$  \hspace{1cm} (37)

where $I(x, y)$ represents the gray value of the image on $(x, y)$. $I_x$ and $I_y$ are the gradient values of $I(x, y)$ in the $x$ and $y$ respectively. And $G_i(x, y)$, $i = 1, \ldots, 8$ is the response values of 8 DOG (Difference of Offset Gaussian) filters at $(x, y)$. Then, each 15x15 image can generate a vector group including 225 15-dimensional vectors. The covariance matrix of this vector group is the SPD matrix that characterizes the image.

2) THE EXPERIMENTAL RESULTS

In our experiments, we randomly select 20 images from each object as training samples for dictionary learning. The remaining 52 images are used as test samples for classification. The training and test samples were randomly selected 20 times. We calculate the classification accuracy after each selection, and the average of the results is taken as the final result. The regular term $\lambda_{\text{min}}$ is set to 0.1, and the sparse term $\delta_{\text{sp}}$ is set to 0.5.

Table 5 shows the classification accuracy of each dictionary learning algorithm. We can see that RLH-SDL achieves the best classification accuracy. However, the

| Method                  | Classification accuracy (%) |
|-------------------------|----------------------------|
| RSR-S [9]               | 70.5                       |
| RSR-J [9]               | 74.3                       |
| Riem-DLSC-Frob [15]     | 22.1                       |
| KLE-DLSC-Gaussian [17]  | 64.2                       |
| MKSR [20]               | 64.7                       |
| SNP [19]                | 62.7                       |
| RLH-SDL                 | 80.0                       |

**TABLE 4.** Classification accuracy of each dictionary learning algorithm on the ETHZ dataset (%).

D. The Experimental Results on PIE Dataset

1) DATASET DESCRIPTION

The PIE dataset [33] is a facial image dataset, where PIE represents "Pose", "Illustration" and "Expression". The PIE dataset contains 41368 grayscale facial images of 68 individuals under different poses, illustrations and expressions. Each image is 64x64 pixels in the PIE dataset. In this experiment, we select 5 subsets from the dataset according to the different poses of individuals, i.e., PIE05 for the left pose, PIE07 for the upward pose, PIE09 for the downward pose, PIE27 for the front pose and PIE29 for the right pose. In each subset, all facial images were taken under different illustration conditions or expression conditions. Figure 6 shows some samples of the selected five subsets.

![Example images of the 5 subsets of the PIE dataset](image)

**FIGURE 6.** Example images of the 5 subsets of the PIE dataset.

In the experiment, for each pixel of the image, we define a 43-dimensional characteristic vector. The generation of it is as follows:

$$[I(x, y), I(x, y), G_{\text{xx}}(x, y), \ldots, G_{\text{xy}}(x, y)]^T$$  \hspace{1cm} (38)

where $(x, y)$ is the position coordinate, $I(x, y)$ is the gray value at $(x, y)$, and $G_i(x, y)$ is the response value of the two-
dimensional Gabor filter at the \((x,y)\). The \(u\) of the Gabor filter \(g_u\), represents the direction, and its value ranges from 0 to 4. The \(v\) represents the scale, and its value ranges from 0 to 7. As a result, there are 40 Gabor filters in total. Therefore, a 64x64 image produces a vector group containing 4096 43-dimensional vectors, and the covariance matrix of this vector group is the SPD matrix that characterizes this image.

2) THE EXPERIMENTAL RESULTS

In the experiment, we randomly select 20 images of each person in the selected PIE05 subset as the training dataset for dictionary learning. The remaining images are used as the test dataset of the classifier. PIEALL represents the entire PIE dataset. For the experiment of PIEALL, we randomly choose 70 images of each person as the training dataset for dictionary learning, and the rest are used as the test dataset. The above process is repeated 20 times to obtain reliable statistics. The regular term \(\lambda_u\) is set to 0.1, and the sparse term \(\lambda_v\) is set to 0.5. Here we use the KNN classifier.

We compare the performance of the proposed algorithm with other SPD data DLSR algorithms, including KLE-DLSC, RSR, Riem-DLSC, MKSR, SNP and BLKM-DL. Table 6 shows the result comparisons.

| Method               | PIE05 | PIE07 | PIE09 | PIE27 | PIE29 | PIEALL | Average |
|----------------------|-------|-------|-------|-------|-------|--------|---------|
| Riem-DLSC-AIRM [15]  | 33.8  | 36.1  | 38.6  | 37.8  | 34.9  | 22.2   | 33.9    |
| Riem-DLSC-Froeb [15] | 27.6  | 30.5  | 36.0  | 32.4  | 29.4  | 19.9   | 29.3    |
| RSR-S [9]            | 83.2  | 83.6  | 89.3  | 82.8  | 85.7  | 72.9   | 82.9    |
| KLRM-DL [22]         | 75.9  | 80.7  | 87.1  | 77.8  | 87.8  | 68.9   | 79.5    |
| KLE-DLSC-Gaussian [17]| 44.2  | 43.1  | 54.8  | 51.4  | 52.9  | 35.4   | 47.0    |
| MKSR [20]            | 61.9  | 66.5  | 78.3  | 60.5  | 62.9  | 50.9   | 63.5    |
| SNP [19]             | 62.9  | 58.4  | 75.4  | 65.6  | 69.9  | 52.2   | 64.1    |
| RLH-SDL              | 88.8  | 88.5  | 92.7  | 90.0  | 90.1  | 82.1   | 88.4    |

From Table 6, we can see that the Riem-DLSC algorithm, which models directly on SPD manifolds, performs very poorly on all subsets of PIE and the entire PIE dataset. There is also a significant gap with other comparison algorithms. Therefore, it can be seen that the method of modeling directly on the SPD manifold is not suitable for the PIE dataset. For the other comparison algorithms, the accuracy of the RSR-S with the Stein kernel function is the highest on the four subsets of PIE05, PIE07, PIE09, PIE27 and PIEALL. While the accuracy of the KLRM-DL algorithm with the Log-Euclidean kernel function and data dependence reaches the highest on the PIE29. The average accuracy of RSR-S is the highest among the comparison algorithms.

The RLH-SDL algorithm proposed in this paper achieves the best performance in each subset and the entire PIE dataset. Therefore, it can show that the RLH-SDL is more suitable for the PIE dataset, which is mapped the original data into the tangent space of the SPD Riemannian manifold. The Graph Laplacian regularization can well retain the similarity between the original data, so that the sparse coding is more discriminative.

VOLUME XX, 2017

REFERENCES

[1] Peclat C, Bougleux G, Chevet U. Extracting Most Impacting Emergency Department Patient Flow by Embedding Laboratory confirmed and Clinical Diagnosis on The Stiefel Manifold[C] // 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). 2019: 1 - 4.

[2] Du Y, Zhu G, Zhang J, et al. Automatic recognition of space-time constellations by learning on the Grassmann manifold[J]. IEEE Transactions on Signal Processing, 2018, 66(22): 6031 – 6046.

[3] Fehr D, Cherian A, Sivalingam R, et al. Compact covariance descriptors in 3d point clouds for object recognition[C] // 2012 IEEE International Conference on Robotics and Automation. 2012: 1793 – 1798.

[4] Cherian A, Morellas V, Papanikolopoulos N. Bayesian nonparametric clustering for positive definite matrices[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 38(5): 862 – 874.

[5] Matsukawa T, Okabe T, Suzuki E, et al. Hierarchical gaussian descriptors with application to person re-identification[J]. IEEE transactions on pattern analysis and machine intelligence, 2019, 42(9): 2179 –2194.

[6] Somasundaram G, Cherian A, Morellas V, et al. Action recognition using global spatio-temporal features derived from sparse representations[J]. Computer Vision and Image Understanding, 2014, 123: 1 – 13.

[7] Dodero L, Minh H Q, San Biagio M, et al. Kernel-based classification for brain connectivity graphs on the Riemannian manifold of positive definite matrices[C] // 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI). 2015: 42 – 45.

[8] Porikli F, Tuzel O, Meer P. Covariance tracking using model update based on lie algebra[C] // 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06): Vol 1.2006: 728 – 735.

[9] Zhang Z, Li F, Chow T W, et al. Sparse codes auto-extractor for classification: A joint embedding and dictionary learning framework for representation[J]. IEEE Transactions on Signal Processing, 2016,64(14): 3790 – 3805.

[10] Leal N, Moreno S, Zurek E. Simple method for detecting visual saliencies based on DLSR[C] // 2019 14th Iberian Conference on Information Systems and Technologies (CISTI). 2019: 1 – 5.
[11] Harandi M T, Hartley R, Lovell B, et al. Sparse coding on symmetric positive definite manifolds using bregman divergences[J]. IEEE transactions on neural networks and learning systems, 2015, 27(6): 1294 – 1306.

[12] Jayasumana S, Hartley R, Salzmann M, et al. Kernel methods on Riemannian manifolds with Gaussian RBF kernels[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 37(12): 2464 – 2477.

[13] Sivalingam R, Boley D, Morellas V, et al. Tensor sparse coding for positive definite matrices[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 36(3): 592 – 605.

[14] Sivalingam R, Boley D, Morellas V, et al. Tensor dictionary learning for positive definite matrices[J]. IEEE Transactions on Image Processing, 2015, 24(11): 4592 – 4601.

[15] Cherian A, Sra S. Riemannian DLSR for positive definite matrices[J]. IEEE transactions on neural networks and learning systems, 2016, 28(12): 2859 – 2871.

[16] S. Sra and A. Cherian, “Generalized dictionary learning for symmetric positive definite matrices with application to nearest neighbor retrieval,” in Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases, 2011, pp. 318 – 332.

[17] Li P, Wang Q, Zuo W, et al. Log-Euclidean kernels for sparse representation and dictionary learning[C] // Proceedings of the IEEE international conference on computer vision. 2013: 1601 – 1608.

[18] Das A, Nair M S, Peter S D. Sparse representation over learned dictionaries on the Riemannian manifold for automated grading of nuclear pleomorphism in breast cancer[J]. IEEE Transactions on Image Processing, 2018, 28(3): 1248 – 1260.

[19] Li D, Chen L, Wang F. Semantic and neighborhood preserving dictionary learning for symmetric positive definite matrices[C] // 2016 IEEE 13th international conference on signal processing (ICSP), 2016: 654 – 658.

[20] Wu Y, Jia Y, Li P, et al. Manifold kernel sparse representation of symmetric positive definite matrices and its applications[J]. IEEE Transactions on Image Processing, 2015, 24(11): 3729 – 3741.

[21] Sindhvani V, Niyogi P, Belkin M. Beyond the point cloud: from transductive to semi-supervised learning[C] // Proceedings of the 22nd international conference on Machine learning. 2005: 824 – 831.

[22] Zhuang R, Ma Z, Feng W, et al. SPD data dictionary learning based on kernel learning and Riemannian metric[J]. IEEE Access, 2020, 8: 61956 – 61972.

[23] Zhang X, Li W, Hu W, et al. Block covariance based L1 tracker with a subtle template dictionary[J]. Pattern Recognition, 2013, 46(7): 1750 – 1761.

[24] Guo K, Ishwar P, Konrad J. Action recognition using sparse representation on covariance manifolds of optical flow[C] // 2010 7th IEEE international conference on advanced video and signal based surveillance. 2010: 188 – 195.

[25] Yuan C, Hu W, Li X, et al. Human action recognition under log-Euclidean Riemannian metric[C] // Asian conference on computer vision. 2009: 343 – 353.

[26] Zheng M, Bu J, Chen C, et al. Graph regularized sparse coding for image representation[J]. IEEE transactions on image processing, 2010, 20(5): 1327 – 1336.

[27] M. Moakher and P. G. Batchelor, “Symmetric positive-definite matrices: From geometry to applications and visualization,” in Visualization and Processing of Tensor Fields. Berlin, Germany: Springer, 2006, pp. 285 – 298.

[28] R. Bhatia, Positive Definite Matrices. Princeton, NJ, USA: Princeton Univ. Press, 2007.

[29] X. Pennec, P. Fillard, and N. Ayache, “A Riemannian framework for tensor computing,” Int. J. Comput. Vis., vol. 66, no. 1, pp. 41 – 66, 2006.

[30] Tosato D, Spera M, Cristiani M, et al. Characterizing humans on riemannian manifolds[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 35(8): 1972 – 1984.

[31] Schwartz W R, Davis L S. Learning discriminative appearance-based models using partial least squares[C] // 2009 XXII Brazilian symposium on computer graphics and image processing. 2009: 322 – 329.

[32] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

Hui He received the B.S. degree in electronic information engineering from Wuhan University of Science and Technology, Wuhan, China, in 2020 and she is pursuing a master’s degree with School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China. Her current research interest is machine learning.

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

Zhengming Ma received a B.Sc. degree in Radio Technology and a M.Sc. degree in Electronic and Communication System from South China University of Technology, Guangzhou, China, in 1982 and 1985, respectively, and a Ph.D. degree in Pattern Recognition and Intelligent Control from Tsinghua University, Beijing, China, in 1989. He currently is a Professor with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China. His current research interests include machine learning.

Weijia Feng received the B.Sc. degree from the Guangdong University of Technology, Guangzhou, China, in 2019, and M.Sc. degree with the School of Electronics and Information Technology from Sun Yat-sen University, Guangzhou, China, in 2021. He is currently working in the China Telecom Cloud Technology Co., Ltd.