2016

Developing advanced methods for covariance representations in computer vision

Jianjia Zhang
University of Wollongong

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Developing Advanced Methods for Covariance Representations in Computer Vision

Jianjia Zhang

Supervisor:
A/PR Lei Wang
Co-supervisors:
Dr. Luping Zhou
A/PR Wanqing Li

This thesis is presented as part of the requirements for the conferral of the degree:

Doctor of Philosophy

The University of Wollongong
School of Computing and Information Technology

Nov. 2016

Declaration

I, Jianjia Zhang, declare that this thesis submitted in partial fulfilment of the requirements for the conferral of the degree That Degree You're Studying, from the University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualifications at any other academic institution.

Jianjia Zhang
November 3, 2016

Publications During Pursuing Ph.D

This thesis is mainly developed based on the following five papers which are listed in the order they appear in the thesis.

1. **J. Zhang**, L. Wang, L. Zhou, W. Li. Learning Discriminative Stein Kernel for SPD Matrices and Its Applications. IEEE Transactions on Neural Networks and Learning Systems (IEEE T-NNLS), vol.27, no.5, pp.1020-1033, 2016. (Chapter 3)

2. **J. Zhang**, L. Zhou, L. Wang, and W. Li. Exploring Compact Representation of SICE Matrices for Functional Brain Network Classification. MICCAI Workshop on Machine Learning in Medical Imaging (MLMI), Boston, USA, 2014, pp.59-67. (Chapter 4)

3. **J. Zhang**, L. Zhou, L. Wang and W. Li. Functional Brain Network Classification with Compact Representation of SICE Matrices. IEEE Transactions on Biomedical Engineering (IEEE T-BME), vol.62, no.6, pp.1623-1634, 2015. (Chapter 4)

4. ***J. Zhang**, L. Zhou, L. Wang. Subject-adaptive Integration of Multiple SICE Brain Networks of Different Sparsity, submitted to Pattern Recognition, Minor Revision, 2016.* (Chapter 5)

5. L. Wang, **J. Zhang**, L. Zhou, C. Tang, and W. Li. Beyond Covariance: Feature Representation with Nonlinear Kernel Matrices, International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp.4570-4578. (Chapter 6)

Besides the papers above, I also published the following co-authored papers during my Ph.D course. The work in these publications is not included in this thesis.

1. Z. Gao, L. Wang, L. Zhou, and **J. Zhang**. HEp-2 Cell Image Classification with Deep Convolutional Neural Networks, IEEE Journal of Biomedical and Health Informatics (IEEE JBHI), no.99, 2016.

2. Z. Gao, **J. Zhang**, L. Zhou, and L. Wang. HEp-2 Cell Image Classification with Convolutional Neural Networks. Contest & Workshop on Pattern Recognition Techniques for Indirect Immunofluorescence Images Analysis, held together with ICPR, Sweden, 2014, pp.24-28.

3. **J. Zhang**, L. Wang, L. Liu, L. Zhou, W. Li. Accelerating the Divisive Information-Theoretic Clustering of Visual Words. International Conference on Digital Image Computing Techniques and Applications (DICTA), Tasmania, Australia, 2013, pp.1-8.

4. L. Wang, **J. Zhang**, L. Zhou, and W. Li. A Fast Approximate AIB Algorithm for Distributional Word Clustering. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Oregon, USA, 2013, pp.556-563.

Abstract

Computer vision aims at producing numerical or symbolic information, e.g., decisions, by acquiring, processing, analyzing and understanding images or other high-dimensional data. In many contexts of computer vision, the data are represented by or converted to covariance-based representations, including covariance descriptor and sparse inverse covariance estimation (SICE), due to their desirable properties. While enjoying beneficial properties, covariance representations also bring challenges. Both covariance descriptor and SICE matrix belong to the set of symmetric positive-definite (SPD) matrices which form a Riemannian manifold in a Euclidean space. As a consequence of this special geometrical structure, many learning algorithms which are developed in Euclidean spaces cannot be directly applied to covariance representations because they do not take such geometrical structure into consideration. However, the increasingly wider applications of covariance representations in computer vision tasks urge the need for advanced methods to process or analyze covariance representations.

This thesis aims to develop advanced learning methods for covariance representations in computer vision. This goal is achieved from four perspectives. 1) This thesis first proposes a novel kernel function, discriminative Stein kernel (DSK), for covariance descriptor. DSK is learned in a supervised manner through eigenvalue adjustment. 2) Then this thesis pushes forward the application of covariance representations. This thesis finds that the high dimensionality of SICE matrices can adversely affect the classification performance. To address this issue, this thesis uses SPD-kernel PCA to extract

principal components to obtain a compact and informative representation for classification. 3) In order to fully utilize the complementary information in SICE matrices at multiple sparsity levels, this thesis develops a subject-adaptive integration of SICE matrices for joint representation and classification. 4) Furthermore, considering the issues encountered by covariance descriptor in dealing with the case of high feature dimensionality and small sample size, this thesis generalizes covariance descriptor with a kernel matrix over feature dimensions. By doing this, the fixed form of covariance descriptor is extended to an open framework of kernel matrix based representations.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Research Background

In various computer vision tasks, feature representation is a critical step to transform raw features input to a representation that can be effectively exploited with machine learning algorithms. The raw features can be numerous local features extracted from pixels or patches of an image, e.g. intensity, gradient, location etc., or an series of frame features in an action sequence. Usually, these raw features can hardly be directly used as input of machine learning algorithms due to their large size and substantial noise, instead, a higher level of representation is required. However, how to represent this set of raw features is a challenging issue. The representation should be informative to represent the characteristics of this feature set. Also, the final representation should be consistent to allow comparison between differently sized feature sets.

Over the past decade, covariance-based representations have attracted much attention and been applied to various computer vision and image analysis tasks. One good example of covariance-based representations is covariance descriptor [93], which is widely used in many visual tasks, e.g., image classification, visual tracking and object detection. Essentially, a covariance descriptor assumes that features in one set, e.g.,

all the features extracted from the patches of an image, follow a Gaussian distribution and the covariance of each distribution contains enough information to differentiate it from others. It's not surprising for covariance descriptor to be widely used because there are several advantages to use it as a feature representation. Firstly, a covariance matrix carries the statistical characteristics of feature variables and is usually sufficient to different feature sets. The diagonal entries of the covariance matrix represent the variance of each feature variable while the off-diagonal entries represent the correlations between different feature variables. The effect of noise in each feature can be largely filtered out by an average operation during covariance computation. Secondly, the size of the covariance matrix is independent of the number of feature vectors extracted. Therefore, it allows direct comparison between differently-sized feature sets. Thirdly, the covariance matrix does not contain any information about the order or the number of feature vectors, which implies a certain invariance to image rotation or scaling if such information is not presented in the raw features [93].

Sparse inverse covariance matrix, as another form of covariance representation, has been proposed to model sparse undirected graphs. The estimated graphical models provide a systematic way of identifying conditional dependencies in high-dimensional data, such as gene data, functional Magnetic Resonance Imaging (fMRI), spectroscopy, social and economic data. These dependencies can be interpreted as meaningful interactions among the data variables depending on specific applications. For instance, one important application of sparse inverse covariance matrix in computer vision is modeling functional brain networks based on fMRI [118] data. In this case, the dependencies reflect the co-varying patterns of fMRI time series across brain regions and can be interpreted as functional connectivity between brain regions. The resulting networks or graphs, represented as sparse inverse covariance matrices, can be used for further analysis. For example, functional brain networks can be used to help the diagnosis of brain diseases, such as Alzheimer's disease (AD) [111].

However, how to process and analyze these covariance representations in an effective and efficient manner remains a challenging issue. This is because the covariance representations belong to a set of symmetric positive-definite (SPD) matrices which constitute a Riemannian manifold, specifically, a convex half-cone in the vector space of matrices, rather than a Euclidean space. Commonly used algorithms in image analysis and computer vision, e.g. interpolation, clustering, support vector machines (SVM), partial least squares (PLS) and linear discriminant analysis (LDA), are primarily developed for data points lying in Euclidean spaces. Direct application of these algorithms to SPD matrices usually yields unsatisfactory performance and undesirable effects, such as the swelling effect of diffusion tensors (SPD matrices) [2] in interpolation, due to the ignorance of the special manifold structure.

1.2 Research Questions

This thesis will focus on four key questions related to covariance representations:

- Kernel learning. The use of kernels has received considerable attention and become increasingly common in the last two decades. The main reason of kernels being popular is that they allow to map low-dimensional data into a high-dimensional (or probably infinite-dimensional) feature space in order to increase the efficacy of linear machines [24, 99]. However, how to design a proper kernel for covariance representations to achieve good performance remains challenging. We will explore discriminative kernel learning, making the kernels better align with the data and achieve better performance.

- Extracting features from covariance representations. With obtained covariance representations, how to classify them is a critical step to attain satisfying classification performance. To this end, feature extraction from these covariance representations is worth exploration. Ideal features used for classification should

essentially take advantage of the data distribution and utilize prior knowledge in specific applications. For example, we may need to distinguish Alzheimer's disease (AD) from normal controls by constructing and classifying functional brain networks which are represented by covariance representations. In this application, both AD patients and normal controls are human, therefore, they should have similar functional brain networks, which can be considered as prior knowledge in designing feature extraction methods. This thesis will investigate the characteristics of covariance representations in this specific application and extract a compact and informative feature from covariance representations to boost classification performance.

- Integrating sparse inverse covariance at multiple sparsity levels. As previously mentioned, sparse inverse covariance is a powerful modeling tool for graphs. Sparse inverse covariance estimation (SICE) is often carried out by minimizing a sparsity penalized log-likelihood. The log-likelihood promotes goodness-of-fit of the estimator while the penalty promotes sparsity. In this case, the estimation method naturally generates different SICEs with varying sparsity. This arises an issue: which sparsity level(s) of SICE should we select for further analysis? This thesis argues that selecting an single sparsity level may ignore complementary information contained in other levels and identifying the best single level could be time-consuming. Instead, multiple SICEs at different sparsity levels should be jointly considered to improve classification performance.

- Beyond covariance representation. Although covariance-based representations enjoy desirable properties in theory and outstanding performance in practice, they encounter several issues in new applications. Specifically, covariance representation has shortcomings such as being prone to singularity, limited capability in modeling complicated feature relationship, and having a fixed form of repre-

sentation. It is of great significance to develop an advanced SPD representation that can avoid these issues of covariance representations while inheriting their merits. Motivated by this, this thesis will explore a novel representation beyond covariance to achieve this goal.

1.3 Contributions

The main contributions of this thesis are summarized as follows.

- This thesis proposes a novel kernel function for covariance descriptor through supervised learning. It first analyzes one of the state-of-the-art similarity measures, Stein kernel, for covariance descriptors and identifies two issues with it. Based on the analysis and identification, this thesis proposes a discriminative Stein kernel (DSK) to address the issues, leading to a better similarity measure for covariance descriptors. This work brings forth two advantages: i) DSK mitigates the negative impact of the biasness of covariance estimation to class discrimination; ii) By adaptively learning the adjustment parameters from training data, it makes Stein kernel better align with specific classification tasks. Both advantages help to boost the classification performance of Stein kernel in practical applications. This thesis also provides more insights on when and how DSK works, and discusses the aspects that could contribute to discovering better SPD kernels.

- This thesis also provides a novel method to extract compact and representative features from SICE matrices. Specifically, manifold-based similarity measures and kernel-based PCA are employed to extract principal connectivity components as a compact representation of brain networks. This representation takes advantage of the distribution of SICE matrices and incorporates prior knowledge in functional brain network modeling, boosting the classification accuracy considerably. Moreover, to cater for the requirement of both discrimination and interpreta-

tion in neuroimage analysis, we develop a novel pre-image estimation algorithm to make the obtained connectivity components anatomically interpretable.

- This thesis develops a subject-adaptive integration of SICE matrices at multiple sparsity levels. Firstly, this method utilizes different levels of details complementary for classification. Also, it is noticed that some sparsity levels that are useful for one subject could become less useful for another due to the variation of subject-specific characteristics, e.g., disease phase, age and gender. Considering this, we allow the integration to be subject-adaptive, and achieve this through a sparse representation framework. By the integration, our proposed framework learns a unique, enhanced, and new network representation for each subject in a kernel induced feature space which respects the specific geometrical property of SICE matrix. Although the integration takes place in a feature space, this thesis provides a feasible method to project the integration result back into the original network space for visualization and further analysis. This could help to understand the underlying pathophysiology of brain diseases.

- This thesis generalizes the covariance descriptor to a wider framework of kernel matrix representation. In order to further improve the classification accuracy of covariance descriptor, it is essential to develop an advanced SPD representation which admits more representative power than covariance descriptor. Following this direction, this thesis identifies several shortcomings of covariance descriptor. To address these shortcomings, it proposes an open framework to use the kernel matrix over feature dimensions as a generic representation. The improved expressive power of this representation has been demonstrated in a variety of computer vision tasks. In addition, the resulting representation maintains the SPD property and the size of the original covariance representation, and thus all the methods developed for covariance descriptor can also be applied to this representation. Be-

sides, the computational load of the proposed representation is kept as low as that of covariance descriptor. More importantly, since this work significantly extends the covariance descriptor to a wide range of general kernel matrices, this novel representation holds great promise in various applications considering the fact that there are abundant off-the-shelf kernel functions in computer vision research.

1.4 Outline of This Thesis

The outline of this thesis is as follows.

- Chapter 2 gives an introduction of covariance-based representations and reviews the related literature.
- Chapter 3 analyzes two potential issues of the recently proposed Stein kernel and proposes a novel kernel called discriminative Stein kernel. It automatically manipulates the input covariance descriptors to help Stein kernel to achieve greater discrimination. This problem is formulated as a kernel parameter learning process and solved in three frameworks. The proposed kernel is evaluated on both synthetic and real data sets for a variety of applications in pattern analysis and computer vision.
- Chapter 4 finds that the high dimensionality of SICE matrices can adversely affect the classification performance. Taking advantage of the SPD property of SICE matrices, this chapter uses SPD-kernel PCA to extract principal components to obtain a compact representation for classification. This chapter also proposes a pre-image estimation algorithm, which allows visualization and analysis of the extracted principal connectivity patterns in the input space. The efficacy of the proposed method is verified by extensive experimental study on synthetic data and real rs-fMRI data.

- Chapter 5 explores the complementary information in a set of SICE matrices at different sparsity levels. To this end, this chapter proposes a learning framework that integrates networks while respecting the underlying manifold structure of the SPD network representations. The proposed framework conducts a subject-adaptive integration via a kernel sparse learning scheme, and the obtained integrated network representation can be projected back into the original space for exploration. The effectiveness of the proposed method is verified on benchmark data sets.

- Chapter 6 addresses the new issues encountered by covariance representation and proposes to use kernel matrix as a generic feature representation. This new representation models more sophisticated feature relationship, is more robust against sample scarcity, and maintains computational efficiency. The significant improvement achieved by this representation is verified on a variety of tasks, including skeleton-based human action recognition, object recognition, and image set classification.

- Chapter 7 concludes this thesis and discusses future work.

Chapter 2

Literature Review

The studies in this thesis focus on two forms of covariance representations, i.e., covariance descriptor and sparse inverse covariance estimation (SICE). This chapter gives an introduction of these two representations and reviews the related literature.

2.1 Covariance Descriptor in Computer Vision

As illustrated in Figure 2.1, given a set of feature vectors $\{\mathbf{x}_i; \mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$, this feature set can be represented by a $d \times d$ covariance matrix $\mathbf{C}$, where $\mathbf{C}$ is defined as $\frac{1}{n-1}\sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top$ and $\boldsymbol{\mu}$ is defined as $\frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$. The matrix $\mathbf{C}$ contains statistical characteristics of this feature set. $\mathbf{C}_{i,j}$ is the covariance between the i^{th} and j^{th} feature variables while the diagonal entry $\mathbf{C}_{i,i}$ is just the variance of i^{th} feature variable.

The past several years have seen an expansion of covariance descriptor in vision applications. Covariance descriptor is firstly proposed in [93] to represent an image region in applications of object detection and texture classification. In that work, pixel locations, color values and derivatives of the intensities are used as feature vectors. Then it is used in [67, 83, 31, 95, 65, 94, 11] for face recognition and person identification or tracking. In these applications, more advanced Gabor features [67] are used

to replace these basic features used in [93]. In comparison, Gabor features could carry more important information and exhibit strong characteristics of spatial locality, scale, and orientation selectivity. Covariance descriptor also becomes popular in action recognition [26, 41, 119, 27, 38], where motion-related feature vectors, e.g. optical flow, locations of 3D joints, are used. Besides, its representative effectiveness has been verified on image set classification as well [108]. Here an image set is a collection of images belonging to the same class but with variation, for example, images of the same object under different views. It is the image set, rather than an individual image, that will be classified. In this case, each image is vectorized into a feature vector and the covariance matrix of these feature vectors is computed to represent this set of images. In addition, covariance descriptor is applied to 3D shape matching/retrieval [35]. In this application, a set of overlapping shape patches are the feature vectors to be represented.

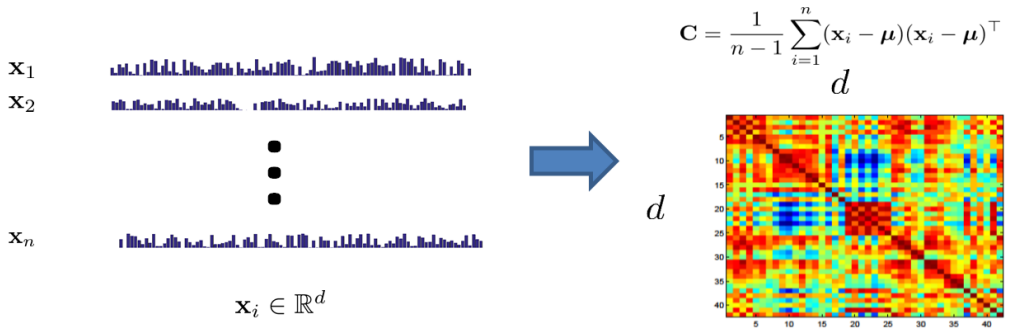


Figure 2.1: The illustration of a covariance descriptor.

2.2 Sparse Inverse Covariance Estimation (SICE)

Another form of covariance representation, sparse inverse covariance estimation (SICE), has been recently employed to model sparse graphs or networks. This thesis focuses on its application in modeling functional brain connectivity networks. A functional brain

connectivity network can be modeled based on the co-varying patterns of resting-state functional Magnetic Resonance Imaging (rs-fMRI) time series across brain regions. Two issues are generally involved: identifying network nodes and inferring the functional connectivity between nodes. The network nodes are often defined as anatomically separated brain regions of interest (ROIs) or alternatively as latent components in some data-driven methods, e.g. independent component analysis [8, 43], and clustering-based methods [12, 98]. Given a set of network nodes, the functional connectivity between two nodes is conventionally measured by the correlation coefficient of time series associated with the two nodes (e.g., the averaged time series from all voxels within a node) [86, 51, 112], and the brain network is then represented by a correlation matrix. However, partial correlation has been argued as a better measure since it measures the correlation of two nodes by regressing out the effects from all other nodes [84]. It often results in a more accurate estimate of network structure in comparison with those correlation-based methods, e.g., covariance matrix. SICE is a principled method for partial correlation estimation, which could produce a stable estimation with the help of the sparsity regularization [22].

Let $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M\}$ be a time series of length M, where $\mathbf{x}_i$ is a d-dimensional vector, corresponding to an observation of d brain nodes. Following the literature of SICE [22, 37], $\mathbf{x}_i$ is assumed to follow a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Each off-diagonal entry of $\boldsymbol{\Sigma}^{-1}$ indicates the partial correlation between two nodes by eliminating the effect of all other nodes. $\boldsymbol{\Sigma}^{-1}_{ij}$ will be zero if nodes i and j are independent of each other when conditioned on the other nodes. In this sense, $\boldsymbol{\Sigma}^{-1}_{ij}$ can be interpreted as the existence and strength of the connectivity between nodes i and j. The estimation of $\mathbf{S} = \boldsymbol{\Sigma}^{-1}$ can be obtained by maximizing the penalized log-likelihood over positive definite matrix $\mathbf{S}$ ($\mathbf{S} \succ 0$) [22, 37]:

$$\mathbf{S}^* = \arg\max_{\mathbf{S} \succ 0} \quad \log\big(\det(\mathbf{S})\big) - \mathrm{tr}(\mathbf{CS}) - \lambda||\mathbf{S}||_1 \tag{2.1}$$

where $\mathbf{C}$ is the sample-based covariance matrix; $\det(\cdot)$, $\mathrm{tr}(\cdot)$ and $||\cdot||_1$ denote the determinant, trace and the sum of the absolute values of the entries of a matrix. $||\mathbf{S}||_1$ imposes sparsity on $\mathbf{S}$ to achieve more reliable estimation by considering the fact that a brain region often has limited direct connections with other brain regions in neurological activities. The tradeoff between the degree of sparsity and the log-likelihood estimation of $\mathbf{S}$ is controlled by the regularization parameter λ. Larger λ makes $\mathbf{S}^*$ sparser. The maximization problem in Eq. (2.1) can be efficiently solved by the off-the-shelf packages, such as SLEP [54]. SICE has been widely used in [37, 62, 123] to model functional brain connectivity networks. For brevity, we call the resulting matrix $\mathbf{S}^*$ "SICE matrix" throughout this thesis.

2.3 Properties of Covariance Representations

A (nonsingular) covariance matrix and SICE matrices belong to the set of symmetric positive-definite (SPD) matrices. The set of SPD matrices with the size of $d \times d$ can be defined as $\mathrm{Sym}_d^+ = \{\mathbf{A} | \mathbf{A} = \mathbf{A}^\top, \forall \mathbf{x} \in \mathbb{R}^d, \mathbf{x} \neq \mathbf{0}, \mathbf{x}^\top \mathbf{A} \mathbf{x} > 0\}$. Given any two SPD matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ and two positive scalars α and β, then $\alpha\mathbf{A}_1 + \beta\mathbf{A}_2$ is also SPD. Therefore, SPD matrices, as illustrated in Fig. 2.2(a), form a convex cone, which is a Riemannian manifold [1] in the Euclidean space [94, 88]. The manifold structure makes it difficult to process and analyze SPD matrices. For example, one critical issue is how to effectively and efficiently measure the similarity between two SPD matrices. As seen in Fig. 2.2(b), methods that respect the geodesic distance rather than Euclidean distance should be used [3]. How to devise such distance measures is still an open issue.

[1] A Riemannian manifold is a real smooth manifold that is differentiable and equipped with a smoothly varying inner product for each tangent space.

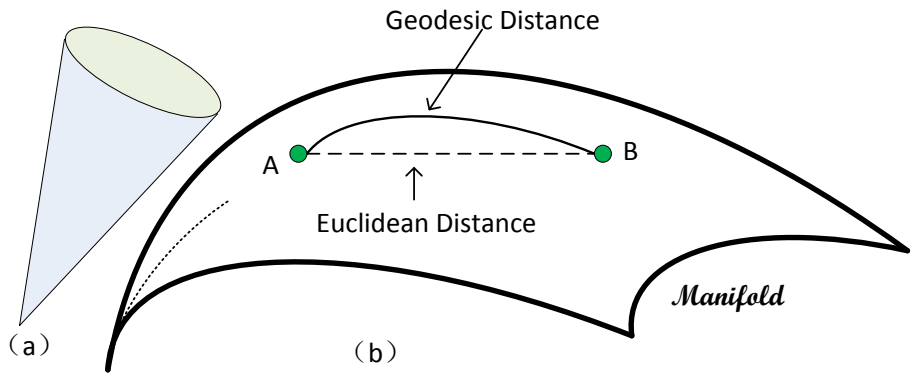


Figure 2.2: The illustration of the Riemannian manifold of SPD matrices. (a) Sym_d^+ forms a closed, self-dual convex cone, which is a Riemannian manifold in the Euclidean space $\mathbb{R}^{d\times d}$ [88]. (b) To measure the distance between two SICE matrices A and B, Euclidean distance is not accurate since it does not consider the special geometry of the manifold structure. Instead, geodesic distance, which is defined as the shortest curve connecting A and B on the manifold, is more accurate.

2.4 Methods for Covariance Representations

In relation to feature representation, covariance descriptor was initially proposed as a region descriptor [93]. Given an image region, a feature vector, $\mathbf{x}$, is extracted from each pixel to describe its location, colour, gradient, filter response, etc. With these feature vectors, covariance matrix is computed to characterize this region. As introduced above, covariance descriptor has been quickly applied to various vision tasks since it was introduced. At the same time, the increasingly wider applications of covariance descriptor promote studies on how to process and improve covariance representations. Existing literature in this aspect can generally be categorized into three approaches.

The first approach improves the quality of covariance representation. Considering that Gabor features could extract more important information, they are used to replace the first- and second-order gradients at each pixel to compute covariance matrix in face

recognition [67]. In object tracking [115], pixels are weighted in computing covariance matrix, and the farther a pixel is from the center of a region, the lower is its weight. Similarly, in action recognition [27], to avoid including background pixels, only the pixels whose temporal gradients are above a threshold (identified as "the pixel related to the movement") are used to compute covariance matrix. The work in [66, 32] considers to model high-order statistics of features. They map feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n$ to a feature space by a kernel function. This results in a potentially *infinite-dimensional* covariance matrix (defined in the kernel-induced feature space) as feature representation. Because these covariance matrices cannot be explicitly computed, a special measure has to be derived to evaluate their similarity. The computational complexity of this measure for a pair of covariance matrices is $\mathcal{O}(n^3)$ due to the need of eigen-decomposition [32], where n is the number of feature vectors. This becomes computationally expensive when n is large. In order to handle high-dimensional covariance descriptor, the work in [30] reduces the dimensions by modeling the mapping from the high-dimensional SPD manifold to the low-dimensional one with an orthonormal projection.

The second approach proposes similarity measures for covariance representations, or general SPD matrices. How to measure the similarity between SPD matrices is a fundamental issue in the analysis of SPD matrices. For example, when a sufficiently good similarity measure is available, a simple k-nearest neighbor classifier will be able to achieve excellent classification performance. Considering that SPD matrices reside on a Riemannian manifold [40], commonly used Euclidean-based measures may not be effective since they do not take the manifold structure into account. To circumvent this problem, affine-invariant Riemannian metric (AIRM) has been proposed in [21, 68] for comparing covariance matrices, and used in [20, 50, 68] for statistical analysis of diffusion tensor imaging (DTI) data. Although improving similarity measurement, AIRM involves matrix inverse and square rooting [3], which could result in high computational cost when the dimensions of SPD matrices are high. In the past decade, measuring the

similarity between SPD matrices effectively and efficiently on the Riemannian manifold has attracted much attention. One attempt is to map the manifold to a Euclidean space [95, 101, 19], i.e. the tangent space at the mean point. However, these approaches suffer from two limitations: i) Mapping the points from manifold to the tangent space or vice-versa is also computationally expensive; ii) More importantly, the tangent space is merely a local approximation of the manifold at the mean point, leading to a suboptimal solution.

To address these drawbacks, kernel based methods have been generalized to handle SPD data residing on a manifold. A point $\mathbf{X}$ on a manifold $\mathcal{M}$ is mapped to a feature vector $\phi(\mathbf{X})$ in some feature space $\mathcal{F}$. The mapping is usually implicitly induced by a kernel function $k : (\mathcal{M}, \mathcal{M}) \rightarrow \mathbb{R}$, which defines the inner product in $\mathcal{F}$, i.e. $k(\mathbf{X}_i, \mathbf{X}_j) = \langle \phi(\mathbf{X}_i), \phi(\mathbf{X}_j) \rangle$. This can bring at least three advantages: i) The computational cost can be reduced by selecting an efficient kernel; ii) The manifold structure can be well incorporated in the embedding; iii) Many Euclidean algorithms, e.g. support vector machines (SVM) [99], can be readily used. Several kernel functions for SPD matrices have been defined via SPD-matrix-based distance functions [40], including Log-Euclidean distance [3, 103], Cholesky distance [16], Power Euclidean distance [16], Stein kernel [88], and Log-Hilbert-Schmidt metric [28] which is a generalization of the Log-Euclidean metric on the Riemannian manifold of positive definite matrices to the infinite-dimensional setting. Riemannian metrics mentioned above and the corresponding kernels are summarized in Table 2.1.

The third approach focuses on classification methods for covariance descriptors. Given any similarity measure mentioned above, a simple k-nearest neighbor (KNN) can be used to perform classification. Moreover, if a valid kernel function is available, SVM can be applied. Also, with the aid of Stein kernel [88], the work in [31] proposes to perform classification based on sparse coding and dictionary learning in reproducing kernel Hilbert spaces. In addition, kernel variants of Linear Discriminant Analysis (LDA) [4]

Table 2.1: Definitions and properties of the metrics on Sym_d^+.

Metric name	Formula (denoted as d)	Does $k = \exp(-\theta \cdot d^2)$ define a valid kernel?	Kernel abbr. in the thesis	Time complexity
AIRM [68]	$\| \log(\mathbf{X}_1^{-\frac{1}{2}} \cdot \mathbf{X}_2 \cdot \mathbf{X}_1^{-\frac{1}{2}}) \|_F$	No	N.A.	$\mathcal{O}(d^3)$
Cholesky [16]	$\| \operatorname{chol}(\mathbf{X}_1) - \operatorname{chol}(\mathbf{X}_2) \|_F$	Yes	CHK	$\mathcal{O}(d^3)$
Euclidean [16]	$\|\mathbf{X}_1 - \mathbf{X}_2\|_F$	Yes	EUK	$\mathcal{O}(d^2)$
Log-Euclidean [3]	$\| \log(\mathbf{X}_1) - \log(\mathbf{X}_2) \|_F$	Yes	LEK	$\mathcal{O}(d^3)$
Power-Euclidean [16]	$\frac{1}{\zeta} \|\mathbf{X}_1^{\zeta} - \mathbf{X}_2^{\zeta}\|_F$	Yes	PEK	$\mathcal{O}(d^3)$
S-Divergence root [88]	$\left[\log\left(\det\left(\frac{\mathbf{X}_1+\mathbf{X}_2}{2}\right)\right) - \frac{1}{2}\log\left(\det(\mathbf{X}_1\mathbf{X}_2)\right)\right]^{\frac{1}{2}}$	Yes ($\theta \in \mathbf{\Theta}$)	SK	$\mathcal{O}(d^{2.373})$

and Partial Least Squares (PLS) [78] are also explored in [109] to train a classifier for covariance descriptors in Log-mapped spaces.

Chapter 3

Learning Discriminative Stein Kernel for Covariance Descriptors

Stein kernel has recently shown promising performance on classifying images represented by covariance descriptors. It evaluates the similarity between two covariance descriptors through their eigenvalues. In this chapter, we argue that directly using the original eigenvalues may be problematic because: i) Eigenvalue estimation becomes biased when the number of samples is inadequate, which may lead to unreliable kernel evaluation; ii) More importantly, eigenvalues only reflect the property of an individual covariance descriptor. They are not necessarily optimal for computing Stein kernel when the goal is to discriminate different classes of covariance descriptors. To address the two issues, we propose a discriminative Stein kernel, in which an extra parameter vector is defined to adjust the eigenvalues of input covariance descriptors. The optimal parameter values are sought by optimizing a proxy of classification performance. To show the generality of the proposed method, three kernel learning criteria that are commonly used in the literature are employed respectively as a proxy. A comprehensive experimental study is conducted on a variety of image classification tasks to compare the proposed discriminative Stein kernel with the original Stein kernel and other methods

for evaluating the similarity between covariance descriptors. The results demonstrate that the discriminative Stein kernel can attain greater discrimination and better align with classification tasks by altering the eigenvalues. This makes it produce higher classification performance than the original Stein kernel and other commonly used methods. Note that although the discriminative Stein kernel proposed in this chapter is mainly designed for covariance descriptors, it can be readily applied to any symmetric positive definite (SPD) matrices. Therefore, "SPD matrices" is used in the following to indicate the generality of the proposed method.

3.1 Motivation

Stein kernel [88] has demonstrated notable improvement on discriminative power in a variety of applications [31, 79, 1, 29]. Stein kernel evaluates the similarity of SPD matrices through their eigenvalues. Although this is theoretically rigorous and elegant, directly using the original eigenvalues may encounter the following two issues in practice.

- *Issue-I.* Some SPD matrices, e.g. covariance descriptors [93], have to be estimated from a set of samples. Nevertheless, covariance matrix estimation is sensitive to the number of samples, and eigenvalue estimation becomes biased when the number of samples is inadequate [36]. Such biasness will affect Stein kernel evaluation.

- *Issue-II.* Even if true eigenvalues could be obtained, when the goal is to discriminate different sets of SPD matrices, computing Stein kernel with these eigenvalues is not necessarily optimal. This is because eigenvalues only reflect the intrinsic property of an individual matrix and the eigenvalues of all the involved matrices have not been collectively manipulated toward greater discrimination.

In this chapter, we propose a *discriminative* Stein kernel (DSK in short) to address the two issues. Specifically, assuming that the eigenvalues of each SPD matrix have been sorted in a given order, a parameter is assigned to each eigenvalue to adjust its magnitude. Treating these parameters as extra parameters of Stein kernel, we automatically learn their optimal values by using the training samples of a classification task. Our work brings forth two advantages: i) Although not restoring the unbiased eigenvalue estimates[1], DSK mitigates the negative impact of the biasness to class discrimination; ii) By adaptively learning the adjustment parameters from training data, it makes Stein kernel better align with specific classification tasks. Both advantages help to boost the classification performance of Stein kernel in practical applications. Three kernel learning criteria, including kernel alignment [81], class separability [106], and the radius margin bound [10], are employed to optimize the adjustment parameters, respectively. The proposed DSK is experimentally compared with the original Stein kernel on a variety of classification tasks. As demonstrated, it not only leads to consistent improvement when the samples for SPD matrix estimation are relatively limited, but also outperforms the original Stein kernel when there are enough samples.

3.2 Proposed Method

3.2.1 Stein kernel

Let $\mathbf{X}$ and $\mathbf{Y}$ be two SPD matrices. Stein kernel is expressed as

$$k(\mathbf{X}, \mathbf{Y}) = \exp\left(-\theta \cdot S\left(\mathbf{X}, \mathbf{Y}\right)\right) \tag{3.1}$$

[1]Note that the proposed method is not designed for (and is not even interested in) restoring the unbiased estimates of eigenvalues. Instead, it aims to achieve better classification when the above two issues are present.

where θ is a tunable positive scalar. $S(\mathbf{X}, \mathbf{Y})$ is called S-Divergence and it is defined as

$$S(\mathbf{X}, \mathbf{Y}) = \log\left(\det\left(\frac{\mathbf{X}+\mathbf{Y}}{2}\right)\right) - \frac{1}{2}\log\left(\det(\mathbf{XY})\right), \tag{3.2}$$

where $\det(\cdot)$ denotes the determinant of a square matrix. The S-Divergence has several desirable properties. For example, i) It is invariant to affine transformations, which is important for computer vision algorithms [29]; ii) The square-root of S-Divergence is proven to be a metric on Sym_d^+ [88]; iii) Stein kernel is guaranteed to be a Mercer kernel when θ varies within the range of $\mathbf{\Theta} = \{\frac{1}{2}, \frac{2}{2}, \frac{3}{2}, \cdots, \frac{(d-1)}{2}\} \cup (\frac{(d-1)}{2}, +\infty)$. Readers are referred to [88] for more details. In general, S-Divergence enjoys higher computational efficiency than AIRM while well maintaining its measurement performance. When compared to other SPD metrics, such as Cholesky distance [16], Euclidean distance [16], Log-Euclidean distance [3, 103], and Power Euclidean distance [16], S-Divergence usually helps to achieve better classification performance [31].

3.2.2 Issues with Stein kernel

Let $\lambda_i(\mathbf{X})$ denote the i^{th} eigenvalue of a SPD matrix $\mathbf{X}$, where $\lambda_i(\mathbf{X})$ is always positive due to the SPD property. Throughout this chapter, we assume that the d eigenvalues have been sorted in descending order. Noting that the determinant of $\mathbf{X}$ equals $\prod_{i=1}^{d}\lambda_i(\mathbf{X})$, the S-Divergence in Eq. (3.2) can be rewritten as

$$S(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{d}\log\lambda_i\left(\frac{\mathbf{X}+\mathbf{Y}}{2}\right) - \frac{1}{2}\sum_{i=1}^{d}\left(\log\lambda_i(\mathbf{X}) + \log\lambda_i(\mathbf{Y})\right). \tag{3.3}$$

We can easily see the important role of eigenvalues in computing $S(\mathbf{X}, \mathbf{Y})$. Inappropriate eigenvalues will affect the precision of S-Divergence and in turn the Stein kernel.

On Issue-I. It has been well realized in the literature that the eigenvalues of sample-based covariance matrix are biased estimates of true eigenvalues [82], especially when

the number of samples is small. Usually, the smaller eigenvalues tend to be underestimated while the larger eigenvalues tend to be overestimated. To better show this case, we conduct a toy experiment. A set of n data is sampled from a 40-dimensional normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where the covariance matrix $\mathbf{\Sigma}$ is defined as $\mathrm{diag}(1, 2, \cdots, 40)$ and the true eigenvalues of $\mathbf{\Sigma}$ are just the diagonal entries. The n data are used to estimate $\mathbf{\Sigma}$ and calculate the eigenvalues. When n is 100, the largest eigenvalue is obtained as 67 while the smallest one is 0.4, which are poor estimates. When n increases to 1000, the largest eigenvalue is still overestimated as 46. From our observation, tens of thousands of samples are required to achieve sufficiently good eigenvalue estimates. Note that the dimensions of 40 are common in practice. A covariance descriptor of 43-dimensional features is used in [31] for face recognition, and it is also used in our experimental study.

On Issue-II. As previously mentioned, even if true eigenvalues could be obtained, a more important issue exists when the goal is to classify different sets of SPD matrices. In specific, a SPD matrix can be expressed as

$$\mathbf{X} = \lambda_1 \mathbf{u}_1 \mathbf{u}_1^\top + \lambda_2 \mathbf{u}_2 \mathbf{u}_2^\top + \cdots + \lambda_d \mathbf{u}_d \mathbf{u}_d^\top,$$

where λ_i and $\mathbf{u}_i$ denote the i^{th} eigenvalue and the corresponding eigenvector. The magnitude of λ_i only reflects the property of this specific SPD matrix, for example, the data variance along the direction of $\mathbf{u}_i$. It does not characterize this matrix from the perspective of discriminating different sets of SPD matrices. We know that, by fixing the d eigenvectors, varying the eigenvalues changes the matrix $\mathbf{X}$. Geometrically, a SPD matrix corresponds to a hyper-ellipsoid in a d-dimensional Euclidean space. This change is analogous to varying the lengths of the axes of the hyper-ellipsoid while maintaining their directions. A question then arises: to make the Stein kernel better prepared for class discrimination, *can we adjust the eigenvalues to make the SPD matrices in the*

same class similar to each other, as much as possible, while maintaining the SPD matrices across classes to be sufficiently different? The "similar" and "different" are defined in the sense of Stein kernel. This idea can also be understood in the other way. An ideal similarity measure shall be more sensitive to inter-class difference and less affected by intra-class variation. Without exception, this shall apply to Stein kernel too.

3.2.3 Proposed discriminative Stein kernel (DSK)

Let $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_d]^\top$ be a vector of adjustment parameters. Let $\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ denote the eigen-decomposition of a SPD matrix, where the columns of $\mathbf{U}$ correspond to the eigenvectors and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_d)$. We use $\boldsymbol{\alpha}$ in two ways for eigenvalue adjustment and define the adjusted $\mathbf{X}$, respectively, as:

$$\tilde{\mathbf{X}}_p = \mathbf{U} \begin{pmatrix} \lambda_1^{\alpha_1} & & & \\ & \lambda_2^{\alpha_2} & & \\ & & \ddots & \\ & & & \lambda_d^{\alpha_d} \end{pmatrix} \mathbf{U}^\top \tag{3.4}$$

$$\text{and } \tilde{\mathbf{X}}_c = \mathbf{U} \begin{pmatrix} \alpha_1\lambda_1 & & & \\ & \alpha_2\lambda_2 & & \\ & & \ddots & \\ & & & \alpha_d\lambda_d \end{pmatrix} \mathbf{U}^\top. \tag{3.5}$$

In the first way, $\boldsymbol{\alpha}$ is used as the *power* of eigenvalues. It can naturally maintain the SPD property because $\lambda_i^{\alpha_i}$ is always positive. In the second way, $\boldsymbol{\alpha}$ is used as the *coefficient* of eigenvalues. It is mathematically simpler but needs to impose the constraint $\alpha_i > 0$ $(i = 1, \cdots, d)$ to maintain the SPD property. The two adjusted matrices are denoted by $\tilde{\mathbf{X}}_p$ and $\tilde{\mathbf{X}}_c$, where p and c are short for "power" and "coefficient". Both ways will be investigated in this chapter.

Given two SPD matrices $\mathbf{X}$ and $\mathbf{Y}$, we define the $\boldsymbol{\alpha}$-adjusted S-Divergence as

$$S_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y}) \triangleq S(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}). \tag{3.6}$$

For the two ways of using $\boldsymbol{\alpha}$, the term $S(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}})$ can be expressed as

$$S(\tilde{\mathbf{X}}_p, \tilde{\mathbf{Y}}_p) = \sum_{i=1}^{d} \log \lambda_i \left(\frac{\tilde{\mathbf{X}}_p + \tilde{\mathbf{Y}}_p}{2} \right) - \frac{1}{2} \sum_{i=1}^{d} \alpha_i \left(\log \lambda_i(\mathbf{X}) + \log \lambda_i(\mathbf{Y}) \right)$$

$$\text{and } S(\tilde{\mathbf{X}}_c, \tilde{\mathbf{Y}}_c) = \sum_{i=1}^{d} \log \lambda_i \left(\frac{\tilde{\mathbf{X}}_c + \tilde{\mathbf{Y}}_c}{2} \right) - \frac{1}{2} \sum_{i=1}^{d} \left(2 \log \alpha_i + \log \lambda_i(\mathbf{X}) + \log \lambda_i(\mathbf{Y}) \right).$$

Based on the above definition, the discriminative Stein kernel (DSK) is proposed as

$$k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y}) = \exp\left(-\theta \cdot S_{\boldsymbol{\alpha}}\left(\mathbf{X}, \mathbf{Y}\right) \right). \tag{3.7}$$

Note that the DSK will remain a Mercer kernel as long as θ varies in the range of $\boldsymbol{\Theta}$ defined in Section 3.2.1, because $k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y})$ can always be viewed as $k(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}})$, the original Stein kernel applied to two adjusted SPD matrices $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$.

Treating $\boldsymbol{\alpha}$ as the kernel parameter of $k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y})$, we resort to kernel learning techniques to find its optimal value. Kernel learning methods have received much attention in the past decade. Many learning criteria such as kernel alignment [81], kernel class separability [106], and radius margin bound [10] have been proposed. In this work, to investigate the generality of the proposed DSK, we employ all the three criteria, respectively, to solve the kernel parameters $\boldsymbol{\alpha}$.

Let $\boldsymbol{\Omega} = \{(\mathbf{X}_i, t_i)\}_{i=1}^{n}$ be a set of n training SPD matrices, each of which represents a sample, e.g., an image to be classified. t_i denotes the class label of the i^{th} sample, where $t_i \in \{1, \cdots, M\}$ with M denoting the number of classes. $\mathbf{K}$ denotes the kernel

matrix computed with DSK on $\mathbf{\Omega}$, with $\mathbf{K}_{ij} = k_{\boldsymbol{\alpha}}(\mathbf{X}_i, \mathbf{X}_j)$. In the following part, three frameworks are developed to learn the optimal value of $\boldsymbol{\alpha}$.

Kernel alignment based framework

Kernel alignment measures the similarity of two kernel functions and can be used to quantify the degree of agreement between a kernel and a given classification task [81]. Kernel alignment possesses several desirable properties, including conceptual simplicity, computational efficiency, concentration of empirical estimate, and theoretical guarantee for generalization performance [81, 110]. Furthermore, kernel alignment is a general-purpose criterion that does not depend on a specific classifier and often leads to simple optimization. Also, it can uniformly handle binary and multi-class classification. Due to these merits, the kernel alignment criterion has been widely used in kernel-related learning tasks [110], including kernel parameter tuning [121], multiple kernel learning [25], spectral kernel learning [57] and feature selection [73].

With the kernel alignment, the optimal $\boldsymbol{\alpha}$ can be obtained through the following optimization problem:

$$\boldsymbol{\alpha}^* = \arg\max_{\boldsymbol{\alpha}\in\mathcal{A}} J(\mathbf{K}, \mathbf{T}) - \lambda\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_0\|_2^2, \tag{3.8}$$

where $\mathbf{T}$ is an $n \times n$ matrix with $\mathbf{T}_{ij} = 1$ if $\mathbf{X}_i$ and $\mathbf{X}_j$ are from the same class and $\mathbf{T}_{ij} = -1$ otherwise. Note that this definition of $\mathbf{T}$ naturally handles multi-class classification. $J(\mathbf{K}, \mathbf{T})$ is defined as the kernel alignment criterion:

$$J(\mathbf{K}, \mathbf{T}) = \frac{\langle \mathbf{T}, \mathbf{K} \rangle_F}{\sqrt{\langle \mathbf{T}, \mathbf{T} \rangle_F \langle \mathbf{K}, \mathbf{K} \rangle_F}} \tag{3.9}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product between two matrices. $J(\mathbf{K}, \mathbf{T})$ measures the degree of agreement between $\mathbf{K}$ and $\mathbf{T}$, where $\mathbf{T}$ is regarded as the ideal kernel of a learning task. The $\boldsymbol{\alpha}_0$ is a priori estimate of $\boldsymbol{\alpha}$, and $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_0\|_2^2$ is the reg-

ularizer which constrains $\boldsymbol{\alpha}$ to be around $\boldsymbol{\alpha}_0$ to avoid overfitting. We can simply set $\boldsymbol{\alpha}_0 = [1, \cdots, 1]^\top$, which corresponds to the original Stein kernel. λ is the regularization parameter to be selected via cross-validation. $\mathcal{A}$ denotes the domain of $\boldsymbol{\alpha}$: when $\boldsymbol{\alpha}$ is used as a power, $\mathcal{A}$ denotes a Euclidean space $\mathbb{R}^d$; when $\boldsymbol{\alpha}$ is used as a coefficient, $\mathcal{A}$ is constrained to $\mathbb{R}^d_+$.

$J(\mathbf{K}, \mathbf{T})$ is differentiable with respect to $\mathbf{K}$ and $\boldsymbol{\alpha}$:

$$\frac{\partial J(\mathbf{K}, \mathbf{T})}{\partial \alpha_z} = \frac{\langle \mathbf{T}, \frac{\partial \mathbf{K}}{\partial \alpha_z} \rangle_F}{\sqrt{\langle \mathbf{T}, \mathbf{T} \rangle_F \langle \mathbf{K}, \mathbf{K} \rangle_F}} - \frac{\langle \mathbf{T}, \mathbf{K} \rangle_F \langle \mathbf{K}, \frac{\partial \mathbf{K}}{\partial \alpha_z} \rangle_F}{\sqrt{\langle \mathbf{T}, \mathbf{T} \rangle_F} \langle \mathbf{K}, \mathbf{K} \rangle_F^{3/2}} \tag{3.10}$$

where α_z denotes the z^{th} parameter of $\boldsymbol{\alpha}$ and the entry of $\frac{\partial \mathbf{K}}{\partial \alpha_z}$ is $\frac{\partial k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y})}{\partial \alpha_z}$. Based on Eq. (3.7), it can be calculated as $\frac{\partial k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y})}{\partial \alpha_z} = \frac{\theta k_{\boldsymbol{\alpha}}(\mathbf{X}, \mathbf{Y})}{2} \operatorname{tr}\left[\tilde{\mathbf{X}}^{-1} \left(\frac{\partial \tilde{\mathbf{X}}}{\partial \alpha_z} \right) + \tilde{\mathbf{Y}}^{-1} \left(\frac{\partial \tilde{\mathbf{Y}}}{\partial \alpha_z} \right) - \left(\frac{\tilde{\mathbf{X}} + \tilde{\mathbf{Y}}}{2} \right)^{-1} \left(\frac{\partial \tilde{\mathbf{X}}}{\partial \alpha_z} + \frac{\partial \tilde{\mathbf{Y}}}{\partial \alpha_z} \right) \right]$, where $\operatorname{tr}(\cdot)$ denotes the trace of a matrix. Therefore, any gradient-based optimization technique can be applied to solve the optimization problem in Eq. (3.8).

On choosing θ

As seen in Eq. (3.7), there is a kernel parameter θ inherited from the original Stein kernel. Note that θ and $\boldsymbol{\alpha}$ play different roles in the proposed kernel and cannot be replaced with each other. The value of θ needs to be appropriately chosen because it impacts the kernel value and in turn the optimization of $\boldsymbol{\alpha}$. A commonly used way to tune θ is k-fold cross-validation. In this chapter, to better align with the kernel alignment criterion, we also tune θ by maximizing the kernel alignment and do this before adjusting $\boldsymbol{\alpha}$,

$$\theta^* = \arg\max_{\theta \in \boldsymbol{\Theta}} J(\mathbf{K}|_{\boldsymbol{\alpha}=\mathbf{1}}, \mathbf{T}). \tag{3.11}$$

where $\mathbf{1}$ is a d-dimensional vector with all entries equal to 1 and $\mathbf{K}|_{\boldsymbol{\alpha}=\mathbf{1}}$ denotes the kernel matrix computed by the original Stein kernel without $\boldsymbol{\alpha}$-adjustment. Through

this optimization, we find a reasonably good θ and then optimize $\boldsymbol{\alpha}$ on top of it. The maximization problem in Eq. (3.11) can be conveniently solved by choosing θ in the range of $\boldsymbol{\Theta} = \left\{\frac{1}{2}, \frac{2}{2}, \frac{3}{2}, \cdots, \frac{d-1}{2}\right\} \cup \left(\frac{d-1}{2}, +\infty\right)$. θ is not optimized jointly with $\boldsymbol{\alpha}$ since the noncontinuous range of θ could complicate the gradient-based optimization. As will be shown in the experimental study, optimizing θ and $\boldsymbol{\alpha}$ sequentially can already lead to promising results.

After obtaining θ^* and $\boldsymbol{\alpha}^*$, the proposed DSK will be applied to both training and test data for classification, with certain classifiers such as k-nearest neighbor (k-NN) or SVM. Note that for a given classification task, the optimization of θ and $\boldsymbol{\alpha}$ only needs to be conducted *once* with training data. After that, they are used as fixed parameters to compute the Stein kernel for each pair of SPD matrices. The DSK with kernel alignment criterion is outlined in Algorithm 4.

Algorithm 1 Proposed discriminative Stein kernel learning with the kernel alignment criterion

Input: A training sample set $\boldsymbol{\Omega} = \{(\mathbf{X}_i, t_i)\}_{i=1}^n$, $\boldsymbol{\alpha}_0$ and λ.
Output: θ^*, $\boldsymbol{\alpha}^*$;

1: Find $\theta^* = \arg\max_{\theta \in \boldsymbol{\Theta}} J(\mathbf{K}|_{\boldsymbol{\alpha}=\mathbf{1}}, \mathbf{T})$ first to obtain θ^*;
2: Learn $\boldsymbol{\alpha}^* = \arg\max_{\boldsymbol{\alpha} \in \mathcal{A}} J(\mathbf{K}|_{\theta=\theta^*}, \mathbf{T}) - \lambda \|\boldsymbol{\alpha} - \boldsymbol{\alpha}_0\|_2^2$;
3: **return** θ^*, $\boldsymbol{\alpha}^*$;

Class separability based framework

Class separability is another commonly used criterion for model and feature selection [107, 106, 116]. Recall that the training sample set is defined as $\boldsymbol{\Omega} = \{(\mathbf{X}_i, t_i)\}_{i=1}^n$, where $t_i \in \{1, \cdots, M\}$. Let $\boldsymbol{\Omega}_i$ be the set of training samples from the i^{th} class, with n_i denoting the size of $\boldsymbol{\Omega}_i$. $\mathbf{K}_{\boldsymbol{\Omega}', \boldsymbol{\Omega}''}$ denotes a kernel matrix computed over two training subsets $\boldsymbol{\Omega}'$ and $\boldsymbol{\Omega}''$, where $\{\mathbf{K}_{\boldsymbol{\Omega}', \boldsymbol{\Omega}''}\}_{ij} = k(\mathbf{X}_i, \mathbf{X}_j) = \langle \phi(\mathbf{X}_i), \phi(\mathbf{X}_j) \rangle$ with $\mathbf{X}_i \in \boldsymbol{\Omega}'$ and $\mathbf{X}_j \in \boldsymbol{\Omega}''$. The class separability in the feature space $\mathcal{F}$ induced by a kernel k can

be defined as

$$J = \frac{\mathrm{tr}(\mathbf{S}_B)}{\mathrm{tr}(\mathbf{S}_W)}, \tag{3.12}$$

where $\mathrm{tr}(\cdot)$ is the trace of a matrix, and $\mathbf{S}_B$ and $\mathbf{S}_W$ are the *between-class scatter matrix* and the *within-class scatter matrix*, respectively. Let $\mathbf{m}$ and $\mathbf{m}_i$ denote the total sample mean and the i^{th} class mean. They can be expressed as $\mathbf{m} = \frac{1}{n}\sum_{\mathbf{X}_i \in \mathbf{\Omega}} \phi(\mathbf{X}_i)$ and $\mathbf{m}_i = \frac{1}{n_i}\sum_{\mathbf{X}_j \in \mathbf{\Omega}_i} \phi(\mathbf{X}_j)$.

$\mathrm{tr}(\mathbf{S}_B)$ and $\mathrm{tr}(\mathbf{S}_W)$ can be expressed as:

$$\mathrm{tr}(\mathbf{S}_B) = \mathrm{tr}\left[\sum_{i=1}^{M} n_i \left(\mathbf{m}_i - \mathbf{m}\right)\left(\mathbf{m}_i - \mathbf{m}\right)^\top\right] = \sum_{i=1}^{M} \frac{\mathbf{1}^\top \mathbf{K}_{\mathbf{\Omega}_i,\mathbf{\Omega}_i}\mathbf{1}}{n_i} - \frac{\mathbf{1}^\top \mathbf{K}_{\mathbf{\Omega},\mathbf{\Omega}}\mathbf{1}}{n}, \tag{3.13}$$

and

$$\begin{aligned}\mathrm{tr}(\mathbf{S}_W) &= \mathrm{tr}\left[\sum_{i=1}^{M}\sum_{j=1}^{n_i}\Big(\phi\left(\mathbf{X}_{ij}\right) - \mathbf{m}_i\Big)\Big(\phi\left(\mathbf{X}_{ij}\right) - \mathbf{m}_i\Big)^\top\right] \\ &= \mathrm{tr}\left(\mathbf{K}_{\mathbf{\Omega},\mathbf{\Omega}}\right) - \sum_{i=1}^{M}\frac{\mathbf{1}^\top \mathbf{K}_{\mathbf{\Omega}_i,\mathbf{\Omega}_i}\mathbf{1}}{n_i}\end{aligned} \tag{3.14}$$

where $\mathbf{1} = [1, 1, \cdots, 1]^\top$. The derivatives of $\mathrm{tr}(\mathbf{S}_B)$ and $\mathrm{tr}(\mathbf{S}_W)$ with respect to α_z can be shown as:

$$\frac{\partial\,\mathrm{tr}(\mathbf{S}_B)}{\partial\alpha_z} = \sum_{i=1}^{M}\frac{\mathbf{1}^\top\frac{\partial \mathbf{K}_{\mathbf{\Omega}_i,\mathbf{\Omega}_i}}{\partial\alpha_z}\mathbf{1}}{n_i} - \frac{\mathbf{1}^\top\frac{\partial \mathbf{K}_{\mathbf{\Omega},\mathbf{\Omega}}}{\partial\alpha_z}\mathbf{1}}{n}, \tag{3.15}$$

$$\text{and}\ \ \frac{\partial\,\mathrm{tr}(\mathbf{S}_W)}{\partial\alpha_z} = \mathrm{tr}\left(\frac{\partial \mathbf{K}_{\mathbf{\Omega},\mathbf{\Omega}}}{\partial\alpha_z}\right) - \sum_{i=1}^{M}\frac{\mathbf{1}^\top\frac{\partial \mathbf{K}_{\mathbf{\Omega}_i,\mathbf{\Omega}_i}}{\partial\alpha_z}\mathbf{1}}{n_i} \tag{3.16}$$

The class separability can reflect the goodness of a kernel function with respect to a given task. The DSK learning procedure outlined in Algorithm 4 can be fully taken advantage to optimize the parameter $\boldsymbol{\alpha}$ when class separability measure is used. The only modification is to replace the definition of J with Eq. (3.12).

Radius margin bound based framework

Radius margin bound is an upper bound on the number of classification errors in a leave-one-out (LOO) procedure of a hard margin binary SVM [10, 44]. This bound can be extended to L_2-norm soft margin SVM with a slightly modified kernel. It has been widely used for parameter tuning [10] and model selection [107]. We first consider a binary classification task and then extend the result to the multi-class case. Let $\mathbf{\Omega}' \cup \mathbf{\Omega}''$ be a training set of l samples, and without loss of generality, the samples are labeled by $t \in \{-1, 1\}$. With a given kernel function k, the optimization problem of SVM with L_2-norm soft margin can be expressed as

$$\begin{aligned} \frac{1}{2}\|\mathbf{w}\|^2 = \quad & \max_{\boldsymbol{\eta}\in\mathbb{R}^l} \Big[\sum_{i=1}^{l} \eta_i - \frac{1}{2} \sum_{i,j=1}^{l} \eta_i \eta_j t_i t_j \tilde{k}(\mathbf{X}_i, \mathbf{X}_j) \Big] \\ \text{subject to:} \quad & \sum_{i=1}^{l} \eta_i t_i = 0;\ \eta_i \geq 0\ (i = 1, 2, \ldots, l) \end{aligned} \tag{3.17}$$

where $\tilde{k}(\mathbf{X}_i, \mathbf{X}_j) = k(\mathbf{X}_i, \mathbf{X}_j) + \frac{1}{C}\delta_{ij}$; C is the regularization parameter; $\delta_{ij} = 1$ if $i = j$, and 0 otherwise; and $\mathbf{w}$ is the normal vector of the optimal separating hyperplane of SVM. Tuning the parameters in $\tilde{k}$ can be achieved by minimizing an estimate of the LOO errors. It is shown in [99] that the following *radius margin bound* holds:

$$E(\text{LOO}) \leq 4 \cdot \frac{R^2}{\gamma^2} = 4R^2\|\mathbf{w}\|^2, \tag{3.18}$$

where $E(\text{LOO})$ denotes the number of LOO errors performed on the l training samples in $\mathbf{\Omega}' \cup \mathbf{\Omega}''$; R is the radius of the smallest sphere enclosing all the l training samples; and γ denotes the margin with respect to the optimal separating hyperplane and equals

$1/\|\mathbf{w}\|$. R^2 can be obtained by the following optimizing problem,

$$\begin{aligned} R^2 = \max_{\beta \in \mathbb{R}^l} \Big[\sum_{i=1}^{l} \beta_i \tilde{k}(\mathbf{X}_i, \mathbf{X}_i) - \sum_{i,j=1}^{l} \beta_i \beta_j \tilde{k}(\mathbf{X}_i, \mathbf{X}_j) \Big] \\ \text{subject to: } \sum_{i=1}^{l} \beta_i = 1; \;\; \beta_i \geq 0 \;\; (i = 1, 2, \ldots, l). \end{aligned} \tag{3.19}$$

Note that both R and $\mathbf{w}$ are the function of the kernel $\tilde{k}$. We set the kernel function k as $k_{\boldsymbol{\alpha}}$ defined in Eq. (3.7). The model parameters in $\tilde{k}$, i.e. $\{\theta, \boldsymbol{\alpha}, C\}$, can be optimized by minimizing $R^2\|\mathbf{w}\|^2$ on the training set. As previous, we can first choose a reasonably good θ^* by optimizing Eq. (3.20) with respect to θ and C while fixing $\boldsymbol{\alpha}$ as $\mathbf{1}$.

$$\{\theta^*, C^*\} = \arg \min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}, C>0 | \boldsymbol{\alpha} = \mathbf{1}} \quad R^2 \|\mathbf{w}\|^2. \tag{3.20}$$

Once θ^* is obtained, $\{\boldsymbol{\alpha}, C\}$, denoted by $\boldsymbol{\upsilon}$, can then be jointly optimized as follows:

$$\boldsymbol{\upsilon}^* = \arg \min_{\boldsymbol{\upsilon} \in \boldsymbol{\Upsilon} | \theta = \theta^*} \quad R^2 \|\mathbf{w}\|^2, \tag{3.21}$$

where $\boldsymbol{\Upsilon} = \{\boldsymbol{\alpha}, C | \boldsymbol{\alpha} \in \mathcal{A}; C > 0\}$. Let υ_z be the z^{th} parameter of $\boldsymbol{\upsilon}$. The derivative of $R^2\|\mathbf{w}\|^2$ with respect to υ_z can be shown as:

$$\frac{\partial (R^2 \|\mathbf{w}\|^2)}{\partial \upsilon_z} = \|\mathbf{w}\|^2 \frac{\partial R^2}{\partial \upsilon_z} + R^2 \frac{\partial \|\mathbf{w}\|^2}{\partial \upsilon_z}, \tag{3.22}$$

$$\text{where} \qquad \frac{\partial \|\mathbf{w}\|^2}{\partial \upsilon_z} = - \sum_{i,j=1}^{l} \eta_i^* \eta_j^* t_i t_j \frac{\partial \tilde{k}(\mathbf{X}_i, \mathbf{X}_j)}{\partial \upsilon_z} \tag{3.23}$$

$$\text{and } \frac{\partial R^2}{\partial \upsilon_z} = \sum_{i=1}^{l} \beta_i^* \frac{\partial \tilde{k}(\mathbf{X}_i, \mathbf{X}_i)}{\partial \upsilon_z} - \sum_{i,j=1}^{l} \beta_i^* \beta_j^* \frac{\partial \tilde{k}(\mathbf{X}_i, \mathbf{X}_j)}{\partial \upsilon_z} \tag{3.24}$$

where η_i^* and β_i^* denote the optimal solutions of Eq. (3.17) and (3.19), respectively.

For multi-class classification tasks, we optimize $\boldsymbol{\upsilon}$ by a pairwise combination of

the radius margin bounds of binary SVM classifiers [107]. Specifically, an M-class classification task can be split into $M(M-1)/2$ pairwise classification tasks by using the one-vs-one strategy. For any class pair (i,j) with $1 \leq i < j \leq M$, a binary SVM classifier, denoted by SVM$_{ij}$, can be trained using the samples from classes i and j. The corresponding radius margin bound, denoted by $R_{ij}^2\|\mathbf{w}_{ij}\|^2$, can be calculated by Eq. (3.17) and (3.19). As shown in [107], the LOO error of an M-class SVM classifier can be upper bounded by the combination of the radius margin bounds of $M(M-1)/2$ pairwise binary SVM classifiers. The combination is defined as:

$$J = \sum_{1 \leq i < j \leq M} R_{ij}^2 \|\mathbf{w}_{ij}\|^2. \tag{3.25}$$

As previous, a reasonably good θ^* can be firstly chosen by

$$\{\theta^*, C^*\} = \arg \min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}, C>0 | \boldsymbol{\alpha}=\mathbf{1}} \quad J \ . \tag{3.26}$$

The optimal kernel parameter $\boldsymbol{\upsilon}^*$ for an M-class SVM classifier can then be obtained by

$$\boldsymbol{\upsilon}^* = \arg \min_{\boldsymbol{\upsilon} \in \boldsymbol{\Upsilon} | \theta=\theta^*} \quad J \ . \tag{3.27}$$

The derivative of J with respect to υ_z is given by

$$\frac{\partial J}{\partial \upsilon_z} = \sum_{1 \leq i < j \leq M} \left(\|\mathbf{w}_{ij}\|^2 \frac{\partial R_{ij}^2}{\partial \upsilon_z} + R_{ij}^2 \frac{\partial \|\mathbf{w}_{ij}\|^2}{\partial \upsilon_z} \right), \tag{3.28}$$

where $\frac{\partial R_{ij}^2}{\partial \upsilon_z}$ and $\frac{\partial \|\mathbf{w}_{ij}\|^2}{\partial \upsilon_z}$ for SVM$_{ij}$ can be obtained by following Eq. (3.24) and (3.23). J can be optimized by using gradient-based methods.

In classification, the label of a test sample $\mathbf{X}$ can be assigned using the max-wins

Algorithm 2 Proposed discriminative Stein kernel learning with the radius margin bound or trace margin criterion

Input: A training set $\mathbf{\Omega} = \{(\mathbf{X}_i, t_i)\}_{i=1}^{n}$, stopping criteria: i) The total number of iterations T; ii) A small positive value τ.

Output: θ^*; $\boldsymbol{\upsilon}^* = \{\boldsymbol{\alpha}^*, C^*\}$.

```
Find {θ*, C*} by solving Eq. (3.26);
for t = 1 : T do
    Solve ‖w_ij‖² in Eq. (3.25) according to Eq. (3.17);
    Solve R²_ij in Eq. (3.25) according to Eq. (3.19) or approximate it with tr(S_T);
    Update υ by a gradient-based method via Eq. (3.28);
    if |J_{t+1} − J_t| ≤ τ J_t  (J is defined in Eq. (3.25)) then
        Break;
    end if
end for
return θ*, υ*;
```

classification rule by

$$t(\mathbf{X}) = \arg \max_{i=1,\ldots,M} \left(\sum_{j=1, j\neq i}^{M} \operatorname{sign}(s_{ij}) \right), \tag{3.29}$$

where s_{ij} is the decision score of the binary classifier SVM$_{ij}$, and it is computed as

$$s_{ij} = \sum_{z=1}^{n_i+n_j} \eta_z^* t_z k(\mathbf{X}, \mathbf{X}_\mathbf{z}) + b_{ij}^*, \quad \mathbf{X}_\mathbf{z} \in \mathbf{\Omega}_i \cup \mathbf{\Omega}_j \ .$$

We also consider a variant of the radius margin bound by replacing R_{ij}^2 with $\operatorname{tr}(\mathbf{S}_T)$, where $\operatorname{tr}(\mathbf{S}_T) = \operatorname{tr}(\mathbf{S}_B) + \operatorname{tr}(\mathbf{S}_W)$. It can be calculated by using Eq. (3.13) and (3.14) on $\mathbf{\Omega}_i \cup \mathbf{\Omega}_j$. As revealed in [106], R_{ij}^2 is closely related to $\operatorname{tr}(\mathbf{S}_T)$ and both of them measure the scattering of samples in a feature space $\mathcal{F}$. Replacing R_{ij}^2 with $\operatorname{tr}(\mathbf{S}_T)$ can often result in more stable optimization [55], and solving the quadratic programming (QP) problem in Eq. (3.19) can also be avoided. In the experimental study, both methods, named *radius margin bound* and *trace margin criterion*, are implemented to investigate the performance of DSK. The overall procedure is outlined in Algorithm 2.

3.3 Computational Issue

As will be shown in the experiments, the optimization problems of the proposed DSK can be efficiently solved and often converge in a few iterations. Two stopping criteria are used: i) Optimization will be terminated when the difference of the objective values at two successive iterations is below a predefined threshold τ; ii) The optimization will also be stopped when the number of iterations exceeds a predefined threshold T. We set $\tau = 10^{-5}$ and $T = 100$ in our experiment.

The kernel alignment and class separability criteria (defined in Eq. (3.9) and (3.12), respectively) can be quickly computed with a given kernel matrix. The major computational bottleneck is at the computation of the kernel matrix $\mathbf{K}$, which is repeatedly evaluated for various $\boldsymbol{\alpha}$ values. For a training set of n SPD matrices, the time complexity of optimizing $\boldsymbol{\alpha}$ using the kernel alignment or the class separability criterion is $\mathcal{O}(mn^2d^3)$, where m is the total number of objective function evaluations and $\mathcal{O}(d^3)$ is the complexity of eigen-decomposition of a $d \times d$ SPD matrix. Note that the complexity $\mathcal{O}(n^2)$ is common for all kernel parameter learning algorithms. Once $\boldsymbol{\alpha}^*$ is obtained, computing the proposed DSK on a pair of SPD matrices is $\mathcal{O}(d^3)$. It is comparable to the original Stein kernel, which has the complexity of $\mathcal{O}(d^{2.373})$ via computing the determinant instead of conducting an eigen-decomposition.

In the framework using the radius margin bound, $\boldsymbol{\upsilon}$ (defined before Eq. (3.21)) is optimized by a combination of $M(M-1)/2$ binary SVM classifiers. Once $\boldsymbol{\upsilon}$ is updated, $\mathcal{O}\left((n_i+n_j)^2 d^3\right)$ is required to calculate the kernel matrix for SVM$_{ij}$, and a QP problem involving $n_i + n_j$ samples needs to be solved to update R_{ij}^2 or $\|\mathbf{w}_{ij}\|^2$. Let $\mathcal{O}(\text{QP}(n))$ denote the computational complexity to solve a QP problem of n samples. The overall complexity to optimize $\boldsymbol{\alpha}$ in the framework of radius margin bound will be $\mathcal{O}\left(m\sum_{1\le i<j\le M}\left[(n_i+n_j)^2 d^3 + 2\text{QP}(n_i+n_j)\right]\right)$. In addition, one QP optimization can be avoided when the trace margin criterion is used, leading to a reduced complexity

of $\mathcal{O}\Big(m\sum_{1\leq i<j\leq M}[(n_i+n_j)^2\,d^3+\text{QP}(n_i+n_j)]\Big)$. At last, the computational complexity of classifying a test sample by Eq. (3.29) is $\mathcal{O}\left(\sum_{1\leq i<j\leq M}\left[(n_i+n_j)\,d^3\right]\right)$.

3.4 Experimental Result

In this experiment, we compare the proposed discriminative Stein kernel (DSK) with the original Stein kernel (SK) on various image classification tasks. The other metrics listed in Table 2.1 will also be compared. Source code implementing the proposed method is publicly available[2].

Three data sets are used. Two of them are the Brodatz data set [74] for texture classification and the FERET data set [69] for face recognition, which have been used in the literature [31]. The third one is the ETH-80 [49] data set widely used for visual object categorization [40, 103]. In Brodatz, FERET and ETH-80 data sets, images are represented by covariance descriptors. The details of these datasets will be introduced in the following subsections.

We employ both k-NN and SVM as the classifiers. For the kernel alignment and class separability frameworks, k-NN is used with the DSK as the similarity measure, since it does not involve any other (except k) algorithmic parameter. This allows the comparison to directly reflect the change from SK to DSK. For the radius margin bound framework, SVM classifier is used since it is inherently related to this bound.

In this experiment, the DSK obtained by the kernel alignment and class separability are called DSK-KA and DSK-CS. Also, DSK-RM indicates the DSK obtained by the radius margin bound, while DSK-TM denotes the DSK obtained by trace margin criterion. Subscripts p or c is used to indicate whether $\boldsymbol{\alpha}$ acts as the power or the coefficient of eigenvalues. All the names are summarized in Table 3.1.

All parameters, including the k of k-NN, the regularization parameter of SVM, λ in

[2] `https://github.com/seuzjj/DSK.git`

Table 3.1: The name of DSK under different learning criteria

$\boldsymbol{\alpha}$ as	Kernel alignment	Class separability	Radius margin bound	Trace margin criterion
power	DSK-KA$_p$	DSK-CS$_p$	DSK-RM$_p$	DSK-TM$_p$
coefficient	DSK-KA$_c$	DSK-CS$_c$	DSK-RM$_c$	DSK-TM$_c$

Eq. (3.8), θ in all the kernels in Table 2.1, and the power order ζ in the Power-Euclidean metric are chosen via multi-fold cross-validation on the training set.

In the experiments, we perform binary classification on the Brodatz and rs-fMRI data sets and multi-class classification on the Brodatz, FERET and ETH-80 data sets. For each experiment on the Brodatz, FERET and ETH-80 data sets, the data are randomly split into two equal-sized subsets for training and test. The procedure is repeated 20 times for each task to obtain stable statistics. Besides classification accuracy, the p-value obtained by paired Student's t-test between DSK and SK will be used to evaluate the significance of improvement (p-value ≤ 0.05 is used).

3.4.1 Results on the Brodatz texture data set

The Brodatz data set contains 112 images, each of which represents one class of texture. Following the literature [31], a set of sub-regions are cropped from each image as the samples of the corresponding texture class. The covariance descriptor [93] is used to describe a texture sample (sub-region) as follows.

(1) Each original texture image is scaled to a uniform size of 256×256;

(2) Each image is then split into 64 non-overlapping sub-regions of size 32×32. Each image is considered as a texture class and its sub-regions are used as the samples of this class;

(3) A five-dimensional feature vector $\phi(x, y) = [I(x, y), |\frac{\partial I}{\partial x}|, |\frac{\partial I}{\partial y}|, |\frac{\partial^2 I}{\partial x^2}|, |\frac{\partial^2 I}{\partial y^2}|]$ is extracted at pixel (x, y) in each sub-region, where $I(x, y)$ denotes the intensity value

at that pixel;

(4) Each sub-region is represented by a 5×5 covariance matrix estimated by using all ($1024 = 32 \times 32$) the features vectors obtained from that sub-region.

For the experiment of binary classification, we first run pairwise classification between the 112 classes by using SK with the k-NN classifier. The obtained classification accuracies are sorted in ascending order. The top 15 pairs with the lowest accuracies, which represent the most difficult classification tasks, are selected. The rest of these pairs are not included because SK has been able to obtain almost 100% accuracy on them. As we observed in the experiment, DSK achieves equally excellent performance as SK on these pairs. The texture images in each pair are visually similar to each other and it is challenging to classify them. In short, we obtain 15 pairs of classes. Each class consists of 64 samples, and each sample is represented by a 5×5 covariance descriptor.

Table 3.2: Comparison of Classification accuracy (in percentage) on each of the 15 most difficult pairs from Brodatz texture data set.

Index	1	2	3	4	5	6	7	8
SK	62.50	67.19	68.75	75.00	75.78	75.79	76.56	77.34
DSK-KA$_p$	**70.31**	**73.44**	**75.00**	**81.25**	**76.56**	**79.69**	**82.81**	**79.69**
Index	9	10	11	12	13	14	15	**Average**
SK	78.13	79.69	80.47	81.25	82.04	83.59	85.94	76.67
DSK-KA$_p$	**84.37**	**84.39**	**84.38**	**84.38**	**84.35**	**84.42**	**87.50**	**80.85**

The average classification accuracies on the 15 binary classification tasks are compared in Table 3.3. The left half of the table shows the results when the k-NN is the classifier, while the right half is for the SVM classifier. As seen from the left half, SK achieves the best performance (76.67%) under the column of "Competing methods". At the same time, the proposed DSK-KA and DSK-CS consistently achieve better performance than SK, when the parameter $\boldsymbol{\alpha}$ is used as the power or coefficient. Especially, DSK-KA$_p$ achieves the best performance (80.85%), obtaining an improvement of above

Table 3.3: Comparison of classification accuracy (in percentage) averaged on 15 most difficult pairs from Brodatz texture data set.

k-NN				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-KA$_p$	DSK-KA$_c$
74.67 ± 4.48	73.78 ± 4.68	73.59 ± 6.31	**80.85** ± 4.96	78.33 ± 4.79
SK	LEK	PEK	DSK-CS$_p$	DSK-CS$_c$
76.67 ± 5.30	74.67 ± 4.17	74.70 ± 4.16	79.69 ± 3.74	77.29 ± 4.25
SVM				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-RM$_p$	DSK-RM$_c$
N.A. N.A.	77.87 ± 5.89	76.04 ± 5.93	**80.57** ± 5.95	79.16 ± 6.34
SK	LEK	PEK	DSK-TM$_p$	DSK-TM$_c$
78.38 ± 6.21	78.89 ± 4.69	78.06 ± 5.90	80.47 ± 6.55	79.08 ± 6.93

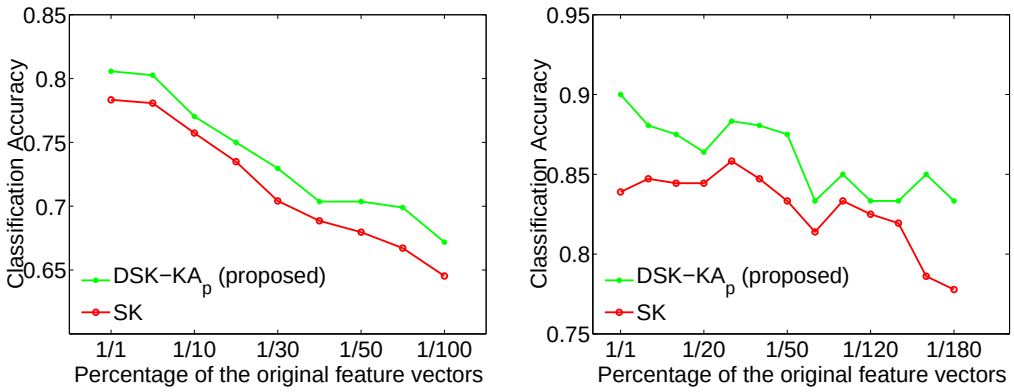
Table 3.4: Comparison of Classification accuracy (in percentage) averaged on 112-class classification on Brodatz texture data set.

k-NN				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-KA$_p$	DSK-KA$_c$
74.93 ± 0.61	72.19 ± 0.55	64.72 ± 0.78	78.12 ± 0.75	77.50 ± 0.69
SK	LEK	PEK	DSK-CS$_p$	DSK-CS$_c$
76.80 ± 0.84	74.38 ± 0.62	72.02 ± 0.65	**78.43** ± 0.59	77.80 ± 0.81
SVM				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-RM$_p$	DSK-RM$_c$
N.A. N.A.	77.39 ± 0.82	74.27 ± 1.26	**83.40** ± 0.58	82.94 ± 0.71
SK	LEK	PEK	DSK-TM$_p$	DSK-TM$_c$
78.01 ± 0.43	78.22 ± 1.00	76.88 ± 0.84	80.41 ± 0.47	80.10 ± 0.53

4 percentage points over SK and more than 6 percentage points over the other methods. The relatively large standard deviation in Table 3.3 is mainly due to the variation of accuracy rates of the 15 tasks. Actually, the improvement of DSK over SK is statistically significant because the p-value between DSK-KA$_p$ and SK is as small as 5.4×10^{-6}. To better show the difference between DSK-KA$_p$ and SK, their performance on each of the 15 pairs is reported in Table 3.2. As seen, DSK-KA$_p$ consistently outperforms SK on each task. The right half of Table 3.3 compares the DSK obtained by the radius margin bound with the other kernel methods, by using SVM as the classifier. As seen, the four variants of DSK in this case, DSK-RM$_p$, DSK-RM$_c$, DSK-TM$_p$ and DSK-TM$_c$, outperform the other kernel methods, including SK. Note that AIRM does not admit a valid kernel [40] and is not included in the comparison.

We also test DSK on multi-class classification involving all the 112 classes of Brodatz data. As seen from Table 3.4, all the DSK methods outperform SK and other methods in comparison. Specifically, as indicated in the left part of Table 3.4, SK has the highest classification accuracy (76.80%) among all these existing methods when k-NN is used as the classifier. Meanwhile, compared with SK, DSK-CS$_p$ achieves a further improvement of 1.6 percentage points with p-value of 0.0018. When SVM is used in the right part of Table 3.4, DSK-RM$_p$ boosts the performance of SK from 78.01% to 83.40%, obtaining an improvement of 5.39 percentage points.

In addition, we investigate the performance of DSK with respect to the number of samples used to estimate the covariance descriptors. Recall that a 5×5 covariance descriptor is estimated from 1024 feature vectors $\phi(x, y)$ to represent an image region. This number is sufficiently large compared with the dimensions of the covariance descriptor, which are only five. The improvement in the above results demonstrates the effectiveness of DSK over SK when there are sufficient samples to estimate the covariance descriptor. As discussed in Section 3.2.2, covariance matrix estimation is significantly affected by the number of samples. This motivates us to investigate how the



(a) On Brodatz dataset (b) On FERET dataset

Figure 3.1: Comparison of DSK-KA$_p$ and SK when different percentage of the 1024 feature vectors are used to estimate the covariance descriptor for each image on the Brodatz and FERET data sets.

performance of DSK and SK will change, if the number of feature vectors used to estimate the covariance descriptor is reduced. We take DSK-KA$_p$ as an example. In Figure 3.1(a), we plot the classification accuracy of DSK-KA$_p$ and SK averaged over the 15 binary classification tasks, when different percentage of the 1024 feature vectors are used. As shown, DSK-KA$_p$ consistently achieves better performance than SK, although both of them degrade with the decreasing number of feature vectors. This result shows that: i) When the samples available for estimation are inadequate, Stein kernel will become less effective; ii) DSK can effectively improve the performance of SK in this case. Figure 3.1(b) plots similar result obtained on the FERET face data set with the same experimental setting. That is, pairwise classification is performed by using SK with the k-NN classifier and the top 15 pairs with the lowest accuracies are selected. The classification accuracies of DSK-KA$_p$ and SK averaged over the 15 selected binary classification tasks are shown in Figure 3.1(b).

In this experiment, we observe that DSK can often be solved efficiently. The optimization in all the three frameworks only require a few iterations to converge.

Table 3.5: Comparison of Classification accuracy (in percentage) averaged on 198-class classification on FERET data set.

k-NN				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-KA$_p$	DSK-KA$_c$
84.45 ± 3.23	80.43 ± 3.54	52.14 ± 4.10	**84.98** ± 3.37	84.83 ± 3.38
SK	LEK	PEK	DSK-CS$_p$	DSK-CS$_c$
83.37 ± 3.33	83.02 ± 3.27	73.07 ± 3.66	84.03 ± 3.55	83.95 ± 3.45
SVM				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-RM$_p$	DSK-RM$_c$
N.A. N.A.	78.52 ± 2.23	68.37 ± 4.04	81.80 ± 2.67	**84.60** ± 1.71
SK	LEK	PEK	DSK-TM$_p$	DSK-TM$_c$
79.7 ± 3.10	78.16 ± 1.73	75.22 ± 3.97	80.70 ± 2.30	83.20 ± 2.44

3.4.2 Results on the FERET face data set

We evaluate the proposed DSK for face recognition on FERET [69] face data set. We use the 'b' subset, which consists of 198 subjects and each subject has 10 images with various poses and illumination conditions. Following the literature [31], to represent an image, a covariance descriptor is estimated for a 43-dimensional feature vector extracted at each pixel:

$$\phi(x, y) = [I(x, y), x, y, |G_{0,0}(x, y)|, \cdots, |G_{0,7}(x, y)|, |G_{1,0}(x, y)|, \cdots, |G_{4,7}(x, y)|];$$

where $I(x, y)$ is the intensity value, and $|G_{u,v}(x, y)|$ is the image feature of 2D Gabor wavelets [48].

Table 3.5 compares the classification accuracy on all the 198 classes of FERET data. As seen, DSK-KA$_p$ obtains an improvement of 1.6 percentage points over SK. The p-value between DSK-KA$_p$ and SK is 0.0026, which indicates the statistical significance

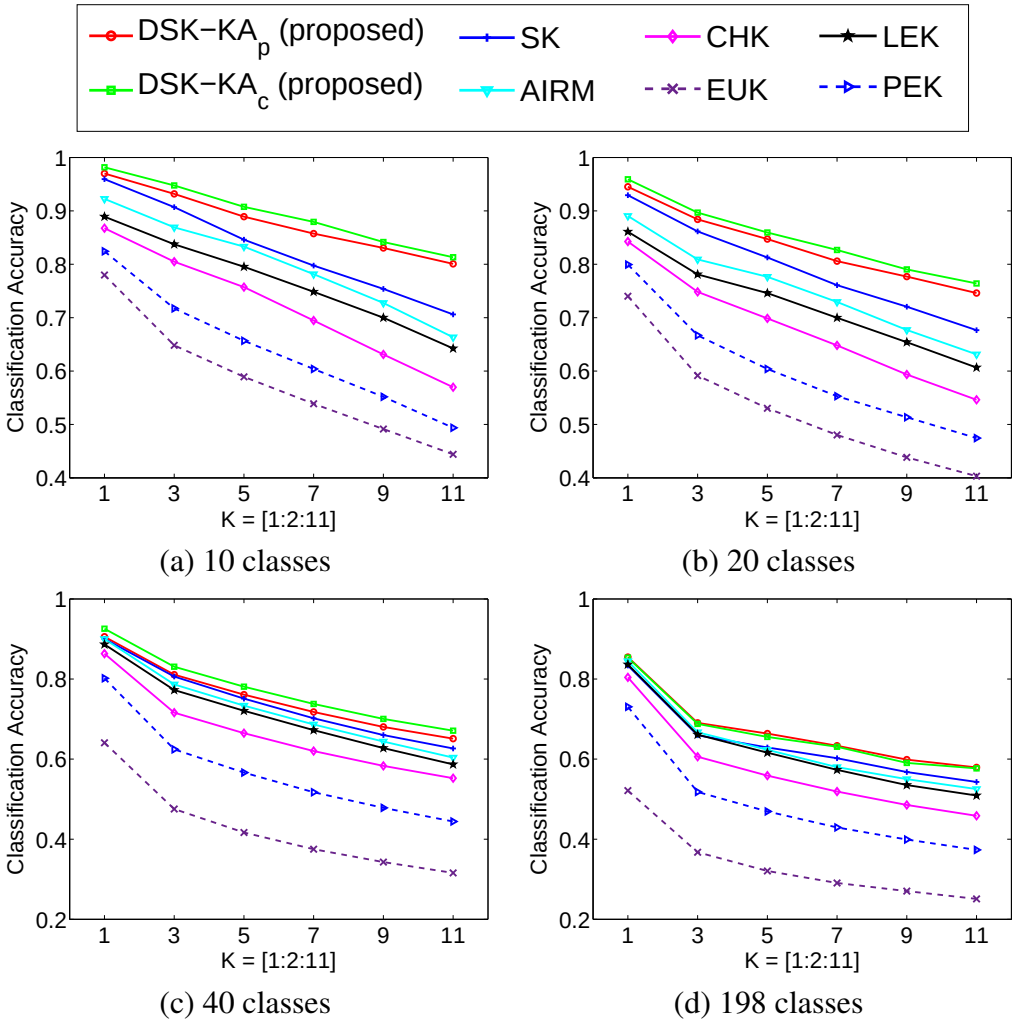


Figure 3.2: Comparison of DSK-KA, SK, and other methods on 10, 20, 40 and all 198 classes from FERET data set with various k values of k-NN.

of the improvement. When the radius margin bound framework is used, DSK also consistently performs better than SK. Especially, DSK-RM$_c$ achieves an improvement as high as 4.9 percentage points over SK, with p-value of 1.8×10^{-5}.

To test how the DSK performs with the number of classes and the k value of k-NN, we evaluate DSK-KA$_p$ and DSK-KA$_c$ by using 10, 20, 40, and all 198 classes, respectively, with $k = [1 : 2 : 11]$. The experiment is conducted as follows.

1) For the classification tasks with 10, 20 and 40 classes, these classes are randomly selected from the 198 classes. Five facial images are randomly chosen from a class for training, and the remaining ones are used for test. Both the selection of classes and samples are repeated 10 times, respectively.

2) For the classification tasks with all the 198 classes, five facial images are randomly chosen from each class for training, and the remaining ones are used for test. The selection of samples is repeated 20 times.

The averaged classification accuracy of 10, 20, 40 and 198 classes with various k values are plotted in Fig. 3.2(a)-(d). As seen, both DSK-KA$_p$ and DSK-KA$_c$ consistently outperform SK and other methods. This confirms in further that the proposed DSK can increase class discrimination by adjusting the eigenvalues of the SPD matrices, making Stein kernel better align with specific classification tasks.

3.4.3 Results on ETH-80 data set

ETH-80 contains eight categories with ten objects per category and 41 images for each object. The features same as those used on the Brodatz texture data set are extracted from each image and a 5×5 covariance descriptor is constructed as a representation of the image. The ten objects in the same category are labeled as the same class. We perform an eight-class classification task using DSK. As previously mentioned, data are randomly split into 20 pairs of training/test subsets ($50\% : 50\%$) to obtain stable statistics. Table 3.6 reports the performance of various methods. As seen, DSK still demonstrates the best performance. Specifically, DSK-CS$_p$ achieves 2.3 percentage points improvement over SK with k-NN as the classifier, while DSK-TM$_p$ achieves an improvement of 2.4 percentage points over SK, when SVM is used as the classifier.

The above experiments demonstrate the advantage of DSK in various important image recognition tasks. Also, this advantage is consistently observed when DSK is learned with three different criteria. This verifies the generality of DSK.

Table 3.6: Comparison of Classification accuracy (in percentage) averaged on ETH-80 data set.

k-NN				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-KA$_p$	DSK-KA$_c$
79.39 $\pm$ 0.78	80.14 $\pm$ 0.47	78.33 $\pm$ 1.19	80.92 $\pm$ 0.87	80.71 $\pm$ 0.85
SK	LEK	PEK	DSK-CS$_p$	DSK-CS$_c$
79.71 $\pm$ 0.82	79.29 $\pm$ 0.75	79.65 $\pm$ 0.65	**82.04** $\pm$ 0.91	80.11 $\pm$ 0.88
SVM				
Competing methods			DSK (proposed)	
AIRM	CHK	EUK	DSK-RM$_p$	DSK-RM$_c$
N.A. N.A.	80.76 $\pm$ 1.10	79.27 $\pm$ 1.98	81.30 $\pm$ 0.81	80.67 $\pm$ 0.93
SK	LEK	PEK	DSK-TM$_p$	DSK-TM$_c$
80.30 $\pm$ 0.79	80.21 $\pm$ 1.16	80.16 $\pm$ 1.04	**82.70** $\pm$ 1.05	81.55 $\pm$ 0.84

3.5 Discussion

3.5.1 On using α as power or coefficient

The two ways of using $\boldsymbol{\alpha}$ can be theoretically related. This is because for any $\lambda^{\alpha_p}(\lambda > 0)$, we can always find a coefficient α_c that satisfies $\alpha_c\lambda = \lambda^{\alpha_p}$ by setting $\alpha_c = \frac{\lambda^{\alpha_p}}{\lambda}$. In practice, these two ways could lead to different solutions, because the corresponding objective functions are different and the resulting optimizations are not convex. Comparatively, the power method is recommended due to: i) using $\boldsymbol{\alpha}$ as a power can automatically maintain the SPD property since $\lambda^{\alpha_p}(\lambda > 0)$ is always positive, while using it as a coefficient requires an additional constraint of $\alpha_c > 0$; ii) we empirically find that the power method often converges faster and achieves better performance than the coefficient method.

3.5.2 On the computational efficiency of DSK

Once the adjustment parameter $\boldsymbol{\alpha}$ is obtained, DSK can be computed for a set of SPD matrices. Fig. 3.3 compares the timing result of the methods in Table 2.1 for computing a similarity matrix of 100 SPD matrices. The dimensions of these SPD matrices are gradually increased from 5 to 100. The experiment is conducted on a desktop computer with 3.6 GHz Core^{TM} $\text{i7}-3820$ CPU and 32GB memory. As seen, DSK can be computed as efficiently as SK, PEK and LEK, and all of them are significantly faster than AIRM. For example, DSK only needs 3.3 seconds to compute the similarity matrix of 100-dimensional SPD matrices, while AIRM needs as many as 51 seconds. AIRM will become less efficient when the dimensions are high or the number of SPD matrices is large. The kernel methods, such as CHK, LEK, EUK, PEK, SK and DSK, can usually handle the situation more efficiently.

In addition, the computational cost of learning $\boldsymbol{\alpha}$ could be reduced by taking advantage of the facts that i) kernel matrix computation can be run in a parallel manner by evaluating every entry separately; ii) the most time-consuming step, eigen-decomposition, could be speeded up by using approximate techniques, such as the Nyström method [113]. These improvements will be fully explored in the future work.

3.5.3 More insight on when DSK works

By adjusting the eigenvalues of SPD matrices, DSK can increase the similarity of the SPD matrices within the same class and decrease the similarity of those different classes. We want to gain more insight on in what case DSK can work effectively. Let $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n, \mathbf{x}_{n+1}\}$ denote a set of d-dimensional vectors randomly sampled from a normal distribution $\mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. It is known that the scatter matrix $\mathbf{S}$ follows the Wishart distribution [92]: $\mathbf{S} \sim \mathcal{W}_d(\boldsymbol{\Sigma}, n)$, where $\mathbf{S}$ is defined as $\sum_{i=1}^{n+1}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\top$; $\mathbf{m}$ is defined as $\frac{1}{n+1}\sum_{i=1}^{n+1}\mathbf{x}_i$; and n is called the degree of freedom. Increasing the degree of

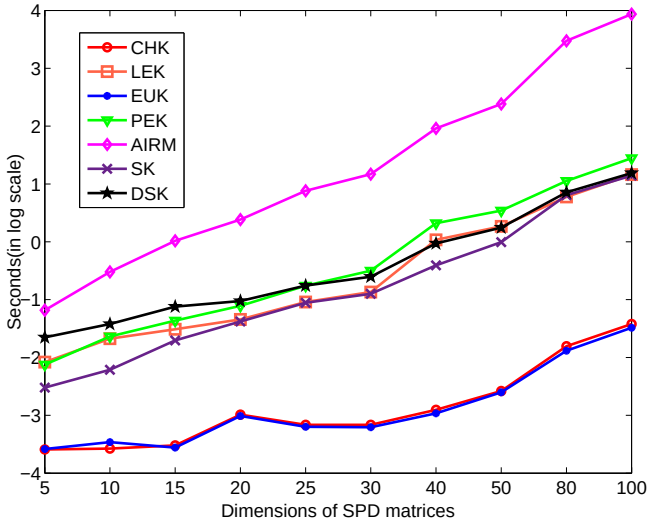
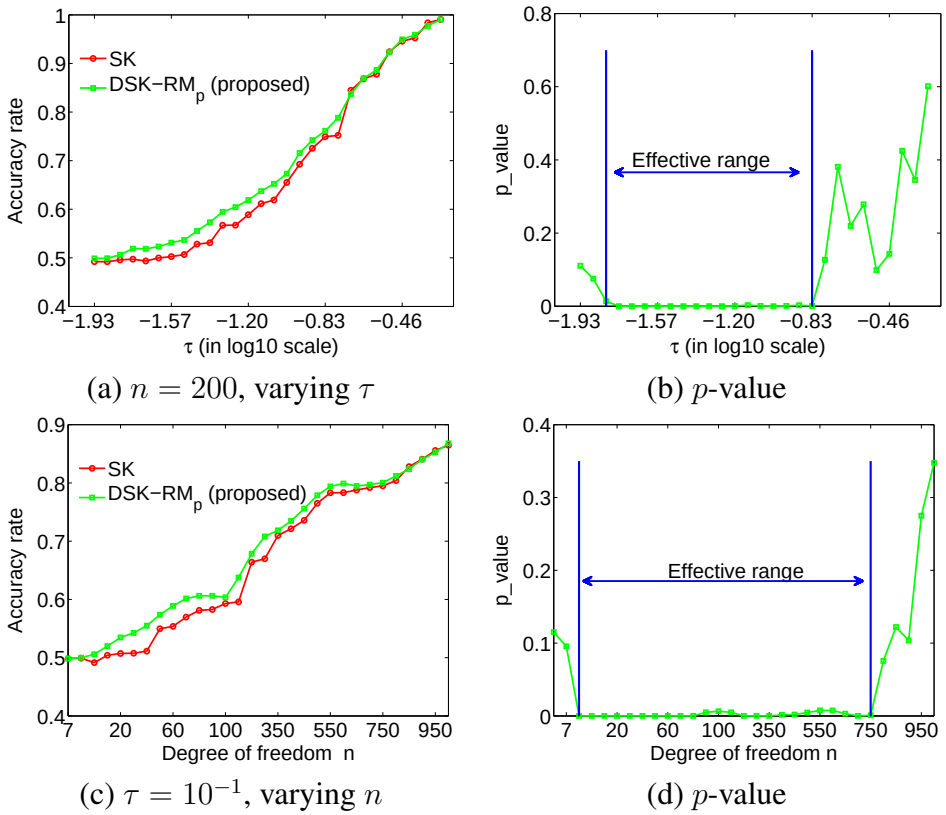


Figure 3.3: Comparison of timing results of the measures with respect to the dimensions of SPD matrices

freedom results in a smaller overall variance of $\mathbf{S}$ [63]. Note that the features extracted from an image region or an entire image can be considered as a random sample set $\mathcal{D}$ and the covariance descriptors used in the above experiments can be considered as the samples from certain Wishart distributions. In light of this connection, we use a set of synthetic SPD matrices sampled from various Wishart distributions to investigate the effectiveness of DSK. Specifically, in our experiment, two classes of SPD matrices are obtained by sampling Wishart distributions $W_d(\mathbf{\Sigma}_1, n)$ and $W_d(\mathbf{\Sigma}_2, n)$, where $\mathbf{\Sigma}_1$ is set as a 5×5 identity matrix $\mathbf{I}_{5\times5}$ and $\mathbf{\Sigma}_2$ is set as $(1+\tau)\mathbf{I}_{5\times5}$. τ is a small positive scalar and its magnitude controls the difference between $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. By varying τ and the degree of freedom n, we can generate a set of binary classification tasks with different levels of classification difficulty to evaluate DSK. First, we set n as 200 and vary τ to generate two classes of SPD matrices, with 1000 samples in each class. Larger τ will make the classification task easier since it leads to more different distributions. For each classification task, we randomly halve the samples to create 20 pairs of training/test subsets. Fig. 3.4(a) shows the performance of DSK (DSK-RM$_p$ is used as an example)



(a) $n = 200$, varying τ (b) p-value

(c) $\tau = 10^{-1}$, varying n (d) p-value

Figure 3.4: Comparison of DSK and SK on the synthetic data. (a) Performance of DSK and SK with fixed $n = 200$, varying τ. (b) The corresponding p-value obtained by the paired Student's t-test between DSK and SK with fixed $n = 200$, varying τ. (c) Performance of DSK and SK with fixed $\tau = 10^{-1}$, varying n. (d) The corresponding p-value obtained by the paired Student's t-test between DSK and SK with fixed $\tau = 10^{-1}$, varying n.

and SK with respect to τ, while Fig. 3.4(b) reports the corresponding p-values between DSK and SK. As seen, DSK can consistently achieve statistically significant (p-value $<$ 0.05) improvement over SK when $10^{-1.8} \leq \tau \leq 10^{-0.83}$. When τ is out of this range, the classification task becomes too difficult or too easy. In this case, DSK cannot improve the performance of SK. Similar results are obtained in Fig. 3.4(c) and Fig. 3.4(d), where we fix τ as 10^{-1} and change the degree of freedom n. As seen, DSK outperforms SK when $8 \leq n \leq 750$ and has a similar performance as SK otherwise.

This experiment reveals that DSK can effectively improve the performance of SK for a wide range of classification tasks unless the task is too difficult or too easy.

3.5.4 Comparison between DSK and the methods of improving eigenvalue estimation

Reshaping the eigenvalues of a sample covariance matrix has been extensively studied to improve the eigenvalue estimation and recover the true covariance matrix, especially when the number of samples is small [58, 17, 5]. We highlight the differences of the proposed DSK from these methods as follows. i) Handling the biasness of the eigenvalue estimation is only one of our motivations to propose DSK. The other more important motivation is to increase the discrimination of different sets of SPD matrices through eigenvalue adjustment; ii) DSK does not aim at (and is not even interested in) restoring the unbiased estimates of eigenvalues. Instead, DSK adaptively adjusts the eigenvalues of sample covariance matrices in a supervised manner to increase the discriminative power of Stein kernel.

Although DSK and the methods of improving eigenvalue estimation have different goals, it is still desirable to make a comparison between them in terms of the classification performance. We perform the comparison by using three kinds of SPD matrices: i) the sample covariance matrix; ii) the covariance matrix improved by the methods in [58, 17, 5]; and iii) the covariance matrix obtained by the proposed eigenvalue adjustment in DSK. Stein kernel is used in the classification with SVM as the classifier. As seen in Table 3.7, the three methods in [58, 17, 5] are comparable to the method using the sample covariance matrices on Brodatz, FERET and ETH80 data sets, while they obtain slightly better performance on fMRI data set. We believe that this is because the improved estimation of eigenvalues may boost the classification performance, when the ratio of sample size to the dimensions of covariance matrix (denoted by n/Dim in the table) is small. For example, the ratio n/Dim is less than two for the fMRI data set.

However, when the ratio becomes larger, the sample covariance matrices will gradually approach to the ground truth, and therefore the methods of improving eigenvalue estimation become less helpful. Since DSK aims to classify different sets of SPD matrices, the eigenvalues are adjusted towards better discriminability. This is why DSK achieves better performance than the methods in [58, 17, 5] on the four data sets. For example, the improvement of DSK is as high as 5.4 and 4.9 percentage points over the method in [17] on Brodatz and FERET data sets.

Table 3.7: Comparison of average classification accuracy (in percentage) between DSK and the methods of improving eigenvalue estimation.

Data	n/Dim	sample cov.	[58]	[17]	[5]	DSK
Brodatz	1,024/5 $\approx$ 205	78.01 $\pm$ 0.43	77.50 $\pm$ 0.41	78.00 $\pm$ 0.43	78.00 $\pm$ 0.48	**83.40** $\pm$ 0.58
FERET	98,304/43 $\approx$ 2286	79.70 $\pm$ 3.10	78.10 $\pm$ 2.98	79.70 $\pm$ 3.10	79.68 $\pm$ 3.10	**84.60** $\pm$ 1.71
ETH80	16,384/5 $\approx$ 3276	80.30 $\pm$ 0.79	78.80 $\pm$ 0.89	80.30 $\pm$ 0.82	80.31 $\pm$ 0.59	**82.70** $\pm$ 1.05
fMRI	130/90 $\approx$ 1.44	54.88	54.88	56.10	56.10	**59.76**

3.5.5 On the discovery of better SPD kernels

At last, we discuss what aspects may benefit the discovery of better kernels for SPD matrices. From our point of view, the following two aspects play an important role.

Distance measure. A good distance measure should effectively take the underlying structure of SPD matrices into account. In this chapter, we utilize the recently developed Stein kernel to meet this requirement. As a specially designed distance measure, the (square-rooted) S-Divergence well respects the Riemannian manifold where SPD matrices reside. The other distance measures, such as Cholesky, Log Euclidean, and Power Euclidean, listed in Table 2.1 could be investigated within our framework in the future work. Also, all the existing distance measures for SPD matrices (except the

simplest Euclidean distance) involve matrix decomposition or inverse operation. This results in significant computational cost, especially when a kernel matrix needs to be computed over a large sample set. In the course of discovering better SPD kernels, a computationally efficient distance measure will be highly desirable.

Class information. The class information should be effectively integrated into SPD kernels to improve its quality in further. In this work, we achieve this by utilizing the class information to adjust the eigenvalues to make Stein kernel better align with specific classification tasks. For Stein kernel, adjusting the eigenvalues only (rather than including the matrix of eigenvectors) may have been sufficient, because this kernel is invariant to affine transformations[3] There could be different but effective adjustment ways for other types of kernels and this is worth exploring in further. Besides, we focus on improving a SPD kernel in the supervised learning case in this work. Nevertheless, the proposed approach shall be extendable to the unsupervised case, which usually has a wider range of applications. In that situation, how to incorporate cluster information to improve SPD kernels will also be an interesting topic to explore.

In addition, as shown in this work, discovering better SPD kernels may need to optimize certain properties of SPD matrices. In this case, how to design and solve the resulting optimization problem becomes a critical issue. In particular, having a convex objective function and a computationally efficient optimization algorithm will be of great importance. This could be possibly achieved by appropriately convexifying and approximating the employed non-convex objective functions. In this work, we focus on validating the effectiveness of the proposed approach and demonstrating its advantages, and employ the commonly used gradient-based techniques to solve the involved optimization problems. In our future work, more advanced optimization techniques and algorithms will be developed to improve the proposed approach in further.

[3]It means that the S-Divergence is invariant to any nonsingular congruence transformation on the SPD matrices, i,e., $S(\mathbf{X}, \mathbf{Y}) = S(\mathbf{W}^\top \mathbf{X}\mathbf{W}, \mathbf{W}^\top \mathbf{Y}\mathbf{W})$, where $\mathbf{W}$ is an invertible matrix.

3.6 Conclusion

In this chapter, we analyzed two potential issues of the recently proposed Stein kernel for classification tasks and proposed a novel method called discriminative Stein kernel. It automatically adjusts the eigenvalues of the input SPD matrices to help Stein kernel to achieve greater discrimination. This problem is formulated as a kernel parameter learning process and solved in three frameworks. The proposed kernel is evaluated on both synthetic and real data sets for a variety of applications in pattern analysis and computer vision. The results show that it consistently achieves better performance than the original Stein kernel and other methods for SPD matrices. We also provided more insights on when and how DSK works, discussed the aspects that could contribute to discovering better SPD kernels, and pointed out the future work to enhance the proposed approach.

Chapter 4

Compact Representation of SICE Matrices in Brain Network Analysis

Recently, sparse inverse covariance estimation (SICE) technique has been employed to model functional brain connectivity. SICE matrix estimated for each subject can be used as a representation of brain connectivity network to discriminate Alzheimer's disease from normal controls. However, this chapter observes that direct use of the SICE matrix does not necessarily give satisfying discrimination, due to its high dimensionality and the scarcity of training subjects. Looking into this problem, this chapter argues that the intrinsic dimensionality of these SICE matrices shall be much lower, considering i) an SICE matrix resides on a Riemannian manifold of symmetric positive definiteness (SPD) matrices, and ii) human brains share common patterns of connectivity across subjects. Therefore, this chapter proposes to employ manifold-based similarity measures and kernel-based PCA to extract principal connectivity components as a compact representation of brain network. Moreover, to cater for the requirement of both discrimination and interpretation in neuroimage analysis, this chapter develops a novel pre-image estimation algorithm to make the obtained connectivity components anatomically interpretable. To verify the efficacy of the proposed method and gain insights into

SICE based brain networks, extensive experimental study is conducted on synthetic data and real rs-fMRI data from the ADNI data set. The proposed method outperforms the comparable methods and improves the classification accuracy significantly.

4.1 Background

As an incurable and the most common form of dementia, Alzheimer's disease (AD) affects tens of million people worldwide. Precise diagnosis of AD, especially at its early warning stage: Mild Cognitive Impairment (MCI), enables treatments to delay or even avoid cognitive symptoms, such as language disorder and memory loss [61]. However, this is a very challenging task. Conventional diagnosis of MCI based on clinical observations and structural imaging [39] can hardly achieve accurate diagnosis since the symptoms of MCI are often ambiguous and not necessarily related to structural alterations [76]. Various studies [118, 100, 97] have demonstrated that diagnosis of AD can be achieved by constructing and classifying functional brain networks based on resting-state functional Magnetic Resonance Imaging (rs-fMRI) using SICE technique.

However, how to classify the constructed SICE matrices can be challenging. A direct approach could be to vectorize each SICE matrix into a feature vector, as in [51]. However, when using it to train a classifier to separate AD from normal controls (NC), the problem of "the curse of dimensionality" arises since the dimensionality of the vector (at the order of $d \times d$[1] for a network with d nodes, for example, $d = 90$ in our study) is usually much larger than the number of training subjects, which is often only tens for each class. This usually leads to poor performance of classification. An alternative approach is to summarize a $d \times d$ SICE matrix into lower dimensional graphical features such as local clustering coefficient (LCC) [112] or hubs [87]. Nevertheless, these approaches have the risk of losing useful information contained in the SICE matrices.

[1]To be precise, the dimensionality of the vector is $\frac{d(d-1)}{2}$ because the SICE matrix is symmetric and its diagonal entries are not used.

This chapter aims to address the high dimensionality issue of these SICE matrices by extracting compact representation for classification.

4.2 Motivation

As an inverse covariance matrix, an SICE matrix is symmetric positive definite (SPD). This inherent property restricts SICE matrices to a lower-dimensional Riemannian manifold rather than the full $d \times d$ dimensional Euclidean space. In medical image analysis, the concept of Riemannian manifold has been widely used for DTI analysis [68], shape statistics [20] and functional-connectivity detection [100]. Moreover, considering the fact that brain connectivity patterns are specific and generally similar across different subjects, the SICE matrices representing the brain connectivity should concentrate on an even smaller subset of this manifold. In other words, the intrinsic degree of freedom of these SICE matrices shall be much lower than the apparent dimensions of $d \times d$. These two factors motivate us to seek a compact representation that better reflects the underlying distribution of the SICE matrices.

Principal component analysis (PCA), the commonly used unsupervised dimensionality reduction method, is a natural option for this task. However, a linear PCA is not expected to work well for manifold-constrained SICE matrices. Recently, advances have been made on measuring the similarity of SPD matrices considering the underlying manifold that they reside. In particular, a set of SPD kernels, e.g. Stein kernel [88] and Log-Euclidean kernel [3], have been proposed with promising applications [31, 40]. These kernels implicitly embed the Riemannian manifold of SPD matrices to a kernel-induced feature space $\mathcal{F}$. They offer better measure than their counterparts in Euclidean spaces and require less computation than Riemannian metric, as detailed in [88]. This chapter takes advantage of these kernels to conduct a SPD-kernel-based PCA. This provides two advantages: i) It produces a compact representation that can mitigate the curse

of dimensionality and, thus, improves classification. ii) The extracted leading eigenvectors in $\mathcal{F}$ can reveal the intrinsic structure of the SICE matrices, and, hence, assist brain network analysis.

While the approach introduced above could significantly improve the classification accuracy, another problem arises: how to interpret the obtained compact representation anatomically, or more specifically, is it possible to visualize the principal connectivity components identified by a SPD-kernel PCA? This is important in neuroimage analysis, as it could possibly help to reveal the disease mechanisms behind. Since SPD-kernel PCA is implicitly carried out in the kernel-induced feature space $\mathcal{F}$, the extracted eigenvectors in $\mathcal{F}$ are not explicitly known and therefore cannot be readily used for anatomical analysis. A kernel pre-image method has to be employed to recover these eigenvectors in the original input space. However, estimating the pre-images of an object in $\mathcal{F}$ is challenging. Existing pre-image methods [47, 75] require the knowledge of an explicit distance mapping between an input space and the feature space $\mathcal{F}$. Unfortunately, such an explicit distance mapping is intractable for SPD kernels, and thus the existing pre-image methods cannot be applied to our case. To solve this problem, this chapter further proposes a novel pre-image method for the SPD kernels and use it to gain insight into SICE-based brain network analysis.

To verify the proposed approach, this chapter conducts extensive experimental study on both synthetic data set and rs-fMRI data from the benchmark dataset ADNI [2]. As will be seen, the results well demonstrate the effectiveness and advantages of our method. Specifically, the proposed compact representation obtained via the SPD-kernel PCA achieves superior classification performance to that from linear PCA and the graphical feature LCC. Also, the proposed pre-image method can effectively recover in the original input space the principal connectivity components identified in a feature space and enables the visualization and anatomical analysis of these components.

[2] http://adni.loni.usc.edu

In addition, we would like to point out that besides SICE matrices, the proposed method can be seamlessly applied to the correlation matrices previously mentioned, because they are also symmetric positive definite. We focus on SICE matrices in this chapter because SICE matrices model the partial correlations which enjoy theoretical advantages and generally admit more stable connectivity in comparison with correlation [85].

The rest of the chapter is organized as follows: Section 4.3 details the proposed SPD-kernel PCA and pre-image method. Section 4.4 presents the experimental results on synthetic and real rs-fMRI data sets. And finally section 4.5 concludes this chapter.

4.3 Proposed Method

4.3.1 SICE representation using SPD-kernel based PCA

In spite of individual variation, human brains do share common, specific connectivity patterns across different subjects. Therefore, the SICE matrices used to represent brain networks shall have similar structures across subjects. This makes them be further restricted into a small subset of the Riemannian manifold of SPD matrices, with a limited degree of freedom. Inspired by this observation, we aim to extract a compact representation of these SICE matrices for better classification and analysis. Principal component analysis (PCA) is a commonly used technique to generate a compact representation of data by exploring a subspace that can best represent the data. Therefore, PCA is a natural choice for our task. However, linear PCA is not expected to work well for the SICE matrices because it does not consider the manifold structure. Consequently, we adopt kernel PCA [80] and integrate SPD kernels for similarity measure. This effectively accounts for the manifold structure of SICE matrices when exploring the subspace of the data. Our method is elaborated as follows.

The SICE method is applied to N subjects to obtain a training set $\{\mathbf{S}_1, \mathbf{S}_2, \cdots, \mathbf{S}_N\} \subset$

Sym_d^+, where $\mathbf{S}_i$ is the SICE matrix for the i-th subject. We define the kernel mapping $\Phi(\cdot)$: $\mathrm{Sym}_d^+ \mapsto \mathcal{F}$, which cannot be explicitly solved but implicitly induced by a given SPD kernel. As an extension of PCA, kernel PCA generalizes linear PCA to a kernel-induced feature space $\mathcal{F}$. For the self-containedness of this thesis, we briefly describe Kernel PCA as follows and the details can be found in [80]. Without loss of generality, it is assumed that $\Phi(\mathbf{S}_i)$ is centered, i.e. $\sum_{i=1}^N \Phi(\mathbf{S}_i) = \mathbf{0}$, and, as in [80], this can be easily achieved by simple computation with kernel matrix. Then a $N \times N$ kernel matrix $\mathbf{K}$ can be obtained with each entry $\mathbf{K}_{ij} = \langle \Phi(\mathbf{S}_i), \Phi(\mathbf{S}_j) \rangle = k(\mathbf{S}_i, \mathbf{S}_j)$. Kernel PCA first performs the eigen-decomposition on the kernel matrix: $\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$. The i-th column of $\mathbf{U}$, denoted by $\mathbf{u}_i$, corresponds to the i-th eigenvector, and $\mathbf{\Lambda}$ = diag($\lambda_1, \lambda_2, \cdots, \lambda_N$), where λ_i corresponds to the i-th eigenvalue in a descending order. Let $\mathbf{\Sigma}_\Phi$ denote the covariance matrix computed by $\{\Phi(\mathbf{S}_i)\}_{i=1}^N$ in $\mathcal{F}$. The i-th eigenvector of $\mathbf{\Sigma}_\Phi$ can be expressed as:

$$\mathbf{v}_i = \frac{1}{\sqrt{\lambda_i}} \mathbf{\Phi} \mathbf{u}_i, \tag{4.1}$$

where $\mathbf{\Phi} = [\Phi(\mathbf{S}_1), \Phi(\mathbf{S}_2), \ldots, \Phi(\mathbf{S}_N)]$. Analogous to linear PCA, for a given SICE matrix $\mathbf{S}$, $\Phi(\mathbf{S})$ can then be projected onto the top m eigenvectors to obtain an m-dimensional principal component vector:

$$\boldsymbol{\alpha} = \mathbf{V}_m^\top \Phi(\mathbf{S}),$$

where $\mathbf{V}_m = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m]$. Note that the i-th component of $\boldsymbol{\alpha}$, denoted by α_i, is $\mathbf{v}_i^\top \Phi(\mathbf{S})$. With the kernel trick, it can be computed as:

$$\alpha_i = \mathbf{v}_i^\top \Phi(\mathbf{S}) = \frac{1}{\sqrt{\lambda_i}} \mathbf{u}_i^\top \mathbf{\Phi}^\top \Phi(\mathbf{S}) = \frac{1}{\sqrt{\lambda_i}} \mathbf{u}_i^\top \mathbf{k}_\mathbf{S}, \tag{4.2}$$

where $\mathbf{k}_\mathbf{S} = [k(\mathbf{S}, \mathbf{S}_1), k(\mathbf{S}, \mathbf{S}_2), \ldots, k(\mathbf{S}, \mathbf{S}_N)]^\top$. Once $\boldsymbol{\alpha}$ is obtained as a new representation for each SICE matrix, an SVM or k-NN classifier can be trained on $\boldsymbol{\alpha}$ with

class labels.

In this chapter, we study four commonly used SPD kernels, namely, Cholesky kernel (CHK) [40], Power Euclidean kernel (PEK) [40], Log-Euclidean kernel (LEK) [3] and Stein kernel (SK) [88]. The four kernels are all in a form of

$$k(\mathbf{S}_i, \mathbf{S}_j) = \exp\big(-\theta \cdot d^2(\mathbf{S}_i, \mathbf{S}_j)\big), \tag{4.3}$$

where $d(\cdot,\cdot)$ is a kind of distance between two SPD matrices. Different definitions of $d(\cdot,\cdot)$ lead to different kernels, and the distance functions in the four kernels are Cholesky distance [16], Power Euclidean distance [16], Log-Euclidean distance [3] and root Stein divergence [88], respectively.

The four distance functions and the corresponding kernels are summarized in Table 2.1. They will be applied to SPD-kernel PCA to produce the principal component vector $\boldsymbol{\alpha}$.

4.3.2 Pre-image estimation

As will be shown in the experimental study, the principal components $\boldsymbol{\alpha}$ extracted by the above SPD-kernel PCA offer promising classification performance. Note that $\boldsymbol{\alpha}$ is fundamentally determined by the m leading eigenvectors $\mathbf{v}_1, \cdots, \mathbf{v}_m$, which capture the underlying structure of SICE matrices and can be deemed as the building blocks of this representation of brain connectivity. Therefore, analyzing these eigenvectors is important for the understanding and interpretation of the obtained principal connectivity patterns. However, the eigenvectors are derived in $\mathcal{F}$ via the implicit kernel mapping $\Phi(\cdot)$, and thus are not readily used for analysis in the input space Sym_d^+. To tackle this issue, we aim to develop a method that can project a data point in the subspace spanned by the m leading eigenvectors in $\mathcal{F}$ back to the input space. This will allow the visualization of the principal connectivity patterns in the input space for interpretation.

This is known as the "pre-image" problem of kernel methods in the literature [47, 75, 122]. Unfortunately, existing pre-image methods, such as those in [47, 75], cannot be applied to our case, because they require an explicit mapping between the Euclidean distance in $\mathcal{F}$ and the Euclidean distance in the input space, which is unavailable when the SPD kernels are used. In the following, we develop a novel pre-image method for the SPD kernels to address this issue.

Let $\Phi_m(\mathbf{S})$ denote the projection of $\Phi(\mathbf{S})$ into the subspace spanned by the m leading eigenvectors in $\mathcal{F}$, that is:

$$\begin{aligned}\Phi_m(\mathbf{S}) &= \sum_{i=1}^{m} \alpha_i \mathbf{v}_i = \sum_{i=1}^{m} \frac{1}{\sqrt{\lambda_i}} \mathbf{u}_i^\top \mathbf{k}_\mathbf{S} \cdot \frac{1}{\sqrt{\lambda_i}} \mathbf{\Phi} \mathbf{u}_i \\ &= \sum_{i=1}^{m} [\mathbf{k}_\mathbf{S}^\top \frac{1}{\lambda_i} \mathbf{u}_i \cdot \mathbf{u}_i^\top \mathbf{\Phi}^\top]^\top = \mathbf{\Phi} \mathbf{M} \mathbf{k}_\mathbf{S}\end{aligned} \tag{4.4}$$

where $\mathbf{M} = \sum_{i=1}^{m} \frac{1}{\lambda_i} \mathbf{u}_i \mathbf{u}_i^\top$ and recall $\mathbf{\Phi} = [\Phi(\mathbf{S}_1), \Phi(\mathbf{S}_2), \ldots, \Phi(\mathbf{S}_N)]$ and $\mathbf{k}_\mathbf{S} = [k(\mathbf{S}, \mathbf{S}_1), k(\mathbf{S}, \mathbf{S}_2), \ldots, k(\mathbf{S}, \mathbf{S}_N)]^\top$. Our aim is to find a pre-image $\hat{\mathbf{S}}$ in the original input space (that is, Sym_d^+) which best satisfies $\Phi(\hat{\mathbf{S}}) = \Phi_m(\mathbf{S})$. Considering the fact that Riemannian manifold is locally homeomorphic with a Euclidean space [95], we model $\hat{\mathbf{S}}$ by a linear combination[3] of its neighboring SICE matrices in Sym_d^+. Similar to the work in [47], we assume that if $\mathbf{S}_i$ and $\mathbf{S}_j$ are close in Sym_d^+, then $\Phi(\mathbf{S}_i)$ and $\Phi(\mathbf{S}_j)$ shall also be close in $\mathcal{F}$. With this assumption, we can obtain the neighbors of $\hat{\mathbf{S}}$ in Sym_d^+ by finding the neighbors of $\Phi_m(\mathbf{S})$ in $\mathcal{F}$.

Specifically, $\hat{\mathbf{S}}$ is estimated as follows. Firstly, we find a set of nearest neighbors

[3]Using linear combination of neighbors may restrict the search space of pre-image and could affect the reconstruction accuracy. Here we use it for three reasons: i) our experiment on synthetic data (with ground truth) has demonstrated good reconstruction result; ii) using linear combination can significantly simplify the optimization problem of pre-image estimation; iii) by using linear combination of neighbors, we can better enforce the constructed pre-image to follow the underlying distribution of training samples.

$\Omega = \{\mathbf{S}_j\}_{j=1}^{L}$ for $\hat{\mathbf{S}}$ from a training set $\{\mathbf{S}_i\}_{i=1}^{N}$ by sorting the following distance

$$\begin{aligned} d^2(\Phi_m(\mathbf{S}), \Phi(\mathbf{S}_i)) \triangleq & ||\Phi_m(\mathbf{S}) - \Phi(\mathbf{S}_i)||^2 = ||\Phi_m(\mathbf{S})||^2 + ||\Phi(\mathbf{S}_i)||^2 - 2\Phi_m(\mathbf{S})^\top \Phi(\mathbf{S}_i) \\ = & \left(\sum_{i=1}^{m} \alpha_i \mathbf{v}_i\right)^\top \left(\sum_{i=1}^{m} \alpha_i \mathbf{v}_i\right) + k(\mathbf{S}_i, \mathbf{S}_i) - 2\left(\mathbf{\Phi M k_S}\right)^\top \Phi\left(\mathbf{S}_i\right) \\ = & \sum_{i=1}^{m} \alpha_i^2 + k(\mathbf{S}_i, \mathbf{S}_i) - 2\mathbf{k}_\mathbf{S}^\top \mathbf{M} \mathbf{\Phi}^\top \Phi\left(\mathbf{S}_i\right) \\ & \text{(By applying Eq. } (4.2)) \\ = & (\mathbf{k}_\mathbf{S}^\top - 2\mathbf{k}_{\mathbf{S}_i}^\top)\mathbf{M k_S} + k(\mathbf{S}_i, \mathbf{S}_i). \end{aligned} \tag{4.5}$$

This distance can be easily computed because it is fully represented by the kernel functions.

Secondly, we model the pre-image $\hat{\mathbf{S}}$ by a convex (linear) combination of its neighbors as

$$\hat{\mathbf{S}} = \sum_{j=1}^{L} w_j \mathbf{S}_j, \tag{4.6}$$

where $\mathbf{S}_j \in \Omega$, $w_j \geq 0$, and $\sum_{j=1}^{L} w_j = 1$. This convex combination guarantees the SPD of $\hat{\mathbf{S}}$ and also makes it be effectively constrained by its L neighbors. Defining $\mathbf{w} = [w_1, w_2, \cdots, w_L]^\top$, we seek the optimal $\mathbf{w}$ by solving

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \geq 0;\ \mathbf{w}^\top \mathbf{1} = 1} d^2 \left(\Phi_m(\mathbf{S}), \Phi\left(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j\right)\right). \tag{4.7}$$

where $d^2\left(\Phi_m(\mathbf{S}), \Phi(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j)\right) = d^2(\Phi_m(\mathbf{S}), \Phi(\hat{\mathbf{S}})) = (\mathbf{k}_\mathbf{S}^\top - 2\mathbf{k}_{\hat{\mathbf{S}}}^\top)\mathbf{M k_S} + k(\hat{\mathbf{S}}, \hat{\mathbf{S}})$ by applying Eq. (4.5) and (4.6). This optimization problem can be efficiently solved using gradient descent based algorithms. Note that Eq. (5.9) can be used to compute the pre-image of any data point $\Phi_m(\mathbf{S})$ in $\mathcal{F}$. In addition, when estimating the pre-

image of a specific eigenvector $\mathbf{v}_i$, we can simply set $\Phi_m(\mathbf{S})$ as $\mathbf{v}_i$ and solve the same optimization problem in Eq. (5.9). In this case, the objective function reduces to:

$$\begin{aligned} d^2(\mathbf{v}_i, \Phi(\mathbf{S}_i)) =& ||\mathbf{v}_i - \Phi(\mathbf{S}_i)||^2 = ||\mathbf{v}_i||^2 + ||\Phi(\mathbf{S}_i)||^2 - 2\mathbf{v}_i^\top \Phi(\mathbf{S}_i) \\ =& 1 + k(\mathbf{S}_i, \mathbf{S}_i) - 2\left(\frac{1}{\sqrt{\lambda_i}}\mathbf{\Phi}\mathbf{u}_i\right)^\top \Phi(\mathbf{S}_i) = 1 + k(\mathbf{S}_i, \mathbf{S}_i) - \frac{2}{\sqrt{\lambda_i}}\mathbf{u}_i^\top \mathbf{k}_{\mathbf{S}_i}. \end{aligned} \tag{4.8}$$

Algorithm 4 outlines the proposed pre-image algorithm.

Algorithm 3 Pre-image estimation for $\Phi_m(\mathbf{S})$ in $\mathcal{F}$

Input: A training set $\{\mathbf{S}_i\}_{i=1}^N$, test data $\mathbf{S}$, m;
Output: Pre-image $\hat{\mathbf{S}}$

1: Find a set of L neighbors $\Omega = \{\mathbf{S}_j\}_{j=1}^L$ for $\hat{\mathbf{S}}$ by sorting $d^2(\Phi_m(\mathbf{S}), \Phi(\mathbf{S}_i))$, $i = 1, \cdots, N$, according to Eq. (4.5);
2: Solve Eq. (5.9) to obtain $\mathbf{w}^*$:
$\mathbf{w}^* = \arg\min_{\mathbf{w}\geq 0;\ \mathbf{w}^\top \mathbf{1}=1} d^2(\Phi_m(\mathbf{S}), \Phi(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j))$;
3: **return** $\hat{\mathbf{S}} = \sum_{j=1}^L w_j \mathbf{S}_j$.

4.4 Experimental Study

4.4.1 Data preprocessing and experimental settings

Rs-fMRI data of 196 subjects were downloaded from the ADNI website[4] in June 2013. Nine subjects were discarded due to the corruption of data and the remaining 187 subjects were preprocessed for analysis. After removing subjects that had problems in the preprocessing steps, such as large head motion, 156 subjects were kept, including 26 Alzheimer's disease (AD), 44 early Mild Cognitive Impairment (MCI), 38 late MCI, 38 Normal Controls (NC) and 10 Significant Memory Concern (SMC), labeled by ADNI.

[4]http://adni.loni.usc.edu

We used the 38 NC and the 44 early MCI in this chapter because our focus in this chapter is to identify MCI at very early stage, which is the most challenging and significant task in AD prediction. The IDs of the 82 (38 NC and 44 early MCI) subjects are provided in the supplementary material. The data are acquired on a 3 Tesla (Philips) scanner with TR/TE set as 3000/30 ms and flip angle of 80°. Each series has 140 volumes, and each volume consists of 48 slices of image matrices with dimensions 64×64 with voxel size of $3.31 \times 3.31 \times 3.31\ mm^3$. The preprocessing is carried out using SPM8[5] and DPARSFA [9]. The first 10 volumes of each series are discarded for signal equilibrium. Slice timing, head motion correction and MNI space normalization are performed. Participants with too much head motion are excluded. The normalized brain images are warped into automatic anatomical labeling (AAL) [96] atlas to obtain 90 ROIs as nodes. By following common practice [86, 51, 112], the ROI mean time series are extracted by averaging the time series from all voxels within each ROI and then band-pass filtered to obtain multiple sub-bands as in [112].

The functional connectivity networks of 82 participants are obtained by the SICE method using SLEP [54], with the sparsity levels of $\lambda = [0.1 : 0.1 : 0.9]$. For comparison, constrained sparse linear regression (SLR) [112] is also used to learn functional connectivity networks with the same setting. Functional connectivity networks constructed by SICE and SLR are called "SICE matrices" and "SLR matrices" respectively. To make full use of the limited subjects, a leave-one-out procedure is used for training and test. That is, each sample is reserved for test in turn while the remaining samples are used for training. Both SVM and k-NN are used as the classifier to compare the classification accuracy of different methods. The parameters used in the following classification tasks of this rs-fMRI data set, including the sparsity level λ, the sub-band of the time series, the number of eigenvectors m and the regularization parameter of SVM are tuned by five-fold cross-validation on the training set. θ in all the four SPD kernels

[5]http://www.fil.ion.ucl.ac.uk/spm/software/

is empirically set as 0.5, and the k of k-NN is set as 7.

4.4.2 Experimental result

The experiment consists of three parts: 1) Evaluating the classification performance when the original SICE or SLR matrices are used as the features; 2) Evaluating the classification performance when the compact representation of SICE or SLR matrices is used as the features; 3) Investigating the effectiveness of the proposed pre-image method.

Table 4.1: Classification accuracy (in %) by directly using SICE/SLR matrices as features.

	Linear kernel (vectorized [51])		LEK (proposed)		SK (proposed)		CHK (proposed)		PEK (proposed)	
	k-NN	SVM	k-NN	SVM	k-NN	SVM	k-NN	SVM	k-NN	SVM
SLR [112]	53.7	52.4	N.A. Because SLR matrices are not necessarily SPD.							
SICE	57.3	57.3	61.0	61.0	63.4	64.6	61.0	62.2	61.0	**65.9**

Classification using original SICE or SLR matrices

By applying the SICE or SLR method to the rs-fMRI data, we can obtain the SICE or SLR matrices as the representation of brain networks. These matrices can be directly used as features to train a classifier. A straightforward way is to vectorize the matrices into high-dimensional vectors as features as in [51], which are then used to train a linear SVM or k-NN with linear kernel as the similarity measure to search nearest neighbors to perform classification. Note that linear kernel is Euclidean distance-based similarity measure. As shown in the second and third columns in Table 4.1 (labeled by 'linear kernel'), this method produces poor classification performance (lower than 60%) on both SICE and SLR matrices, be it k-NN or linear SVM is used as the classifier. Specifically, it only achieves 53.7% for the k-NN classifier using SLR matrices. When SICE matrices are used, the classification performance is only 57.3% too. The result

does not change much when a linear SVM is used. The poor classification performance of this method is largely due to two issues: i) The vectorization ignores the underlying structure of SICE matrices, and the linear kernel in SVM and in the k-NN classifier cannot effectively evaluate their similarity and distance; and ii) The "small sample size" problem occurs because the dimensionality of the resulting feature vectors is high while the training samples are limited.

In order to effectively consider the manifold geometry of SICE matrices, we employ the four aforementioned SPD kernels to evaluate the similarity between SICE matrices and adopt k-NN and SVM classifiers with these kernels to perform classification. As seen in the columns under "LEK", "SK", "CHK", "PEK" in Table 4.1, the classification accuracy with respect to each SPD kernel is above 60%, which clearly outperforms that of their linear counterparts. In particular, PEK obtains 65.9% with SVM as the classifier, achieving an improvement of 8.6 percentage points over linear SVM. This well verifies the importance of considering the manifold structure of SICE matrices for the classification. Note that because SLR matrices are not necessarily SPD, the SPD kernels cannot be applied. Therefore, no classification result is reported in the row of "SLR" in Table 4.1.

Table 4.2: Classification accuracy (in %) of compact representation on SICE/SLR matrices.

	LCC		Linear PCA		LEK PCA (proposed)	
	k-NN	SVM	k-NN	SVM	k-NN	SVM
SLR [112]	65.9	64.6	67.1	65.9	N.A.	
SICE	65.9	63.4	67.1	68.3	69.5	69.5
	SK PCA (proposed)		CHK PCA (proposed)		PEK PCA (proposed)	
	k-NN	SVM	k-NN	SVM	k-NN	SVM
SLR [112]	N.A. Because SLR matrices are not necessarily SPD.					
SICE	72	**73.2**	68.3	70.7	72	**73.2**

Table 4.3: Classification accuracy (in %) by using original SICE/SLR matrices and pre-images of $\Phi_m(\mathbf{S})$ with k-NN.

	SLR [112]	SICE	Pre-images of SICE (LEK, proposed)
Linear kernel	53.7	57.3	**68.3**
LCC	65.9	65.9	67.1
	Pre-images of SICE (SK, proposed)	Pre-images of SICE (CHK, proposed)	Pre-images of SICE (PEK, proposed)
Linear kernel	67.1	63.4	63.4
LCC	67.1	64.6	**68.3**

Classification using the compact representation

In this experiment, we compare the classification performance of the compact representation obtained by the proposed SPD-kernel PCA, linear PCA and the method computing local clustering coefficient (LCC) [112]. LCC, as a measure of local neighborhood connectivity for a node, is defined as the ratio of the number of existing edges between the neighbors of the node and the number of potential connections between these neighbors [42]. In this case, LCC can map a network, represented by a $d \times d$ Adjacency matrix, to a d-dimensional vector, where d is the number of nodes in the network.

Table 4.2 shows the classification results when using the compact representation of SICE or SLR matrices using k-NN with Euclidean distance and linear kernel SVM. LCC achieves 65.9% for both SICE and SLR matrices with k-NN as the classifier. It is better than the result (53.7% and 57.3% in the second column of Table 4.1) of directly using the original matrices and is comparable to the result (65.9%) of applying PEK-SVM, the best one obtained in Table 4.1. When linear PCA is applied to the vectorized SICE or SLR matrices to extract the top m principal components as features, the classification accuracy increases to 67.1% for both SICE and SLR matrices. This performance is better than LCC and all the methods in Table 4.1. Such a result indicates the power of compact representation and also preliminarily justifies our idea of exploring the lower intrinsic dimensions of the SICE matrices. By further taking the SPD property into

account and using the proposed SPD-kernel PCA to extract the compact representation, the classification accuracy is significantly boosted up to 73.2% for both SK-PCA and PEK-PCA, with SVM as the classifier. This achieves an improvement of 4.9 percentage points (73.2% vs. 68.3%) over linear PCA and 7.3 percentage points (73.2% vs. 65.9%) over LCC. These results well demonstrate that: i) The obtained compact representation can effectively improve the generalization of the classifier in the case of limited training samples. ii) It is important to consider the manifold property of SICE matrices in order to obtain better compact representation. Cross-referencing the SICE results in Table 4.1 and Table 4.2, SPD-kernel PCA achieves the best classification performance, i.e. 73.2%, obtaining an improvement of 15.9 percentage points over the linear kernel method (57.3%, in Table 4.1).

Investigating the proposed pre-image method

The two goals of the pre-image method, which is shown in Algorithm 4, is to estimate the pre-image of i) $\Phi_m(\mathbf{S})$, which is the projection of $\Phi(\mathbf{S})$ into the m leading eigenvectors in $\mathcal{F}$ and ii) one single eigenvector $\mathbf{v}_i$ of SPD-kernel PCA in $\mathcal{F}$.

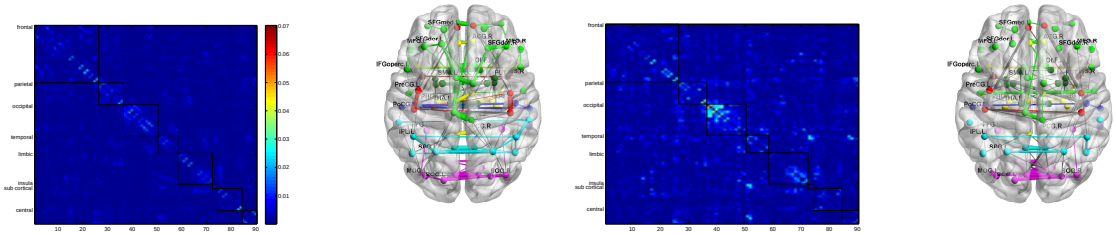
The motivation of the first goal to recover the pre-image of $\Phi_m(\mathbf{S})$ is inspired by the property of PCA. It is known that projecting data into the m leading eigenvectors discards the minor components which often correspond to data noise. Therefore, when an SICE matrix $\mathbf{S}$ is contaminated by noise (and it makes $\Phi(\mathbf{S})$ noisy), $\Phi_m(\mathbf{S})$ can be regarded as a "denoised" version of $\Phi(\mathbf{S})$. As a result, if the proposed pre-image method really works, the recovered pre-image shall be closer to the true inverse covariance matrix than $\mathbf{S}$ is. In the literature, such a property has been extensively used for data and image denoising [59].

The proposed pre-image method is performed on the real rs-fMRI data. Here we aim to investigate if the pre-images can boost the classification performance in comparison with the original SICE matrices based on the assumption that the pre-image of $\Phi_m(\mathbf{S})$

can bring some kind of denoising effect. We first estimate the pre-images of $\Phi_m(\mathbf{S}_i)$, $\mathbf{S}_i \in \{\mathbf{S}_i\}_{i=1}^{82}$ and redo classification using two methods: i) *Linear kernel method*. As what we did in the second column of Table 4.1, k-NN classifier is directly applied to the obtained pre-images with linear kernel as the similarity measure; ii) *LCC method*. As what we did in the second column of Table 4.2, LCC is extracted as a feature from the obtained pre-images and apply k-NN classifier to LCC with Euclidean distance. The number of leading eigenvectors m is selected by cross-validation from the range of $[1:5:80]$ on the training set while the number of neighbors L is empirically set as 20. In our experiment, we observe that i) A larger L will make the optimization significantly more time-consuming while the performance of the method remains similar; ii) The selected value of m is usually in the range of $[15 \sim 35]$.

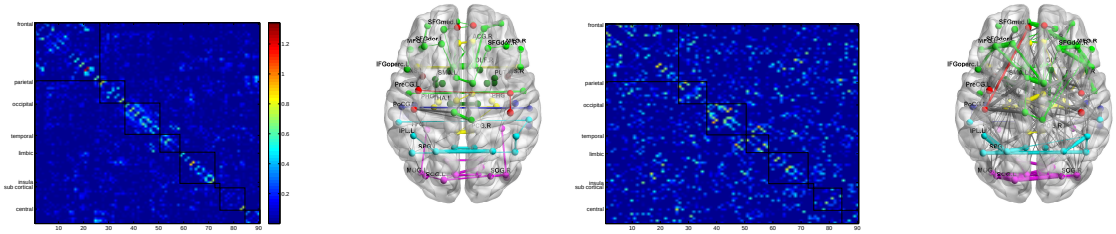
Table 4.3 shows the classification result on the pre-images of $\Phi_m(\mathbf{S}_i)$, $\mathbf{S}_i \in \{\mathbf{S}_i\}_{i=1}^{82}$, obtained on the real rs-fMRI data. The classification performance with the pre-images when SK, LEK, and PEK are used can consistently outperform the classification performance with original SICE or SLR matrices using either linear kernel method or LCC method. Specifically, the performance of linear kernel method on SICE matrices is boosted to 68.3% (the fourth column, with pre-images when LEK is used) from 57.3% (the third column). We believe that the improvement is due to that, by estimating the pre-images of $\Phi_m(\mathbf{S}_i)$ in $\mathcal{F}$, the resulting matrices are more reliable than the original SICE matrices.

Recall that the leading eigenvectors $\mathbf{v}_i$ in $\mathcal{F}$ capture the underlying structure of SICE matrices and can be deemed as the building blocks of the representation for brain connectivity. Thus we estimate the pre-image of top eigenvectors $\mathbf{v}_i$ in $\mathcal{F}$ for anatomical analysis. In this experiment, the pre-images of the top two eigenvectors, which pose the most significant variance of SICE matrices in $\mathcal{F}$, are visualized in Fig. 4.1. The lobe, index, and name of each ROI in AAL [96] atlas are listed in Table 4.4. We observe that: i) Compared with the eigenvectors in linear PCA, the eigenvectors obtained in the SPD-



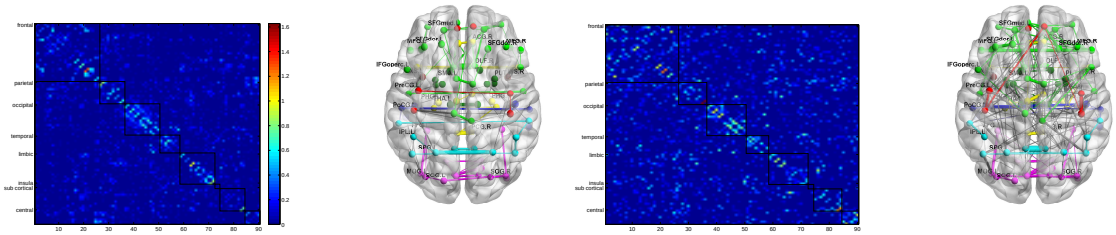
(a) Eigenvector 1 of linear PCA

(b) Eigenvector 2 of linear PCA



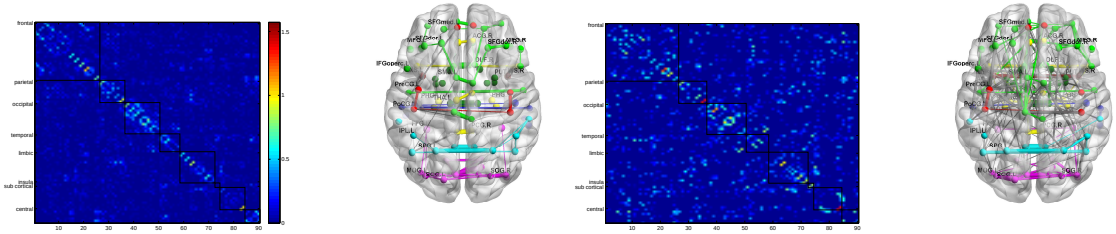
(c) Eigenvector 1 of CHK PCA

(d) Eigenvector 2 of CHK PCA



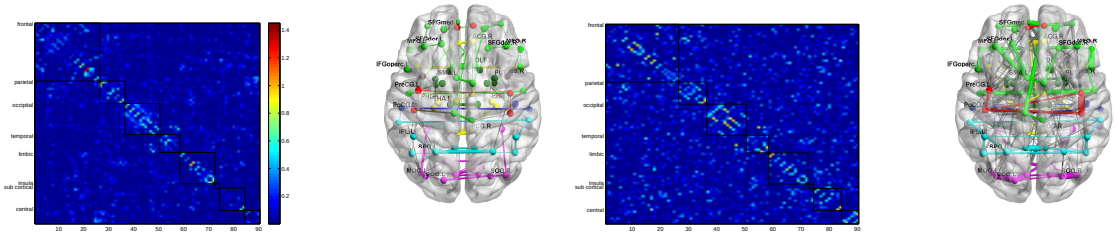
(e) Eigenvector 1 of PEK PCA

(f) Eigenvector 2 of PEK PCA



(g) Eigenvector 1 of LEK PCA

(h) Eigenvector 2 of LEK PCA



(i) Eigenvector 1 of SK PCA

(j) Eigenvector 2 of SK PCA

Figure 4.1: The top two eigenvectors extracted in linear PCA (The first row), CHK PCA (The second row), PEK PCA (The third row), LEK PCA (The fourth row) and LEK PCA (The fifth row).

kernel PCA capture richer connection structures. Specifically, as seen from Fig. 4.1(a), the first eigenvector in linear PCA only presents very weak intra-lobe connections in frontal and occipital lobes. In contrast, the first eigenvector obtained by each of the SPD-kernel PCA well captures the intra-lobe connections in all the lobes. Especially, as indicated in Fig. 4.1(c), (e), (g) and (i), there are strong connections at orbitofrontal cortex (ROI index: 8, 19-22), rectus gyrus (23, 24), occipital gyrus (43-48), temporal gyri (53-58), Hippocampus (65-66) and temporal pole (69-72). Respecting the second eigenvector, the eigenvectors obtained by the SPD-kernels PCA (Fig. 4.1(d), (f), (h) and (j)) incorporate both intra-lobe and inter-lobe connections while the eigenvector in linear PCA (Fig. 4.1(b)) mainly captures only intra-lobe connections in occipital lobe; ii) The pre-images obtained when different SPD kernels are used, as seen in Fig. 4.1(c)-(j), are very similar with each other with slight variation. This is expected since they all reflect the underlying manifold structure of SICE matrices. Further exploration of their clinical interpretation will be included in our future work.

4.4.3 Evaluation of the pre-image method using synthetic data

To further investigate the efficacy of the proposed pre-image method, a synthetic data set is specially designed for evaluation. The synthetic data set is used for two purposes: i) It allows the comparison between the recovered pre-image of $\Phi_m(\cdot)$ and the ground truth inverse covariance matrix, which is not available for real rs-fMRI data; ii) By adjusting the parameters used to generate the synthetic data, the behavior of the proposed pre-image method can be demonstrated. The synthetic data are generated by mimicking the following data generation process in practice.

(1) Generate a set of 82 covariance matrices of the size of 90×90, by sampling a Wishart distribution [6] [92]. Let $\mathbf{\Sigma}_i$ $(i = 1, \cdots, 82)$ be the i-th covariance matrix

[6]The Wishart distribution is used as $\mathbf{\Sigma}_i \sim \mathcal{W}_{90}(\mathbf{\Sigma}_0, n)$, where $\mathbf{\Sigma}_0 \in \mathrm{Sym}^+_{90}$ is set as a block-wise covariance matrix for a better illustration of the result, and n is the degree of freedom set as 1000.

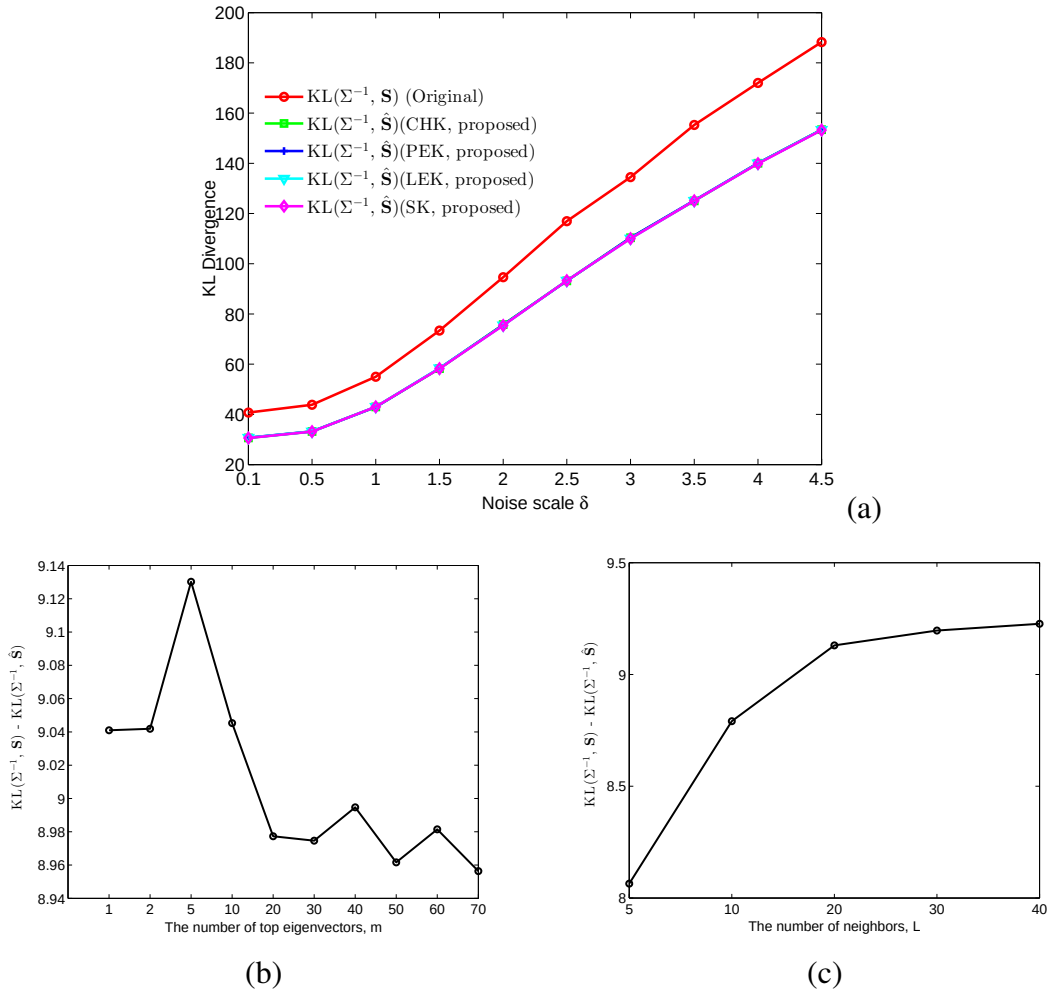


Figure 4.2: The performance of the proposed pre-image method on synthetic data set. (a) The averaged KL divergence between the ground truth inverse covariance matrix $\mathbf{\Sigma}^{-1}$ and the original SICE matrix $\mathbf{S}$ (labeled by 'original') or the pre-images $\hat{\mathbf{S}}$ when four SPD kernels are used (labeled by 'CHK', 'PEK', 'LEK' and 'SK', respectively) at various noise levels with m and L set as 5 and 20, respectively. As indicated, the resulting KL divergence values corresponding to the four SPD kernels are consistently smaller than $KL(\mathbf{\Sigma}^{-1}, \mathbf{S})$ at all noise levels. Moreover, the improvement of $KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$ over $KL(\mathbf{\Sigma}^{-1}, \mathbf{S})$, i.e. $KL(\mathbf{\Sigma}^{-1}, \mathbf{S}) - KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$, becomes more significant with increase of δ. Note that the KL divergence values corresponding to the four kernels are similar and overlapped in the figure; (b) The improvement of the proposed pre-image method (using Stein kernel) with various number of leading eigenvectors m when L is set as 20, and (c) The improvement of the proposed pre-image method (using Stein kernel) with various number of neighbors L when m is set as 5.

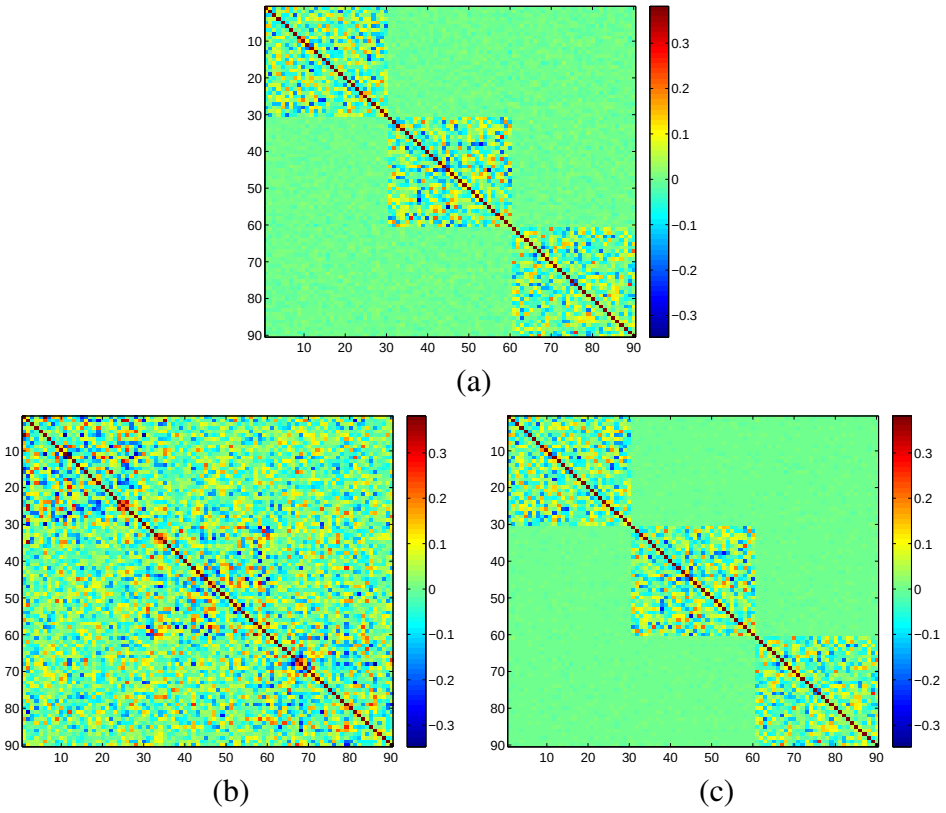


Figure 4.3: Illustration of the result obtained by our proposed pre-image method. (a) shows a ground truth inverse covariance matrix $\mathbf{\Sigma}^{-1}$, (b) plots the original SICE matrix $\mathbf{S}$ and (c) shows the estimated pre-image $\hat{\mathbf{S}}$ of $\Phi_m(\mathbf{S})$. As seen, $\hat{\mathbf{S}}$ is more similar to $\mathbf{\Sigma}^{-1}$ in comparison with $\mathbf{S}$, indicating that the proposed pre-image method brings some kind of denoising effect.

and its inverse $\mathbf{\Sigma}_i^{-1}$ will be used as a ground truth inverse covariance matrix;

(2) A set of 130 vectors are randomly sampled from each normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i)$, where $i = 1, \cdots, 82$;

(3) Gaussian noise is added to each set of 130 vectors to simulate that the data are contaminated. The noise level is denoted by δ;

(4) A sample-based covariance matrix $\mathbf{C}$ is computed by using each set of the (noisy) 130 vectors and 82 covariance matrices are obtained in total. They are denoted as

$\mathbf{C}_1, \mathbf{C}_1, \cdots, \mathbf{C}_{82}$;

(5) Apply the SICE method to each $\mathbf{C}_i$ to obtain the SICE matrix, and they are collectively denoted by $\{\mathbf{S}_i\}_{i=1}^{82}$. These SICE matrices form the synthetic data set. Note that they are affected by the noise added in Step 3.

From the synthetic data set $\{\mathbf{S}_i\}_{i=1}^{82}$, every $\mathbf{S}_i$ is selected in turn as the test data and the remainder are used as the training set. Algorithm 4 is then applied to estimate the pre-image $\hat{\mathbf{S}}_i$ for $\Phi_m(\mathbf{S}_i)$. Then the recovered pre-image $\hat{\mathbf{S}}_i$ and the test data $\mathbf{S}_i$ are compared, respectively, with the ground truth inverse covariance matrix $\mathbf{\Sigma}_i^{-1}$ prepared in Step 1. This is to see whether $\hat{\mathbf{S}}_i$ is really closer to $\mathbf{\Sigma}_i^{-1}$ than $\mathbf{S}_i$. Following the literature [89], we use Kullback-Liebler (KL) divergence to compare $\hat{\mathbf{S}}_i$ (or $\mathbf{S}_i$) with $\mathbf{\Sigma}_i^{-1}$. Given a pair of SPD matrices $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$, KL divergence measures the similarity of two Gaussians $\mathcal{N}(\boldsymbol{\mu}_1, \mathbf{\Sigma}_1)$ and $\mathcal{N}(\boldsymbol{\mu}_2, \mathbf{\Sigma}_2)$. It can be used to measure the similarity between the two SPD matrices by relating them to the covariance matrices and setting the means as zero. KL divergence in our case is expressed as $KL(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = \mathrm{tr}(\mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1) - \log\det(\mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1) - d$, where d is the number of network nodes. It is nonnegative and a smaller divergence indicates that these two matrices are more similar.

The result is shown in Fig. 4.2. As seen in Fig. 4.2(a), $KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$ (averaged over all 82 test cases and with m and L set as 5 and 20, respectively.) is consistently lower than $KL(\mathbf{\Sigma}^{-1}, \mathbf{S})$ for all the different noise levels and the SPD kernels used in the kernel PCA. This result suggests that the obtained pre-image $\hat{\mathbf{S}}$ is closer to the ground truth inverse covariance matrix $\mathbf{\Sigma}^{-1}$ in comparison with the original SICE matrix $\mathbf{S}$. Relating back to the idea that we use to design this experiment, this result shows that the proposed pre-image method indeed works. Also, the improvement of $KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$ over $KL(\mathbf{\Sigma}^{-1}, \mathbf{S})$, i.e. $KL(\mathbf{\Sigma}^{-1}, \mathbf{S}) - KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$, becomes more significant with the increase of the noise level δ introduced in Step 3 of the synthetic data generation process. To demonstrate the result obtained by the proposed pe-image method, an example is

given in Fig. 4.3, where Fig. 4.3(a) shows a ground truth inverse covariance matrix $\mathbf{\Sigma}^{-1}$, Fig. 4.3(b) plots the estimated SICE matrix $\mathbf{S}$ and Fig. 4.3(c) shows the pre-image $\hat{\mathbf{S}}$ of $\Phi_m(\mathbf{S})$. As seen, $\hat{\mathbf{S}}$ is more similar to $\mathbf{\Sigma}^{-1}$ in comparison with $\mathbf{S}$.

As indicated in Algorithm 4, the number of leading eigenvectors m and the number of neighbors L are two important parameters. We evaluate how the performance of the proposed pre-image method will change with these two parameters. Stein kernel is taken as an example. Fig. 4.2(b) and Fig. 4.2(c) show the improvement, i.e. $KL(\mathbf{\Sigma}^{-1}, \mathbf{S}) - KL(\mathbf{\Sigma}^{-1}, \hat{\mathbf{S}})$, of our method with different m and L, respectively. As seen in Fig. 4.2(b), when L is set as constant 20, the improvement first increases with m and then decreases, achieving the highest value when m is five. This is because the first several leading eigenvectors $\mathbf{v}_i$ in $\mathcal{F}$ represent the dominant network structures of the network while the following ones intend to characterize more detailed structures which are vulnerable to noise. As a result, with the increasing value of m, the components often correspond to noise. Therefore, when $m > 5$, noisy components could be included, and this reduces the magnitude of the improvement. At the same time, note that the improvement does consistently hold although its magnitude is reduced. Fig. 4.2(c) shows that, when m is fixed at 5, the improvement with the increase of L becomes saturated when $L = 20$. This is because the constraint of $\sum_{j=1}^{L} w_j = 1$ in $\hat{\mathbf{S}} = \sum_{j=1}^{L} w_j \mathbf{S}_j$ (Eq.(4.6)) imposes the sparsity of w_j, limiting the actual number of neighbors $\mathbf{S}_j$ used to estimate $\hat{\mathbf{S}}$. Based on our experience, a relatively large initial number of L is recommended, e.g. one fourth of the number of training samples, and the constraint of $\sum_{j=1}^{L} w_j = 1$ will implicitly and automatically select a small set of $\mathbf{S}_j$ by setting most w_j as zero.

Table 4.4: The name and lobe of each ROI in Fig. (4.1).

Lobe	ROI index	ROI name	ROI index	ROI name
	1	Frontal_Sup_L	2	Frontal_Sup_R

frontal

	3	Frontal_Sup_Orb_L	4	Frontal_Sup_Orb_R
	5	Frontal_Mid_L	6	Frontal_Mid_R
	7	Frontal_Mid_Orb_L	8	Frontal_Mid_Orb_R
	9	Frontal_Inf_Oper_L	10	Frontal_Inf_Oper_R
	11	Frontal_Inf_Tri_L	12	Frontal_Inf_Tri_R
	13	Frontal_Inf_Orb_L	14	Frontal_Inf_Orb_R
	15	Supp_Motor_Area_L	16	Supp_Motor_Area_R
	17	Olfactory_L	18	Olfactory_R
	19	Frontal_Sup_Medial_L	20	Frontal_Sup_Medial_R
	21	Frontal_Mid_Orb_L	22	Frontal_Mid_Orb_R
	23	Rectus_L	24	Rectus_R
	25	Paracentral_Lobule_L	26	Paracentral_Lobule_R
parietal	27	Parietal_Sup_L	28	Parietal_Sup_R
	29	Parietal_Inf_L	30	Parietal_Inf_R
	31	SupraMarginal_L	32	SupraMarginal_R
	33	Angular_L	34	Angular_R
	35	Precuneus_L	36	Precuneus_R
occipital	37	Calcarine_L	38	Calcarine_R
	39	Cuneus_L	40	Cuneus_R
	41	Lingual_L	42	Lingual_R
	43	Occipital_Sup_L	44	Occipital_Sup_R
	45	Occipital_Mid_L	46	Occipital_Mid_R
	47	Occipital_Inf_L	48	Occipital_Inf_R
	49	Fusiform_L	50	Fusiform_R
temporal	51	Heschl_L	52	Heschl_R
	53	Temporal_Sup_L	54	Temporal_Sup_R
	55	Temporal_Mid_L	56	Temporal_Mid_R

	57	Temporal_Inf_L	58	Temporal_Inf_R
limbic	59	Cingulum_Ant_L	60	Cingulum_Ant_R
	61	Cingulum_Mid_L	62	Cingulum_Mid_R
	63	Cingulum_Post_L	64	Cingulum_Post_R
	65	Hippocampus_L	66	Hippocampus_R
	67	ParaHippocampal_L	68	ParaHippocampal_R
	69	Temporal_Pole_Sup_L	70	Temporal_Pole_Sup_R
	71	Temporal_Pole_Mid_L	72	Temporal_Pole_Mid_R
insula	73	Insula_L	74	Insula_R
sub cortical	75	Amygdala_L	76	Amygdala_R
	77	Caudate_L	78	Caudate_R
	79	Putamen_L	80	Putamen_R
	81	Pallidum_L	82	Pallidum_R
	83	Thalamus_L	84	Thalamus_R
central	85	Precentral_L	86	Precentral_R
	87	Rolandic_Oper_L	88	Rolandic_Oper_R
	89	Postcentral_L	90	Postcentral_R

4.5 Conclusion

Recently, sparse inverse covariance matrix (SICE) has been used as a representation of brain connectivity to classify Alzheimer's disease and normal controls. However, its high dimensionality can adversely affect the classification performance. Taking advantage of the SPD property of SICE matrices, this chapter uses SPD-kernel PCA to extract principal components to obtain a compact representation for classification. this chapter

also proposes a pre-image estimation algorithm, which allows visualization and analysis of the extracted principal connectivity patterns in the input space. The efficacy of the proposed method is verified by extensive experimental study on synthetic data and real rs-fMRI data from the ADNI.

Chapter 5

Sample-adaptive Integration of Multiple SICE Matrices

This chapter proposes a subject-adaptive method to integrate multiple SICE networks as a unified representation for classification. The integration weight is learned adaptively for each subject in order to endow the method with the flexibility in dealing with subject variations. Furthermore, to respect the manifold geometry of SICE networks, Stein kernel is employed to embed the manifold structure into a kernel-induced feature space, which allows a linear integration of SICE networks to be designed. The optimization of the integration weight and the classification of the integrated networks are performed via a sparse representation framework. Through the proposed method, this chapter provides a unified and effective network representation that is transparent to the sparsity level of SICE networks, and can be readily utilized for further medical analysis. Experimental study on ADHD and ADNI data sets demonstrates that the proposed integration method achieves notable improvement of classification performance in comparison with methods using a single sparsity level of SICE networks and other commonly used integration methods, such as Multiple Kernel Learning.

5.1 Motivation

The last chapter and previous studies have shown the great potential of SICE networks for analyzing brain connectivity patterns [45, 71, 62, 60] and diagnosing brain diseases [37, 6, 77, 123]. For each subject, the SICE method naturally produces a set of connectivity networks by specifying different sparsity regularization parameters. Monotonic increase of this parameter would lead to gradually sparser networks. Sparse connectivity networks present the stronger connectivities while dense ones also include weaker connectivities. In this case, at which sparsity level the SICE network should be used for classification becomes a critical issue, because different connectivity patterns could possess different discriminative power. A common practice is to select one sparsity level from this set of networks [60, 6, 77] and ignore other sparsity levels. However, there are at least two drawbacks with this approach: 1) It does not fully exploit the information contained in these ignored sparsity levels; 2) The selection of the most appropriate sparsity level is usually carried out by multi-fold cross-validation, which often has high computational complexity and becomes unreliable when the sample size is small.

We argue that SICE networks with different sparsity levels could provide complementary information that is of great value for classifying the SICE networks. They should be jointly considered in order to improve the discriminative power. Also, this will circumvent the rigid selection of a single sparsity level. To this end, a straightforward method might be to concatenate the features extracted from these SICE networks. However, this method suffers from two limitations: 1) not all the extracted features are sufficiently discriminative; and 2) this method treats each network separately and indiscriminately, failing to integrate them in an inherent and adaptive manner. Another appealing method may be multiple kernel learning (MKL). However, although MKL combines multiple kernels from different channels, it ignores the inter-channel information, as will be explained in Section 5.3.5.

This chapter proposes a learning based framework that integrates a set of SICE networks with the aim of attaining more discriminative power. Our framework has at least four contributions.

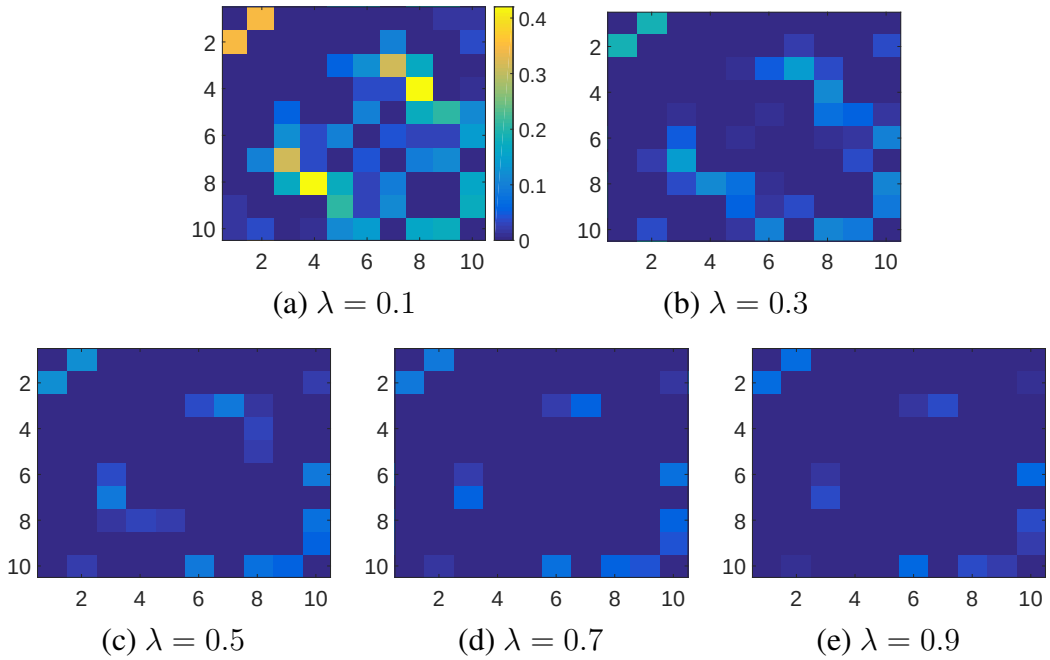
(1) It makes use of the whole spectrum of SICE matrices at different sparsity levels, which not only improves the classification performance, but also circumvents the need of presetting the employed sparsity level.

(2) The proposed framework provides subject-adaptive integration of SICE networks. It is noticed that some sparsity levels that are useful for one subject could become less useful for another due to the variation of subject-specific characteristics, e.g., disease phase, age and gender. In this case, we allow the integration to be subject-adaptive, and achieve this through a sparse representation framework.

(3) Our integration of SICE networks respects the specific geometrical property of SICE matrix. As known, SICE matrices are symmetric position definite (SPD) and form a Riemannian manifold [3, 40], which makes a linear combination of them in the input (Euclidean) space improper. To address this issue, we propose to embed the Riemannian manifold into a reproducing kernel Hilbert space (RKHS), where linear operations become sufficiently good to handle SICE matrices. This embedding is achieved through SPD-matrix based kernels, such as the Stein kernel [88]. Following that, a linear combination of multiple SICE networks is optimized in this kernel induced feature space through the sparse representation framework mentioned above.

(4) By the integration, our learning framework provides a unique, enhanced, and new network representation for each subject. Although the integration takes place in a kernel induced feature space, it is feasible to project the integration result back into the original network space for visualization and further medical analysis. This could help to understand the underlying pathophysiology of brain diseases.

The rest of the chapter is organized as follows: Section 5.2 reviews the SICE algorithm, the properties and measurements of SICE networks and the sparse representation technique. Section 5.3 details the proposed subject-adaptive integration method and discusses several issues regarding the proposed method. Section 5.4 presents the experimental results on ADHD and ADNI rs-fMRI data sets. And Section 5.5 concludes this chapter.



(a) $\lambda = 0.1$ (b) $\lambda = 0.3$

(c) $\lambda = 0.5$ (d) $\lambda = 0.7$ (e) $\lambda = 0.9$

Figure 5.1: An example of 10×10-dimensional SICE networks with different λ values. The colorbar indicates the strength of connectivities. A smaller λ results in a denser connectivity network while a larger λ leads to a sparser network.

5.2 Related Work

5.2.1 Sparse representation

Sparse representation [114] of signals have received intensive attention over the last decade due to its effectiveness in modeling signals and the robustness in dealing with corrupted data. Sparse representation aims to search for the sparsest representation of a signal by using a linear combination of atoms in an overcomplete dictionary. Specifically, given a signal $\mathbf{x} \in \mathbb{R}^n$ and a set of atoms $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_N]$, $\mathbf{d}_i \in \mathbb{R}^n$ as a dictionary, the idea of sparse representation is to reconstruct signal $\mathbf{x}$ by using a few atoms in dictionary $\mathbf{D}$. The sparse representation coefficient of $\mathbf{x}$, denoted as $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_N]^\top$, can be obtained by solving the following problem:

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \quad \| \mathbf{x} - \sum_{i=1}^{N} \alpha_i \mathbf{d}_i \|_2^2 + \gamma \| \boldsymbol{\alpha} \|_1 \tag{5.1}$$

where the first term of Eq.(5.1) is the reconstruction error of $\mathbf{x}$, and the second term is used to control the sparsity of the coefficient $\boldsymbol{\alpha}$. The parameter γ is the tradeoff parameter used to balance the reconstruction error and sparsity. A larger γ leads to a sparser solution. Kernel-based sparse representation (KSR) [23] extends the concept of sparse representation from a vector space to a kernel-induced feature space $\mathcal{F}$. KSR shares the same form of Eq.(5.1) except that both the signal and the atoms become the mapped features in $\mathcal{F}$.

5.3 Proposed Method: SASNI

5.3.1 Problem formulation

As previously mentioned, the SICE representation naturally leads to a set of networks with different sparsity levels. Each sparsity level of the network captures specific con-

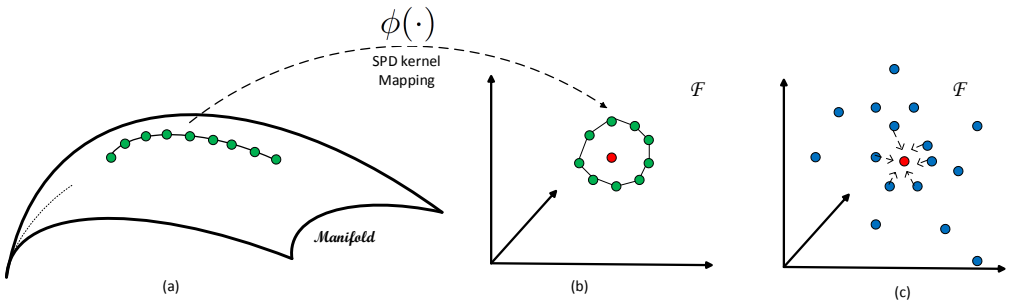


Figure 5.2: Illustration of the proposed SICE network integration method. (a) a set of SICE networks with various sparsity (denoted by green dots) residing on a Riemannian manifold can be obtained for a single subject; (b) the SICE networks are mapped to a kernel-induced feature space $\mathcal{F}$ by using a SPD kernel. Then the mapped networks in $\mathcal{F}$ are convexly combined as a unified representation $\phi(f_{\boldsymbol{\beta}}(\mathbb{X}))$ (denoted by a red dot); (c) the unified representation is sparsely represented by the atoms (denoted by blue dots) in $\mathcal{F}$.

nectivity patterns, which are different from that of other sparsity levels. Therefore, when the final aim is to classify different groups of subjects, such as subjects with ADHD versus healthy controls, multiple sparsity levels of connectivity networks should be jointly considered and integrated. By doing so, the complementary and discriminative information of the connectivity patterns from multiple sparsity levels could be fully explored to boost the classification performance.

To integrate multiple sparsity levels of SICE networks, a natural approach is linear combination. However, as previously mentioned, linearly combining manifold data in a Euclidean space fails to respect the non-linear structure of the Riemannian manifold and could lead to spurious result, such as the swelling effect [3]. Accounting for the manifold structure of SICE networks, we propose a method that conducts such an integration in a kernel-induced feature space $\mathcal{F}$. Firstly, multiple sparsity levels of SICE networks from the same subject are mapped into $\mathcal{F}$ by a non-linear mapping function $\phi(\cdot) : \mathrm{Sym}_d^+ \rightarrow \mathcal{F}$. This $\phi(\cdot)$ mapping brings at least two advantages. 1) The Riemannian manifold geometry has been well considered by using distance functions specially

designed for SPD matrices, such as the Stein divergence; 2) The images of SICE matrices can be linearly processed in the kernel-induced feature space $\mathcal{F}$. After the mapping, we utilize a convex combination to integrate multiple ϕ-mapped SICE networks in $\mathcal{F}$. Convex combination constrains the resulting network to lie in the convex hull of these ϕ-mapped SICE networks, implicitly making the solution better comply to the local distribution of these networks. Specifically, let $\mathbb{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_M\}$, $\mathbf{X}_i \in \mathrm{Sym}_d^+$, denote the SICE networks of one subject at M different sparsity levels and $f(\mathbb{X})$ denote the combined network of $\mathbb{X}$ with implicit function f. The convex combination is carried out in the following manner:

$$\begin{aligned} \Phi &= \sum_{j=1}^{M} \beta_j \phi(\mathbf{X}_j) \\ \text{subject to:} \quad &\sum_{j=1}^{M} \beta_j = 1; \quad \beta_j \geq 0. \end{aligned} \tag{5.2}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_M]^\top$ is the combination coefficient of function f. Instead of applying a uniform $\boldsymbol{\beta}$ to all the subjects, we assign a subject-adaptive $\boldsymbol{\beta}$ for each subject. We argue that the discriminative power of SICE networks at different sparsity levels may not necessarily be uniform across all the subjects. This can be intuitively understood by the following considerations. 1) Although sharing the same class label, the patient subjects could be experiencing different stages of the disease. As a result, disease-induced alterations in the brain networks will be different across subjects. 2) The brain networks may demonstrate differences across subjects due to different age, gender, etc. Thus, we propose to adaptively optimize $\boldsymbol{\beta}$ for each subject to handle such variation as follows.

To solve $\boldsymbol{\beta}$, this chapter extends the kernel-based sparse representation (KSR) [23]. The original KSR is used to obtain the sparse representation coefficient $\boldsymbol{\alpha}$ only. In this chapter, we develop it to optimize the integration coefficients $\boldsymbol{\beta}$ and the sparse

representation coefficient $\boldsymbol{\alpha}$ simultaneously. The proposed method is called sample-adaptive SICE network integration (SASNI).

Let $\mathbb{D} = \{\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_N\}$, $\mathbf{D}_j \in \mathrm{Sym}_d^+$, denote a dictionary with N SPD matrices. Note that $\mathbf{X}_i$ and $\mathbf{D}_j$ are both SICE matrices, however, $\mathbf{X}_i$ denotes the i-th SICE matrix of a test subject while $\mathbf{D}_j$ denotes the j-th atom of the dictionary, which is formed by SICE matrices from training subjects in this chapter. The objective function of SASNI is formulated as follows.

$$\begin{aligned} \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^N,\boldsymbol{\beta}} \quad & \| \Phi - \sum_{i=1}^{N} \alpha_i \phi(\mathbf{D}_i) \|_2^2 + \gamma \| \boldsymbol{\alpha} \|_1 \\ \text{subject to:} \quad & \sum_{j=1}^{M} \beta_j = 1; \quad \beta_j \geq 0. \end{aligned} \tag{5.3}$$

The first term of Eq.(5.3) is the reconstruction error of Φ, and the second term is used to control the sparsity of $\boldsymbol{\alpha}$.

Figure 5.2 illustrates the proposed method: (a) a set of SICE networks with various sparsity (denoted by green dots) residing on a Riemannian manifold can be obtained for a single subject; (b) the SICE networks are mapped to a kernel-induced feature space $\mathcal{F}$ by using SPD kernels. Then the mapped networks in $\mathcal{F}$ are convexly combined as a unified representation (denoted by a red dot); (c) the unified representation is sparsely represented by the atoms (denoted by blue dots) in $\mathcal{F}$. How to solve the combination coefficient $\boldsymbol{\beta}$ and the sparse representation coefficient $\boldsymbol{\alpha}$ and how they can be used for classification will be elaborated in the following parts.

5.3.2 Solving the optimization problem

As known in kernel methods, the kernel mapping ϕ is usually too complicated to be explicitly computed. Therefore, we need to bypass the mapping ϕ in Eq.(5.3). With the kernel property of $k(\mathbf{A}, \mathbf{B}) = \phi(\mathbf{A})^\top \phi(\mathbf{B})$, the first term in Eq.(5.3) can be expanded

as:

$$\begin{aligned} \| \Phi - \sum_{i=1}^{N} \alpha_i \phi(\mathbf{D}_i) \|_2^2 &= \Phi^\top \Phi - 2 \sum_{i=1}^{N} \alpha_i \Phi^\top \phi(\mathbf{D}_i) + \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j \phi(\mathbf{D}_i)^\top \phi(\mathbf{D}_j) \\ &\text{(Substitute } \Phi \text{ with Eq.(5.2))} \\ &= \boldsymbol{\beta}^\top \mathcal{K}(\mathbb{X}, \mathbb{X}) \boldsymbol{\beta} - 2 \boldsymbol{\alpha}^\top \mathcal{K}(\mathbb{D}, \mathbb{X}) \boldsymbol{\beta} + \boldsymbol{\alpha}^\top \mathcal{K}(\mathbb{D}, \mathbb{D}) \boldsymbol{\alpha} \end{aligned} \tag{5.4}$$

where $\mathcal{K}(\mathbb{X}, \mathbb{X}) = [k(\mathbf{X}_i, \mathbf{X}_j)]_{M \times M}$, $\mathcal{K}(\mathbb{D}, \mathbb{X}) = [k(\mathbf{D}_i, \mathbf{X}_j)]_{N \times M}$ and $\mathcal{K}(\mathbb{D}, \mathbb{D}) = [k(\mathbf{D}_i, \mathbf{D}_j)]_{N \times N}$. Note that, as shown in Eq.(5.4), the objective in Eq.(5.3) is a function of both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. It can be proved that the objective function is jointly convex on both of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. To find the optima for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we devise an iterative procedure that monotonically decreases the objective by alternately optimizing one variable at a time while keeping the other fixed. Specifically, at the t-th iteration, we first optimize the objective in Eq.(5.3) over $\boldsymbol{\alpha}$ with $\boldsymbol{\beta}$ fixed. In this case, $\boldsymbol{\beta}^\top \mathcal{K}(\mathbb{X}, \mathbb{X}) \boldsymbol{\beta}$ reduces to a constant and Eq.(5.3) can be rewritten as follows:

$$\boldsymbol{\alpha}^{(t)} = \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \quad - 2 \boldsymbol{\alpha}^\top \mathcal{K}(\mathbb{D}, \mathbb{X}) \boldsymbol{\beta}^{(t)} + \boldsymbol{\alpha}^\top \mathcal{K}(\mathbb{D}, \mathbb{D}) \boldsymbol{\alpha} + \gamma \| \boldsymbol{\alpha} \|_1 \tag{5.5}$$

The objective of Eq.(5.5) is a quadratic function and convex with respect to $\boldsymbol{\alpha}$. This optimization problem can be efficiently solved via existing packages, such as SPAM-S [56]. After obtaining $\boldsymbol{\alpha}^{(t)}$, $\boldsymbol{\beta}$ can be updated in a similar manner by optimizing the following problem:

$$\begin{aligned} \boldsymbol{\beta}^{(t)} = \arg \min_{\boldsymbol{\beta}} \quad & \boldsymbol{\beta}^\top \mathcal{K}(\mathbb{X}, \mathbb{X}) \boldsymbol{\beta} - 2 {\boldsymbol{\alpha}^{(t)}}^\top \mathcal{K}(\mathbb{D}, \mathbb{X}) \boldsymbol{\beta} \\ \text{subject to:} \quad & \sum_{j=1}^{M} \beta_j = 1; \quad \beta_j \geq 0. \end{aligned} \tag{5.6}$$

This optimization problem can be solved by the reduced gradient descent method as in [72] to handle the convex constrain. $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are optimized alternately until a stop-

ping criterion is satisfied. There are two commonly used stopping criteria: i) a predefined threshold on the difference of consecutive objective function values is met; and ii) the maximum iteration number is reached.

Note that as the original KSR [23], there is no training stage in the proposed method, and the training subjects are used as the atoms in the dictionary $\mathbb{D}$. *α and β are optimized individually for each test subject, so they are adaptive for different test subjects.* This subject-adaptivity of β enhances the flexibility of SASNI to integrate multiple networks across subjects, and in turn improves the effectiveness of SASNI.

Table 5.1: Summary of the ADHD-200 data set.

Training Data Set							
	KKI	NeuroImage	NYU	OHSU	Peking	Pittsburgh	WashU
# sub	83	48	216	79	194	89	59
Control	61	23	98	42	116	89	59
ADHD	22	25	118	37	78	0	0
Age	8-13	11-22	7-18	7-12	8-17	10-20	7-22
Male	46	31	140	43	144	46	32
Female	37	17	76	36	50	43	27
Test Data Set							
	KKI	NeuroImage	NYU	OHSU	Peking	Pittsburgh	Brown
# sub	11	25	41	34	51	9	26
Control	8	14	12	28	27	5	N.A.
ADHD	3	11	29	6	24	4	N.A.
Age	8-12	13-26	7-17	7-12	8-15	14-17	8-18
Male	10	12	28	17	32	7	9
Female	1	13	13	17	19	2	17

5.3.3 Classification based on SASNI

With the obtained sparse representation coefficient α, a test sample can be classified in multiple manners. If the atoms in the dictionary are not associated with class labels, the sparse representation coefficient can be treated as a feature vector, and a classifier, such as support vector machine (SVM), can be trained with the feature vectors of training

samples. The class label of a test sample is then predicted by feeding its feature vector to the trained classifier. If the atoms in the dictionary are associated with class labels, i.e., the atoms are just the training samples, the sparse representation coefficient can be directly used to classify a test sample through calculating class-specific residue. This approach is denoted as sparse representation based classification (SRC) in [114] and also used in this chapter. SRC takes the assumption that the samples from the same class do lie in a subspace. In this case, any new (test) sample from the same class shall approximately lie in the span of the training samples and can therefore be sufficiently represented by the training samples from the same class. Then the class-specific reconstruction residue can be calculated by using the sparse representation coefficients and the training samples associated with each class. Eventually, the test sample is assigned to the class with the minimal class-specific reconstruction residue. SRC has been used in some generic image classification tasks, such as face recognition [114] and object categorization [33, 52], and has demonstrated its effectiveness and high efficiency.

Specifically, in this chapter, once the combination coefficients $\boldsymbol{\beta}$ and the sparse representation coefficient $\boldsymbol{\alpha}$ are obtained by optimizing Eq.(5.3), a new (test) sample $\mathbb{X}$ will be assigned to the class with the minimum reconstruction residue. The reconstruction residue of Φ for class i can be computed as follows:

$$\begin{aligned}\epsilon_i(\Phi) =& \| \Phi - \sum_{j=1}^{N} \alpha_j \delta(l(j) - i)\phi(\mathbf{D}_j) \|_2^2 \\ =& \boldsymbol{\beta}^\top \mathcal{K}(\mathbb{X}, \mathbb{X})\boldsymbol{\beta} - 2\tilde{\boldsymbol{\alpha}}_{\boldsymbol{i}}^\top \mathcal{K}(\mathbb{D}, \mathbb{X})\boldsymbol{\beta} + \tilde{\boldsymbol{\alpha}}_{\boldsymbol{i}}^\top \mathcal{K}(\mathbb{D}, \mathbb{D})\tilde{\boldsymbol{\alpha}}_{\boldsymbol{i}} \, .\end{aligned} \tag{5.7}$$

where $l(j)$ is the label of the j-th atom $\mathbf{D}_j$, $\delta(k)$ is the Kronecker delta function, which is one when k is zero, and zero otherwise, and $\tilde{\boldsymbol{\alpha}}_{\boldsymbol{i}} = \boldsymbol{\alpha} \odot [\delta(l(1) - i), \delta(l(2) - i), \cdots, \delta(l(N) - i)]^\top$, with $\odot$ indicating element-wise product. Based on Eq. (5.7),

the final label of $\mathbb{X}$ is determined as follows:

$$l(\mathbb{X}) = \arg\min_i \epsilon_i(\Phi) \tag{5.8}$$

Algorithm 4 outlines the proposed method SASNI.

Algorithm 4 Proposed subject-adaptive SICE network integration (SASNI).

Input: A training sample set $\mathbb{D} = \{\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_N\}$; A test sample $\mathbb{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_M\}$; γ, the maximum iteration number T and the threshold τ.
Output: $l(\mathbb{X})$, $\boldsymbol{\beta}^*$, $\boldsymbol{\alpha}^*$;

1: Initialize $\boldsymbol{\beta}_0$, $\boldsymbol{\alpha}_0$, $t = 0$, $Obj_0 = inf$;
2: **repeat**
3: $t = t + 1$;
4: Update $\boldsymbol{\beta}^{(t)}$ according to Eq.(5.6) with $\boldsymbol{\alpha}^{(t-1)}$;
5: Update $\boldsymbol{\alpha}^{(t)}$ according to Eq.(5.5) with $\boldsymbol{\beta}^{(t)}$;
6: Update Obj_t according to Eq.(5.3);
7: **until** $t \geq T$ or $\mid Obj_{t-1} - Obj_t \mid \leq \tau$;
8: **return** $l(\mathbb{X}) = \arg\min_i \epsilon_i(\mathbb{X})$,
$\boldsymbol{\beta}^* = \boldsymbol{\beta}^{(t)}$, $\boldsymbol{\alpha}^* = \boldsymbol{\alpha}^{(t)}$.

5.3.4 Projection back to the original space

The integrated network provides a unified network representation for each subject and therefore it is worthy to visualize for future analysis. However, as shown above, the integration takes place in a kernel-induced feature space. As a result, it needs to be projected back into the original space. For this purpose, we use a kernel pre-image estimation method to recover the integrated network in the original input space. The idea is to model the pre-image, denoted as $\hat{\mathbf{S}}$, of the integrated network Φ by a linear combination of its neighboring SICE matrices in Sym_d^+, i.e., $\hat{\mathbf{S}} = \sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j$, where Ω denotes a set of neighboring SICE networks and w_j denotes the combination weight. Note that the neighboring SICE networks can be found by measuring similarities between Φ and $\phi(\mathbf{S}_i)$ in the kernel-induced feature space. The optimal $\mathbf{w}$ can be obtained by solving

the following optimization problem using gradient descent based algorithms.

$$\mathbf{w}^* = \arg \min_{\mathbf{w}\geq 0;\ \mathbf{w}^\top \mathbf{1}=1} \parallel \Phi - \phi \left(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j \right) \parallel_2^2 \tag{5.9}$$

The objective function in Eq.(5.9) can be further reorganized into terms of kernel matrices as in Eq.(5.10).

$$\begin{aligned} &\parallel \Phi - \phi \left(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j \right) \parallel_2^2 \\ =&\boldsymbol{\beta}^\top \mathcal{K}(\mathbb{X}, \mathbb{X}) \boldsymbol{\beta} - 2\mathcal{K} \left(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j, \mathbb{X} \right) \boldsymbol{\beta} + k \left(\sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}, \sum_{\mathbf{S}_j \in \Omega} w_j \mathbf{S}_j \right) \end{aligned} \tag{5.10}$$

5.3.5 Discussion

Comparison between MKL and SASNI

Multiple kernel learning (MKL) has been regarded as a promising technique for integrating multiple data sources [25]. It can also be used to integrate multiple sparsity levels of brain networks. Here we discuss the similarity and difference between MKL and the proposed SASNI. The idea of MKL is to search for an integration of base kernel functions that maximizes a generalized performance measure, such as structural risk minimization [25]. Formally, let $\mathbb{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_M\}$ denote M data sources. MKL uses the concatenation of mapped features as the integrated feature representation:

$$\Phi_{\mathbb{X}} = [\sqrt{\omega_1}\phi(\mathbf{X}_1), \sqrt{\omega_2}\phi(\mathbf{X}_2), \cdots, \sqrt{\omega_M}\phi(\mathbf{X}_M)]^\top$$

where $\boldsymbol{\omega} = [\omega_1, \omega_2, \cdots, \omega_M]^\top$ is the integration coefficient. The dot product in the integrated feature space gives the integrated kernel:

$$\begin{aligned}
&\langle \Phi_{\mathbb{X}^p}, \Phi_{\mathbb{X}^q} \rangle \\
=&[\sqrt{\omega_1}\phi(\mathbf{X}_1^p), \sqrt{\omega_2}\phi(\mathbf{X}_2^p), \cdots, \sqrt{\omega_M}\phi(\mathbf{X}_M^p)] \cdot [\sqrt{\omega_1}\phi(\mathbf{X}_1^q), \sqrt{\omega_2}\phi(\mathbf{X}_2^q), \cdots, \sqrt{\omega_M}\phi(\mathbf{X}_M^q)]^\top \\
=&\sum_{j=1}^{M} \omega_j k(\mathbf{X}_j^p, \mathbf{X}_j^q)
\end{aligned} \tag{5.11}$$

In contrast, the integration of SASNI is carried out in the following manner:

$$\Phi_{\mathbb{X}} = \sum_{j=1}^{M} \beta_j \phi(\mathbf{X}_j)$$

and the dot product in the integrated feature space is:

$$\langle \Phi_{\mathbb{X}^p}, \Phi_{\mathbb{X}^q} \rangle = \sum_{i=1}^{M} \sum_{j=1}^{M} \beta_i \beta_j k(\mathbf{X}_i^p, \mathbf{X}_j^q) \tag{5.12}$$

By cross-referring Eq.(5.11) and Eq.(5.12), we can identify that:

(1) The similarity of MKL and SASNI is that both of them adopt multiple data sources in a kernel-induced feature space $\mathcal{F}$.

(2) The difference between MKL and SASNI is that MKL only considers the similarity between samples from the same data source, i.e. $\mathbf{X}_j^p$ and $\mathbf{X}_j^q$, $j \in [1, M]$. In contrast, SASNI explores all pairs of data sources, i.e. X_i^p and X_j^q, $i, j \in [1, M]$ to measure the similarity between two samples. Thus SASNI admits more flexibility to incorporate cross-source information.

At the same time, note that the proposed method specially caters for brain network data where cross-source data can be compared, and MKL is a more general method for

combining different data sources.

Convergence Analysis and Computational Complexity

As outlined in Algorithm 4, the optimization problem defined in Eq.(5.3) is solved by a commonly used alternate optimization strategy which alternately minimizes the objective function with respect to one of the two variables, i.e. the combination coefficient $\boldsymbol{\beta}$ and the sparse representation coefficient $\boldsymbol{\alpha}$. Considering the two facts 1) these two variables are iteratively optimized to decrease the same objective function monotonically; and 2) the objective function is lower bounded by zero, the optimization problem defined in Eq.(5.3) is guaranteed to converge. Also, it can be proved that the objective function is jointly convex over both $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$. Therefore, the final solution is guaranteed to be a global optimum. The experimental results show that employing this optimization strategy has already been able to achieve promising performance.

The main computational cost of the proposed method is twofold: kernel computation and optimization. As indicated in Eq.(5.4), $\mathcal{K}(\mathbb{X},\mathbb{X})$, $\mathcal{K}(\mathbb{D},\mathbb{X})$ and $\mathcal{K}(\mathbb{D},\mathbb{D})$ can be precomputed before the optimization. The time complexity of computing different kernel functions over a pair of $d \times d$ SICE matrices is listed in Table 2.1. It can be seen that when the Stein kernel (SK) is used, the time complexity involved in calculating the kernel matrices $\mathcal{K}(\mathbb{X},\mathbb{X})$, $\mathcal{K}(\mathbb{D},\mathbb{X})$ and $\mathcal{K}(\mathbb{D},\mathbb{D})$ is $\mathcal{O}(M^2d^{2.373})$, $\mathcal{O}(MNd^{2.373})$ and $\mathcal{O}(N^2d^{2.373})$, respectively, where M indicates the number of sparsity levels of connectivity networks and N indicates the number of atoms used to reconstruct $\mathbb{X}$. In terms of the optimization, either $\boldsymbol{\beta}$ or $\boldsymbol{\alpha}$ is obtained by solving a quadratic programming (QP) problem. Let $\mathcal{O}\left(QP\left(n\right)\right)$ denote the computational complexity to solve a convex QP problem with n variables and T denote the number of iterations performed in the optimization process. The optimization complexity can be expressed as $\mathcal{O}\left(T\left(QP\left(M\right)+QP\left(N\right)\right)\right)$. Once $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are optimized, the class label of $\mathbb{X}$ can be quickly assigned according to Eq.(5.7) and Eq.(5.8). Therefore, the overall time com-

plexity of SASNI for each test subject is $\mathcal{O}\Big(T\left(QP\left(M\right)+QP\left(N\right)\right)+\left(M^2+MN+N^2\right)\cdot d^{2.373}\Big)$. In our experiment, a single subject can be classified in less than one second on a desktop computer with 3.6 GHz CPU and 32 GB memory.

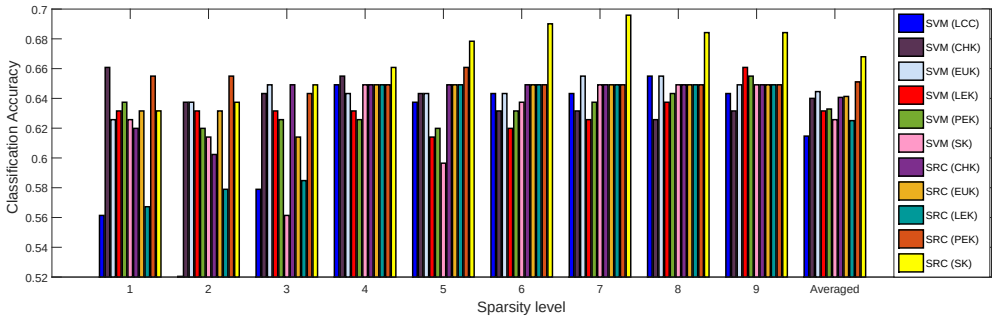


Figure 5.3: Comparison of classification performance between different methods using a single sparsity level of SICE network on ADHD-200 data set. LCC indicates that local clustering coefficient is extracted for classification. CHK, EUK, LEK, PEK, SK denote the five SPD kernels summarized in Table 2.1.

Non-subject-adaptive variants of SASNI

In SASNI, the integration coefficient $\boldsymbol{\beta}$ is adaptively assigned to each subject to handle the variation of the utility of different sparsity levels across subjects. Then a question arises naturally: how will the classification performance be if a uniform $\boldsymbol{\beta}$ is applied for all the subjects to integrate multiple SICE networks at different sparsity levels? To answer this question, we explore two non-subject-adaptive integration variants and conduct a comparison between them and the proposed subject-adaptive counterpart in our experiment (see Section 5.4.2). The first variant is to treat each sparsity level equally and set each entry of $\boldsymbol{\beta}$ to $1/M$, where M is the total number of sparsity levels to integrate. This method, called 'MeanC', can be intuitively understood as using the geometrical center of the networks in the kernel-induced feature space $\mathcal{F}$ as the representation of the network set. The second method is to learn a fixed $\boldsymbol{\beta}$ based on the training data set. To do this, the training data set is split into two folds, one fold as dictionary and the

other fold as the validation set. A fixed $\boldsymbol{\beta}$ is learned in a similar manner as introduced in Section 5.3. The only difference is the reconstruction error defined in Eq. (5.3) is calculated over the subjects in the validation set only. Therefore, the objective function becomes:

$$\begin{aligned} \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^N, \boldsymbol{\beta}} \quad & \sum_{p=1}^{P} \parallel \phi(f_{\boldsymbol{\beta}}(\mathbb{X}^p)) - \sum_{i=1}^{N} \alpha_i \phi(\mathbf{D}_i) \parallel_2^2 + \gamma \parallel \boldsymbol{\alpha} \parallel_1 \\ \text{subject to:} \quad & \sum_{j=1}^{M} \beta_j = 1; \quad \beta_j \geq 0. \end{aligned} \tag{5.13}$$

where P denotes the total number of subjects in the validation set. Eq. (5.13) can be solved in the same way as introduced in Section 5.3. Once a fixed $\boldsymbol{\beta}$ is learned, it will be applied to all the subjects in the test sets to perform classification. This method is called 'FixedTrainC'.

5.4 Experimental Study

5.4.1 Data preprocessing and experimental settings

Two rs-fMRI data sets are used to verify the effectiveness of the proposed method SAS-NI. One data set is ADHD-200 provided by the Neuro Bureau for differentiating Attention Deficit Hyperactivity Disorder (ADHD) from healthy control subjects. ADHD-200 consists of 768 training subjects and 197 test subjects[1] collected from eight independent imaging sites. The summary of this data set is provided in Table 5.1. The rs-fMRI data are processed with Athena pipeline. Specifically, the first four echo-planar imaging (EPI) volumes are removed for signal equilibrium and then slice timing, orientation and motion correction are performed. Each rs-fMRI image is co-registered to T1 image and

[1]The labels of 26 subjects from Brown University in the test set are not released yet. They are not included in our performance evaluation.

warped into MNI space at $4 \times 4 \times 4\ mm^3$ resolution. The time series of 90 brain nodes in gray matter are extracted from the preprocessed data using the automated anatomical labeling (AAL) [96] atlas. Detailed preprocessing descriptions and the processed time series are available at Neuro Bureau website[2].

The other data set is ADNI data set downloaded from the following website `http://adni.loni.usc.edu` with the aim of identifying Mild Cognitive Impairment (MCI), which is very early stage of Alzheimer's disease, from healthy controls. There are 38 healthy controls and 44 MCIs. The data are acquired on a 3 Tesla (Philips) scanner with TR/TE set as 3000/30 ms and flip angle of 80°. Each series has 140 volumes, and each volume consists of 48 slices of 64×64 dimensional image matrices at $3.31 \times 3.31 \times 3.31\ mm^3$ resolution. The preprocessing is carried out using SPM8[3] and DPARSFA [9]. The first 10 volumes of each series are discarded for signal equilibrium. Similar with ADHD-200, slice timing, head motion correction and MNI space normalization are performed. Participants with too much head motion are excluded. The normalized brain images are warped into AAL atlas to obtain 90 ROIs as nodes. The ROI mean time series are extracted by averaging the time series from all voxels within each ROI and then band-pass filtered to obtain the most discriminative frequency band as in [112].

For both of the ADHD-200 and ADNI data sets, the functional connectivity networks are obtained by the SICE method using SLEP [54]. Nine sparsity levels of SICE matrices are obtained for each subject by setting $\lambda = [0.1 : 0.1 : 0.9]$. For the ADHD-200 data set, the predefined training/test sets are used while a leave-one-out procedure is used for ADNI data set to make full use of the limited subjects. For both of the data sets, the dictionary is made up by all the SICE matrices of the training subjects. The parameters used in the classification tasks of these two data sets, including θ in the SPD kernels, γ, and the regularization parameter of SVM are tuned by using five-fold

[2] `http://neurobureau.projects.nitrc.org/ADHD200/Introduction.html`
[3] `http://www.fil.ion.ucl.ac.uk/spm/software/`

cross-validation on the training set.

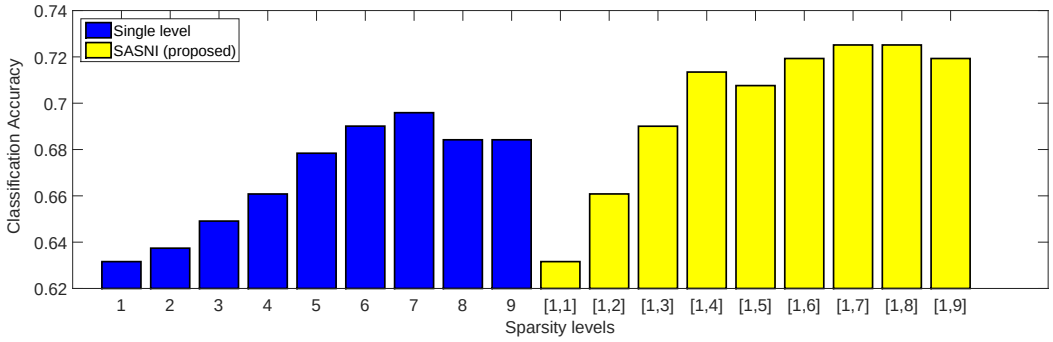


Figure 5.4: Comparison of classification performance between single sparsity level SRC and the proposed SASNI on ADHD-200 data set.

5.4.2 Experimental result

This experiment consists of the following eight parts:

(1) Evaluation of using a single sparsity level on ADHD-200 data set;

(2) Evaluation of the proposed SASNI on ADHD-200 data set;

(3) Comparison with other state-of-the-art methods on ADHD-200 data set;

(4) Comparison with other integration schemes on ADHD-200 data set;

(5) Comparison with non-subject-adaptive integration variants on ADHD-200 data set;

(6) Evaluation of the proposed SASNI on ADNI data set.

(7) Visualization of the integrated brain networks;

(8) Convergence Evaluation.

Evaluation of using a single sparsity level on ADHD-200 data set

Before applying the proposed method to integrating different sparsity levels of brain networks, we first evaluate the performance of a single sparsity level of brain network. In this case, the brain network of one subject is represented as a single SPD matrix, which can be used to train a classifier in multiple manners. The following three common manners are evaluated in this experiment. A straightforward way is to extract graphical features from the network, such as the local clustering coefficient (LCC) feature in [112], and then train SVM with these features. LCC, as a measure of local neighborhood connectivity for a node, is defined as the ratio of the number of existing edges between the neighbors of the node and the number of possible connections between these neighbors [42]. In this case, LCC can map a network, represented by a $d \times d$ adjacency matrix, to a d-dimensional vector, where d is the number of nodes in the network. The second manner is employing one of the five SPD kernels in Table 2.1 to directly evaluate the similarity between SICE matrices and adopt SVM classifier with these kernels to perform classification. The third manner is using SRC instead of SVM as the classifier with the same SPD kernel. Figure 5.3 shows the classification results of these methods on each of nine sparsity levels of SICE networks. As seen, the feature of LCC (labeled by 'SVM (LCC)') produces poor classification with most sparsity levels. When SPD kernels are used, either with SVM or SRC, the classification performance can be improved. In particular, the highest accuracy is achieved by SRC with the Stein kernel (SK), reaching over 69% on the 7th sparsity level.

This demonstrates that: 1) graphical feature of LCC does not sufficiently convey the discriminative information contained in SICE matrices; 2) SPD kernels can achieve reasonably good classification performance by considering the manifold property; 3) using SRC as the classifier and SK as the SPD kernel admits promising classification performance. The good performance verifies the effectiveness of SRC when dealing with brain network classification.

It is also worth noting that the discriminative power of different sparsity levels are not same. The sparser (4th-9th) levels are generally more discriminative than denser (1st-3rd) ones. This consolidates our motivation that combining the complementary information of different sparsity levels could benefit the classification performance.

Evaluation of the proposed SASNI on ADHD-200 data set

SRC with SK achieves the best classification performance when a single sparsity level is used. It is considered as a baseline and compared to the proposed SASNI method. In this experiment, we would like to investigate whether the classification performance can be improved by integrating multiple sparsity levels via SASNI. Note that, SK is also used as the SPD kernel in SASNI. As seen in Figure 5.4, the yellow bars indicate the classification performance of SASNI when different sparsity levels of brain networks are integrated. The tick '$[1, n]$' on the x-axis means that the n densest levels of brain networks are integrated. Compared with SRC (with SK) using a single sparsity level, SASNI can consistently boost the classification performance for all integration settings. When the top four dense levels are integrated, SASNI achieves an accuracy more than 71%. When the top seven dense levels of SICE matrices are integrated, SASNI achieves an accuracy of over 72.5%, obtaining an improvement of three percentage points over SRC with the best single sparsity level.

This demonstrates that integration of multiple sparsity levels could attain more discriminative power and in turn improve the classification performance. As for which sparsity levels to be integrated, they can be selected by cross-validation on the training data set. An alternative way is just integrating all the sparsity levels since, as seen in Figure 5.4, the performance of SASNI is insensitive to the combination range when more than six sparsity levels are integrated.

Comparison with the state-of-the-art methods on ADHD-200 data set

To the best of our knowledge, in the literature, the highest classification accuracy on ADHD-200 data set is reported in [14]. That work computed the pairwise correlation of the time series between brain voxels to model the brain network and extracted multiple kinds of features from the network. Then a PCA-LDA based classifier is trained with the extracted features. Table 5.2 provides the classification performance obtained in the literature [14, 15] and by our proposed SASNI on the whole test set from all imaging sites. As seen, both the proposed SASNI and [14] outperform [15] by a large margin in terms of the overall classification performance. Our proposed SASNI method further exceeds [14] by about three percentage points on the overall test set. In particular, SASNI achieves better performance on OHSU and Peking test sets and perform equally on NeuroImage and NYU test sets in comparison with [14]. On the KKI and Pittsburgh test sets, the performance of SASNI is worse than that of [14]. Note that the number of subjects in KKI and Pittsburgh test sets is only 11 and 9, respectively. The absolute difference is only 1$\sim$2 subjects. On other larger test sets, SASNI consistently achieve better or equal performance in comparison with [14].

Table 5.2: Comparison of classification performance (in%) between the-state-of-the-art methods and the proposed SASNI on ADHD-200 data set.

	# sub	[15]	[14]	SASNI
OHSU	34	**82.4**	73.5	79.4
Peking	51	58.8	62.7	**74.5**
NeuroImage	25	48.0	72	72
NYU	41	-	70.7	70.7
KKI	11	54.6	**72.7**	63.6
Pittsburgh	9	-	**77.8**	55.6
Overall	171	62.8	69.6	**72.5**

$\star$The classification performance of [15] on NYU and Pittsburgh data sets are not reported.

Comparison with other integration schemes on ADHD-200 data set

An experiment is carried out to compare the classification performance of the proposed SASNI method with other integration methods on ADHD-200 data set. These include MKL and a straightforward concatenation of LCC features from different sparsity levels (denoted as LCC in Figure 5.5). As reported in Figure 5.5, SASNI consistently outperforms both LCC feature concatenation method and five MKL methods (using each of the five SPD kernel functions in turn) once the top three or more sparsity levels are integrated.

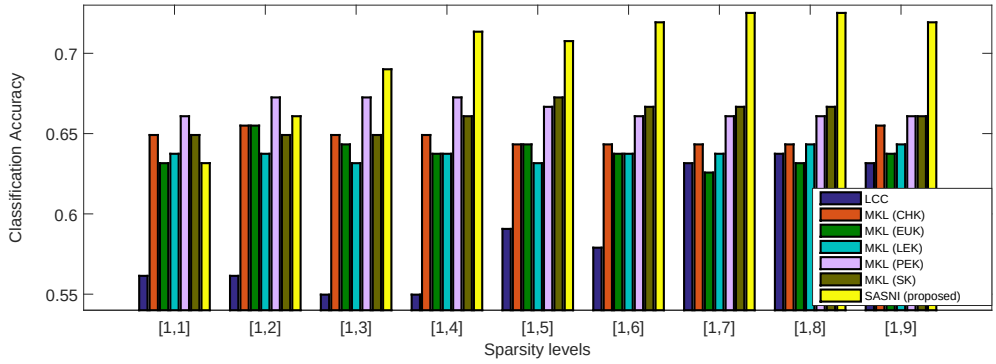


Figure 5.5: Comparison of classification performance between different integration schemes on ADHD-200 data set.

Comparison with non-subject-adaptive integration variants on ADHD-200 data set

In this experiment, the proposed SASNI is compared with two non-subject-adaptive variants of SASNI, i.e. 'MeanC' and 'FixedTrainC' introduced in Section 5.3.5, to investigate the effectiveness of subject-adaptive integration. For 'FixedTrainC', the training data set is split into two equal-sized folds as the dictionary and the validation set, respectively. And the partition of the training data set is repeated 10 times to accumulate statistics. The averaged classification performance is reported in Figure 5.6 in comparison with that of the 'MeanC' and the proposed SASNI method. As seen, the

SASNI method achieves the best classification performance in comparison with the two non-subject-adaptive integration methods. This verifies the advantage of the subject-adaptive mechanism. Note that each of the non-subject-adaptive integration methods outperforms SRC using the best single sparsity level, which indicates that the improved performance of the proposed SASNI method is attributed to both integration of multiple sparsity levels and the subject-adaptive mechanism.

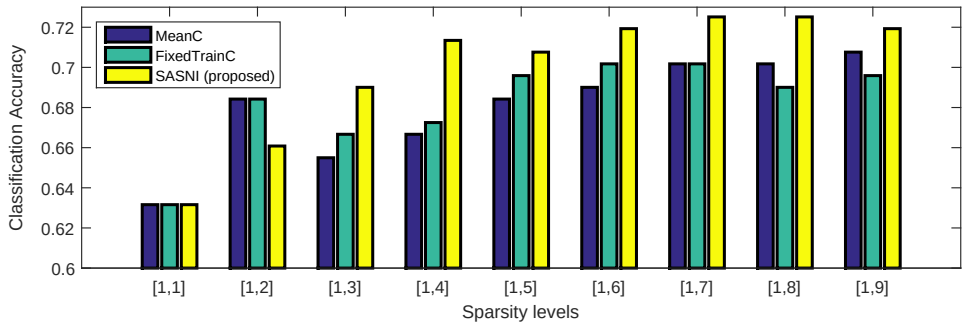


Figure 5.6: Comparison between non-subject-adaptive integration methods and the proposed SASNI on ADHD-200 data set. 'MeanC' indicates average combination while 'FixedTrainC' indicates that a set of fixed integration coefficients is learned with the training data set and uniformly applied to all test subjects.

Evaluation of the proposed method on ADNI data set

As previously mentioned, a leave-one-out procedure is used in the classification on ADNI data set, and the averaged accuracy is reported. As seen in Figure 5.7, the superiority of the proposed SASNI over a single sparsity level based SRC is confirmed again on ADNI data set. Specifically, when all the nine sparsity levels are integrated, SASNI outperforms the best single sparsity level based SRC by a large margin of over five percentage points.

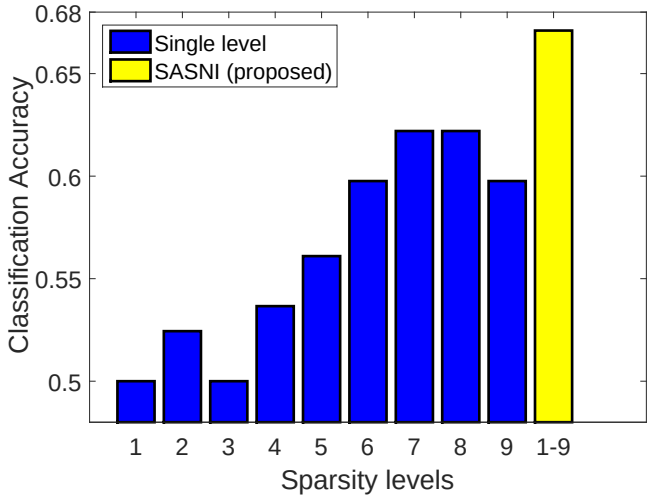


Figure 5.7: Comparison of classification performance between a single sparsity level based SRC and the proposed SASNI on ADNI data set.

Visualization of the integrated brain networks

An example of recovered pre-image is shown in Figure 5.8. The 90 ROIs of AAL atlas are grouped into eight lobes according to the anatomical structure. As seen, evident block-wise structures are presented. This is expected since the ROIs in the same lobe have higher chances to be anatomically and functionally connected. Also, most of the functional abnormalities identified in the literature [15, 13] are within lobes. Especially, as indicated in Figure 8, the most strongest connections include connections within frontal gyrus, e.g. Superior frontal gyrus dorsolateral-Middle frontal gyrus (4, 8), Middle frontal gyrus orbital part-Inferior frontal gyrus opercular part (9, 11) and connections between Hippocampus (37∼40), Amygdala (41∼42) and Calcarine (43∼44). An interesting thing this figure presented is that the same parts in left and right brain are often closely functionally connected. For example, the connections between Inferior frontal gyrus orbital part in left brain and Inferior frontal gyrus orbital part in right brain (15, 16), and the connection between Rolandic operculum in left brain and Rolandic operculum in right brain (17, 18). This is probably due to the extensive cooperation between

the left and right brain in many functions. Since the experimental study has demonstrated that the integrated network possesses more discriminative power in comparison with a network corresponding to a single sparsity level, the recovered pre-image in the original space may reveal more disease-related connectivity patterns. In this sense, the visualization of the integrated brain networks provides medical specialist a new perspective to conduct analysis of the brain networks and this may promote understanding the underlying pathophysiology of brain diseases.

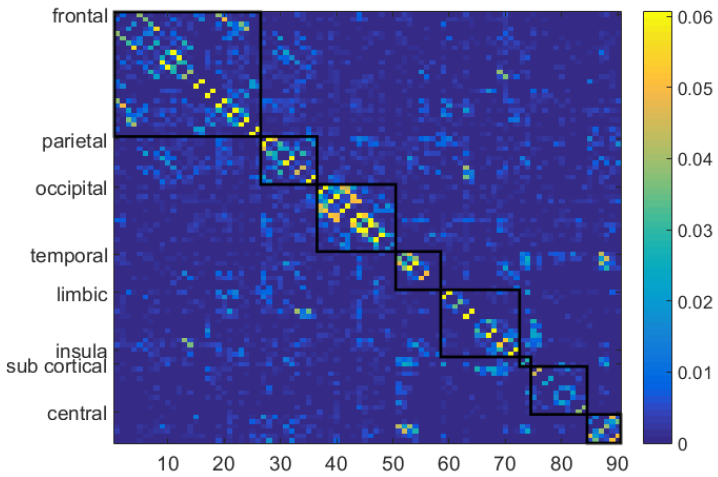


Figure 5.8: An example of the recovered pre-image of the integrated network.

Convergence Evaluation

As discussed in Section 5.3.5, the alternate optimization of the sparse representation coefficient $\boldsymbol{\alpha}$ and the combination coefficient $\boldsymbol{\beta}$ is guaranteed to converge. Here, we would like to verify the evolution of the objective function defined in Eq.(5.3). The evolution of the objective values averaged over all samples in the test set is plotted in Figure 5.9. As seen, the objective is monotonically decreased by optimizing $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ alternately. Moreover, the objective value decreases significantly in the first few

iterations and quickly becomes convergent. This result experimentally demonstrates that the proposed optimization method can be effectively and efficiently solved.

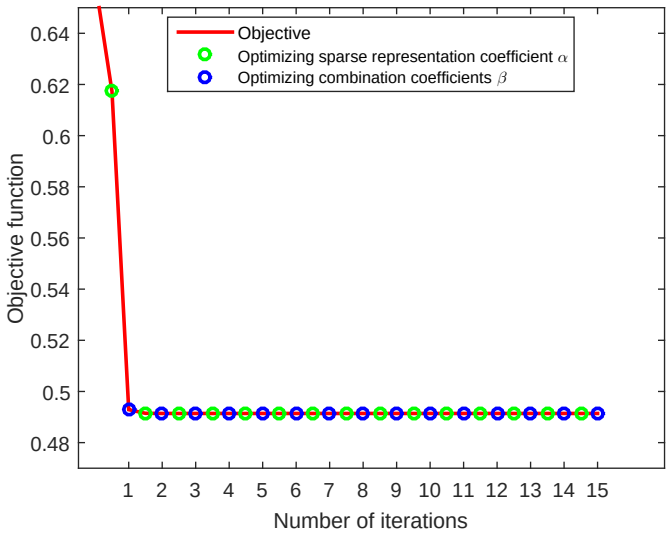


Figure 5.9: Evolution of the objective function with respect to the number of iterations.

5.5 Conclusion

Recently, sparse inverse covariance estimation (SICE) has been widely employed to model brain connectivity networks and naturally leads to a set of connectivity networks with different sparsity for each subject. To explore the complementary information in the set of networks, this thesis proposes a learning framework that integrates brain networks and respects the underlying manifold structure of the SPD-based network representations. The proposed framework conducts a subject-adaptive integration via a kernel sparse learning scheme, and the obtained integrated network representation can be projected back in the original space for medical-related exploration. The effectiveness of the proposed method is verified on both ADHD and ADNI data sets. The experimental results demonstrate that the proposed integration method considerably outperforms

a single network based methods and other commonly used integration methods.

Chapter 6

Beyond Covariance Representation: Kernel Matrices

As introduced in previous chapters, covariance matrix has recently received increasing attention in computer vision by leveraging Riemannian geometry of symmetric positive-definite (SPD) matrices. Originally proposed as a region descriptor, it has now been used as a generic representation in various recognition tasks. However, covariance matrix has shortcomings such as being prone to singularity, being incapable of modeling nonlinear relationship, and lack of flexibility. This chapter argues that more appropriate SPD-matrix-based representations shall be explored to achieve better recognition. It proposes an open framework to use the kernel matrix over feature dimensions as a generic representation and discusses its properties and advantages. The proposed framework significantly elevates covariance representation to the unlimited opportunities provided by this new representation. Experimental study shows that this representation consistently outperforms its covariance counterpart on various visual recognition tasks. In particular, it achieves significant improvement on skeleton-based human action recognition, demonstrating the state-of-the-art performance over both the covariance and the existing non-covariance representations.

6.1 Motivation

The applications of covariance matrix to visual recognition can be categorized into two classes:

i) *As a region descriptor.* This dominates the initial applications. The superiority of region covariance descriptor is firstly shown on object detection and texture classification [93] and then on object tracking [70]. It is further applied to pedestrian detection [94], face recognition [67], and shape retrieval [90]. Two characteristics can be observed on these applications: 1) fast computation of region covariance descriptors is highly essential, especially for the tasks like object detection and tracking; 2) the dimensions of covariance matrix are usually low (e.g., 5×5 or 8×8).

ii) *As a general representation.* This has recently been seen in an increasing number of tasks. For human action recognition, a representation Cov3DJ is proposed to model a sequence of skeletal joint motions over time [38, 32]. In image set classification [108], a feature vector is extracted from every image in a given set and its covariance matrix is then computed to represent this image set. A similar case is observed in gesture recognition, where the covariance matrix of frame-based features is used to represent a video sequence. Two new characteristics have been observed.

1) The wider range of applications poses a challenge on covariance matrix with respect to its effectiveness as a generic representation. The requirement on extensively modeling sophisticated feature relationships becomes evident. As a result, new SPD-matrix-based representations with more expressive power are highly desired.

2) Features are not pixel-based anymore and often have higher dimensions. In an action recognition data set in the experiment, the dimensions are as high as 120, while the number of feature vectors per action instance only ranges from 40 to 500, far from being enough to estimate a reliable covariance matrix. A worse case is in image set classification. The dimensions could be as high as 400 (when reshaping a 20×20 object image), while there are only 41 images in a set [108]. This not only results in unreliable

estimate but also incurs the singularity of covariance matrix.

To address these issues, we propose to use kernel matrix as a generic feature representation. Each of its entries evaluates a kernel function between a pair of *feature dimensions* (rather than between a pair of *samples*, as we usually do in kernel methods). As will be shown, for a large set of kernel functions, the kernel matrix is guaranteed to be nonsingular, even if samples are scarce, which ensures Riemannian metrics to be readily applicable. More importantly, different kernel functions could model different nonlinear feature relationships, making the kernel matrix based representation very flexible in different applications.

The following of this chapter firstly introduces the proposed kernel matrix representation and elaborates its properties and advantages. Following that, the computational issues are discussed. At last, the performance of the proposed representation is verified on skeleton-based human action recognition, image set classification, and object recognition.

6.2 Proposed Method

To keep brevity, we use "covariance representation" and "kernel representation" as the short names of covariance matrix based and kernel matrix based representations.

6.2.1 Issues of covariance representation

Under the above new characteristics, covariance matrix as generic feature presentation has the following drawbacks.

Let $\mathbf{x}$ ($\mathbf{x} \in \mathbb{R}^d$) be a d-dimensional feature vector, $\mathbf{D}_{d\times n} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$ denote a data matrix, and $\mathbf{C}$ be the corresponding sample-based covariance matrix. Firstly, covariance matrix only describes linear correlation of features. Let $\mathbf{f}_i^\top$ $(i = 1, \cdots, d)$ be the i^{th} row of $\mathbf{D}$, standing for the realization of the i^{th} feature. After centering, it can

be written as $\bar{\mathbf{f}}_i = \mathbf{f}_i - \mu_i \mathbf{1}$, where μ_i is the sample mean while $\mathbf{1}$ is a column vector of "1"s. It is trivial to show that the $(i,j)^{th}$ entry of covariance matrix $\mathbf{C}$ is

$$c_{ij} = \left\langle \frac{\bar{\mathbf{f}}_i}{\sqrt{n-1}}, \frac{\bar{\mathbf{f}}_j}{\sqrt{n-1}} \right\rangle, \tag{6.1}$$

where $\langle \cdot, \cdot \rangle$ denotes an inner product. In other words, covariance matrix essentially implements a *linear* kernel function over scaled $\bar{\mathbf{f}}_i$ and $\bar{\mathbf{f}}_j$. When fast computation of a region descriptor is necessary, such linearity brings conceptual simplicity and computational efficiency. Nevertheless, from the perspective of generic representation, modeling only linear relationship significantly constrains its expressive power and in turn affects recognition performance. For example, for action recognition, it is certainly not sufficient to only consider the linear correlation of skeleton joints to model and differentiate various action patterns [91].

Secondly, the rank of covariance matrix obeys $\mathrm{rank}(\mathbf{C}) \leq \min(d, n-1)$. When $\mathbf{C}$ is used as a region descriptor, the number of feature vectors extracted from an image region, n, is usually much larger than the dimensions, d. This ensures $\mathbf{C}$ to be nonsingular and allows it to be reliably estimated. However, this situation has changed in recent applications, and singularity could occur. In that case, in order to utilize Riemannian metrics, a small scaled identity matrix has to be appended [108].

6.2.2 Kernel matrix as feature representation

We propose to use a kernel matrix, $\mathbf{M}$, as a generic feature representation. The $(i,j)^{th}$ entry of $\mathbf{M}$ is defined as

$$k_{ij} = \langle \phi(\mathbf{f}_i), \phi(\mathbf{f}_j) \rangle = \kappa(\mathbf{f}_i, \mathbf{f}_j), \tag{6.2}$$

where $\phi(\cdot)$ is an implicit nonlinear mapping and $\kappa(\cdot,\cdot)$ is the induced kernel function. Covariance matrix corresponds to a special case in which $\phi(\mathbf{f}_i) = (\mathbf{f}_i - \mu_i\mathbf{1})/\sqrt{n-1}$. **Note that** the mapping $\phi(\cdot)$ is applied to each *feature* $\mathbf{f}_i$, rather than to each *sample* $\mathbf{x}_i$ as usually seen in kernel-based learning methods. The most significant advantage of using M lies at that with it, we can have much more flexibility to efficiently model the nonlinear relationship among features.

i) For example, we can evaluate the similarity of feature distributions, by applying the Bhattacharyya kernel [46]

$$\kappa(\mathbf{f}_i, \mathbf{f}_j) = \int \sqrt{p_i(z)}\sqrt{p_j(z)}dz, \tag{6.3}$$

where $p_i(z)$ and $p_j(z)$ denote two univariate distributions estimated from $\mathbf{f}_i$ and $\mathbf{f}_j$. When the two distributions are assumed to be Gaussian, denoted by $\mathcal{N}(\mu_i, \sigma_i)$ and $\mathcal{N}(\mu_j, \sigma_j)$, this kernel has a closed form as

$$\kappa(\mathbf{f}_i, \mathbf{f}_j) = \sqrt{\frac{2\sigma_i\sigma_j}{\sigma_i^2 + \sigma_j^2}} \exp\left[-\frac{1}{4}\frac{(\mu_i - \mu_j)^2}{\sigma_i^2 + \sigma_j^2}\right]. \tag{6.4}$$

ii) We can also model the interaction among samples with respect to a feature. Recall that $\mathbf{f}_j$ is a column vector $(x_{1j}, x_{2j}, \cdots, x_{nj})^\top$, where x_{ij} is the j^{th} feature of the i^{th} sample, $\mathbf{x}_i$. All the p-order "interaction" of samples can be exhaustively generated by mapping $\mathbf{f}_j$ (or $\bar{\mathbf{f}}_j$) as follows

$$\phi(\mathbf{f}_j) = \left\{\sqrt{\left(\frac{p!}{r_1!r_2!\cdots r_n!}\right)} x_{1j}^{r_1} x_{2j}^{r_2} \cdots x_{nj}^{r_n}\right\},\ \forall r_1, \cdots, r_n; \tag{6.5}$$

where $\sum_{i=1}^{n} r_i = p$ and $r_i \geq 0$. Introducing these features could be beneficial. For example, for skeleton-based action recognition, they consider all the p-order interactions of a given feature over the n frames of an action instance. This mapping induces a

simple homogeneous polynomial kernel $\kappa(\mathbf{f}_i, \mathbf{f}_j) = \langle \phi(\mathbf{f}_i), \phi(\mathbf{f}_j) \rangle = (\langle \mathbf{f}_i, \mathbf{f}_j \rangle)^p$, where p is the degree of this kernel [7]. Therefore, with the proposed kernel representation, the relationship between a pair of high-order sample interactions can be conveniently evaluated.

iii) In practice, applying a kernel representation could be even easier, when we do not know beforehand (or are not particularly interested in) what kind of nonlinear relationship shall be modeled. In this case, any general-purpose kernel, such as the Gaussian RBF kernel $\kappa(\mathbf{f}_i, \mathbf{f}_j) = \exp(-\beta \|\mathbf{f}_i - \mathbf{f}_j\|^2)$, can be employed. Also, once it becomes necessary, users are free to design new, specific kernels to serve their goals. Such flexibility is clearly an advantage brought by using a kernel matrix as feature representation.

In relation to the singularity issue, kernel matrix is also a better choice than covariance matrix. When $d \geq n$ is true, covariance matrix is bound to be singular. In contrast, the situation is more favorable for kernel matrix. A direct application of Micchelli's Theorem (1986) [34] gives the following result for our case.

Theorem 1. *Let $\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_d$ be a set of different n-dimensional vectors. The matrix $\mathbf{M}_{d \times d}$ computed with a RBF kernel $\kappa(\mathbf{f}_i, \mathbf{f}_j) = \exp(-\beta \|\mathbf{f}_i - \mathbf{f}_j\|^2)$ is guaranteed to be nonsingular, no matter what values d and n are.*

According to Micchelli's Theorem, the inverse multiquadric kernel $\kappa(\mathbf{f}_i, \mathbf{f}_j) = (\|\mathbf{f}_i - \mathbf{f}_j\|^2 + \beta^2)^{-1/2}$ also satisfies the above theorem. Actually, as pointed out in [18], in addition to these two kernels, there are a large set of various kernels holding this nice property, including radial kernels, translation invariant kernels, multiscale kernels, power series kernels, etc. The presence of these kernels provides us great freedom to choose the most appropriate one for a kernel representation. Lastly, in case we cannot be sure about the nonsingularity for a kernel matrix, we can always analyze it with the definition of positive definiteness and/or append a regularizer to this matrix as a preemptive measure.

6.2.3 Computational issues

Without loss of generality, we use the commonly used RBF kernel as an example. Given n d-dimensional vectors, $\mathbf{x}_1, \cdots, \mathbf{x}_n$, computing all the entries $\|\mathbf{f}_i - \mathbf{f}_j\|^2$ $(i, j = 1, \cdots, d)$ has the complexity of $\mathcal{O}(nd^2)$, same as computing a covariance matrix. Certainly, RBF kernel has an $\exp(\cdot)$ operation and needs a bit more time. In addition, although the case of region descriptor is not our focus, we show that the proposed kernel representation could still be quickly computed via integral images. Noting that $\|\mathbf{f}_i - \mathbf{f}_j\|^2 = \mathbf{f}_i^\top \mathbf{f}_i - 2\mathbf{f}_i^\top \mathbf{f}_j + \mathbf{f}_j^\top \mathbf{f}_j$, we can precompute d^2 integral images for the inner product of any two feature dimensions. It then becomes trivial to compute $\|\mathbf{f}_i - \mathbf{f}_j\|^2$ for any rectangular regions by following [93]. This result is also valid for the polynomial kernel which computes $\mathbf{f}_i^\top \mathbf{f}_j$.

Generally, the availability of more samples makes kernel evaluation more reliable. Take the Bhattacharyya kernel (Eq (6.3)) as an example. More samples make the estimates μ and σ converge towards their true values. This in turn helps the kernel evaluation to converge towards its true value. Certainly, in practice we are constrained by the number of available training samples. Also, recall that the proposed kernel matrix has a fixed size ($d \times d$), independent of the number of samples (n) in a set. Due to this, the kernel-based representations obtained from two different-sized sets can be directly compared. At the same time, considering that n affects the lengths of $\mathbf{f}_i$ and $\mathbf{f}_j$, we scale them accordingly to reduce the impact of n. For example, we divide $\|\mathbf{f}_i - \mathbf{f}_j\|$ by the average pairwise Euclidean distance over a training set, when the RBF kernel is used.

6.3 Experimental Result

The proposed kernel representation (Ker-RP in short) is compared with covariance representation (Cov-RP) on three types of recognition tasks. The first two are human action recognition and image set classification, which use covariance as generic feature

representation. The third one includes the tasks of face, texture, and object recognition traditionally used when covariance acts as a region descriptor.

All the three kernels in Section 6.2.2 are involved. In specific, the representations generated by the Gaussian radial basis function kernel (RBF in short) and the polynomial kernel (POL) are compared with covariance representation to verify their advantages. The representation generated by the Bhattacharyya kernel (BHA) will be combined with covariance representation to demonstrate the benefit of combining two complementary representations.

A nonlinear SVM classifier is used in all experiments. The log-Euclidean kernel, a commonly used kernel function on SPD matrices[1], is employed for the SVM. To ensure fair comparison, all algorithmic parameters, including the regularization parameter in SVM, β in the log-Euclidean kernel, and the parameters in the RBF and POL kernels are tuned by multi-fold cross-validation on the training set only.

6.3.1 Result on human action recognition

Four benchmark data sets are used, including MSR-Action3D, MSR-DailyActivity3D, MSRC-12, and HDM05. For all of them, we only use the *skeleton* data while the other data (e.g., depth maps or RGB videos) are not included. The data set information is in Table 6.1.

Table 6.1: Feature dimensions of four action recognition data sets.

Data set	#Dim. (d)	#frames per instance (n)	Features
MSR-Action3D	120	$40 \sim 60$	Velocity
MSR-DailyActivity3D	120	$125 \sim 500$	Velocity
MSRC-12	60	$50 \sim 300$	Coordinates
HDM05	93	$30 \sim 700$	Coordinates

[1]The log-Euclidean kernel function is defined as $k(\mathbf{X}, \mathbf{Y}) = \exp\left(-\beta \| \log(\mathbf{X}) - \log(\mathbf{Y}) \|_F^2\right)$, where $\mathbf{X}$ and $\mathbf{Y}$ are two SPD matrices and $\log(\cdot)$ denotes the matrix logarithm.

For MSR-Action3D and MSR-DailyActivity3D, we use velocity as the frame features [120] by calculating the coordinate differences of 3D skeleton joints between a frame and its two (before and after) neighboring frames. Each frame feature vector has 120 ($2 \times 3 \times 20$ joints) dimensions. For MSRC-12 and HDM05, the 3D coordinates of each joint are used as the frame features. As seen in Table 6.1, for each data set, the number of frames per instance, n, could be smaller than $(d+1)$, which causes singularity when computing a covariance matrix. In this case, we append a small regularizer $\lambda\mathbf{I}$ (e.g., $\lambda = 10^{-7}$) to the matrix as in the literature [108].

To facilitate comparison with the state-of-the-art methods, the training and test sets of these data sets are partitioned by following the literature. For MSR-Action3D, MSR-DailyActivity3D and MSRC-12, the cross-subject test setting [53] is used, in which the odd-indexed subjects are used for training and the even-indexed ones are for test. For HDM05, we used the instances of two subjects for training and those from the remaining three subjects for test [30].

Result on MSR-Action3D data set

MSR-Action3D contains 20 actions performed by ten subjects. Each action is done two or three times by each subject. The number of frames in each action instance is $40 \sim 60$, which is smaller than the feature dimensions, 120. The classification accuracy is compared in Table 6.2.

The upper portion of this table quotes the state-of-the-art results in the last two years, while the lower portion lists the results of the methods implemented by this work. Cov-RP is the method using covariance representation. Cov-$J_{\mathcal{H}}$-SVM is the method in [32] which uses an infinite-dimensional covariance matrix in a kernel-induced feature space as representation. Ker-RP-POL and Ker-RP-RBF are the proposed methods, in which polynomial and RBF kernels are used to compute the kernel representation. As seen, Cov-RP performs poorly when compared with the quoted state-of-the-art methods. Its

performance is probably affected by the insufficient number of frames for covariance matrix estimation. Cov-$J_{\mathcal{H}}$-SVM well improves over Cov-RP. However, it is still inferior to the quoted state-of-the-art ones. In contrast, the proposed methods significantly outperform Cov-RP, obtaining an improvement over 20 percentage points. Also, both methods outperform Cov-$J_{\mathcal{H}}$-SVM by a large margin, and even win these state-of-the-art methods which use complex feature representation (e.g., sparse coding [117]) or multiple forms of data such as depth maps and skeleton [124]. This result is significant and encouraging, indicating the efficacy of the kernel representation. With it, classification accuracy on this action data set is boosted from 94.3% [124] further to 96.9%.

Table 6.2: Comparison on MSR-Action3D data set.

Methods in comparison	Accuracy
Pose Set [104]	90.0
Hierarchy of Cov3DJs [38]	90.5
Moving Pose [120]	91.7
Lie Group [102]	92.5
SNV [117]	93.1
Spatiotemp. Features Fusing [124]	94.3
Cov-RP [93]	74.0
Cov-$J_{\mathcal{H}}$-SVM [32]	80.4
Ker-RP-POL (proposed)	96.2
Ker-RP-RBF (proposed)	**96.9**

Result on MSR-DailyActivity3D data set

MSR-DailyActivity3D is a challenging data set, because the extracted skeletons are noisy and most activities involve human-object interactions such as *drink*, *eat*, *read book*, etc. Table 6.3 shows the comparison result. As previous, some state-of-the-art results are quoted in the upper portion, followed by the results of the methods implemented by this work. On this data set, Cov-RP becomes better and close to the state-of-the-art ones. However, Cov-$J_{\mathcal{H}}$-SVM does not improve over Cov-RP but shows a degraded performance. The two proposed methods once again demonstrate significant improvement over all the other methods. In specific, Ker-RP-POL yields the highest

accuracy 96.9%. It wins the best state-of-the-art method (SNV [117]) by more than ten percentage points. Ker-RP-RBF also achieves an excellent result of 96.3%, close to Ker-RP-POL and outperforms the other ones by a large margin. Note that in these state-of-the-art methods, depth map is used to extract features in [64, 117], and local occupancy patterns are used in [105] to process human-object interaction cases. We compute the kernel representation using the skeleton data only. In addition, for this data set, the number of frames in each action instance is generally larger than the feature dimensions, making covariance estimation free of the singularity issue. Nevertheless, the result shows that the proposed kernel representation still has an advantage over covariance representation in this case. We attribute this advantage to its capability in modeling nonlinear feature relationship.

Table 6.3: Comparison on MSR-DailyActivity3D data set.

Methods in comparison	Accuracy
Moving Pose [120]	73.8
Local HON4D [64]	80.0
Actionlet Ensemble [105]	86.0
SNV [117]	86.3
Cov-RP [93]	85.0
Cov-$J_{\mathcal{H}}$-SVM [32]	75.0
Ker-RP-POL (proposed)	**96.9**
Ker-RP-RBF (proposed)	96.3

Result on HDM05 data set

HDM05 consists of around 1500 instances from over 100 motion classes. Most classes have 10 to 50 realizations of five actors named "bd", "bk", "dg", "mm" and "tr". We use two subjects "bd" and "mm" for training and the remaining three for test [30]. To compare with the literature, we conduct two experiments. Firstly, we use 14 classes[2] of this data set, and the result is in the left column of Table 6.4. Cov-RP shows quite

[2]They are 'clap above head', 'deposit floor', 'elbow to knee', 'grab high', 'hop both legs', 'jog', 'kick forward', 'lie down floor', 'rotate both arms backward', 'sit down chair', 'sneak', 'squat', 'stand up lie' and 'throw basketball'.

competitive performance and outperforms the quoted methods. However, Cov-$J_{\mathcal{H}}$-SVM shows a degraded performance again. Ker-RP-RBF and Ker-RP-POL are still significantly better than Cov-RP, Cov-$J_{\mathcal{H}}$-SVM, and the other methods. The highest classification accuracy 96.8% is obtained by Ker-RP-RBF. Note that all the quoted methods use covariance-based representation, but sparse coding or dimensionality reduction is additionally applied to improve the performance. To further verify their effectiveness, we conduct comparison on all the classes. As shown in the right column of Table 6.4, although the significant increase on the number of action classes reduces the overall classification accuracy, the proposed methods still outperform the other ones in comparison.

Table 6.4: Comparison on HDM05 data set (Two experiments).

Methods in comparison	14 classes Accuracy	All classes Accuracy
CDL [108]	79.8	Not reported
RSR [31]	76.1	Not reported
RSR-ML [30]	81.9	40.0
Cov-RP [93]	91.5	58.9
Cov-$J_{\mathcal{H}}$-SVM [32]	82.5	-
Ker-RP-POL (proposed)	93.6	64.3
Ker-RP-RBF (proposed)	**96.8**	**66.2**

$\star$The result of Cov-$J_{\mathcal{H}}$-SVM [32] is not obtained in 35 hours.

Result on MSRC-12 data set

MSRC-12 is a large data set, containing the performance of 12 gestures by 30 subjects. As shown in Table 6.5, Ker-RP-RBF again obtains the best classification result, outperforming Cov-RP and the other methods including Cov-$J_{\mathcal{H}}$-SVM. Ker-RP-POL's performance is a bit lower than that of Ker-RP-RBF. This may indicate that the RBF kernel fits better the action data in this data set. Nevertheless, Ker-RP-POL is still higher than Cov-RP and Cov-$J_{\mathcal{H}}$-SVM. Note that the method in [38] uses a hierarchy of multiple covariance matrices to capture the temporal order of motion. For each instance, the

covariance matrix at the top level is computed over the whole sequence, while those at the lower levels are computed over a series of sub-sequences in order. We believe that our methods can be further improved if working in that manner.

Table 6.5: Comparison on MSRC-12 data set.

Methods in comparison	Accuracy
Hierarchy of Cov3DJs [38]	91.7
Cov-RP [93]	89.2
Cov-$J_{\mathcal{H}}$-SVM [32]	89.2
Ker-RP-POL (proposed)	90.5
Ker-RP-RBF (proposed)	**92.3**

6.3.2 Result on image set classification

An image set is a collection of images belonging to the same class but with variation, for example, images of the same object under different views. It is the image set, rather than an individual image, that will be classified. Covariance matrix has been used to model an image set [108]. Now we compare it with the proposed kernel representation. Three data sets used by [108] are tested, including ETH80, CMU MoBo, and YouTube Celebrities. ETH80 has eight categories, with ten objects per category. For each object, there are 41 images showing different views. CMU MoBo has 96 video sequences of 24 subjects. YouTube Celebrities consists of 1910 video clips from 47 subjects. These data sets are preprocessed by [108] as follows. For YouTube and CMU MoBo, face images of each subject are collected by face detectors and resized to 20×20 pixels. Pixel intensities are used as features, leading to a 400-dimensional vector per image. The object images in ETH80 are also resized to 20×20 and pixel intensities are used as features. These data sets are downloaded from [108].

Training and test sets are created as follows. For CMU MoBo, all face images detected from the same video sequence form an image set. One image set is randomly selected from each subject for training, and the remaining image sets are for test. For

YouTube, three image sets are randomly chosen from each subject for training, and another six sets are randomly chosen for test. In ETH80, the ten objects in a category are randomly halved into training and test sets. For each object, the 41 images of different views form an image set. The kernel- and covariance-representations are used to represent each image set. In total, 100 training and test pairs are created for each data set.

Following [108], we use Partial Least Squares (PLS) for classification and the code is downloaded from that work[3]. Table 6.6 reports the average results. Ker-RP-RBF achieves the best classification performance on ETH80, outperforming Cov-RP by 3.2 percentage points and Cov-$J_{\mathcal{H}}$-SVM by 3.5 percentage points. On CMU MoBo, it still significantly improves over Cov-RP and is comparable to Cov-$J_{\mathcal{H}}$-SVM. On YouTube, Ker-RP-RBF performs slightly worse than Cov-RP by 0.8 percentage point but clearly outperforms Cov-$J_{\mathcal{H}}$-SVM. Also, Ker-RP-POL performs better than Cov-RP on ETH80 by 2.1 percentage points, while worse on the other two data sets. This result reflects the importance of choosing an appropriate kernel function for the kernel representation. Also, compared with all the other methods, the RBF-kernel representation shows overall best classification performance over the three data sets.

Table 6.6: Comparison on three image set classification data sets.

Methods	ETH80	CMU MoBo	YouTube
Cov-RP (CDL [108])	92.7	83.9	**61.2**
Cov-$J_{\mathcal{H}}$-SVM [32]	92.4	**88.9**	54.4
Ker-RP-POL (proposed)	94.8	75.3	57.3
Ker-RP-RBF (proposed)	**95.9**	88.4	60.4

[3]The work [108] also investigates Linear Discriminant Analysis. However, PLS always outperforms LDA as shown in that work.

6.3.3 Result on object classification

We further investigate the effectiveness of the proposed kernel representation on the tasks traditionally applied with covariance matrix as a region descriptor. For them, the feature dimensions are usually lower and a larger number of feature vectors are available for covariance estimation. We use three data sets, including Brodatz for texture classification, FERET for face recognition, and ETH80 for object categorization. Brodatz contains 112 textured images. Following the literature [31], each image is partitioned into 64 non-overlapping sub-images. All sub-images from the same image form one texture class, and these sub-images are classified. For FERET, we use the "b" subset of 198 subjects. Each has 10 images with various poses and illumination conditions. ETH80 was used for image set classification in Section 6.3.2, but here each image is considered as a training or test sample and classified.

For all three data sets, every image/sub-image is scaled to a uniform size of 64×64 and a 43-dimensional feature vector is extracted at each pixel, including its intensity, x and y coordinates, and a set of Gabor features (8 orientations and 5 scales). For each experiment, we randomly halve the data set into training and test subsets. This is repeated 20 times to obtain average classification performance. As seen in Table 6.7, Ker-RP-RBF again outperforms Cov-RP, by 3.7 and 4.4 percentage points on Brodatz and FERET. This indicates the effectiveness of the proposed kernel representation even when it acts as a region descriptor. Note that Cov-$J_{\mathcal{H}}$-SVM is not included because it becomes time-consuming when the number of feature vectors, n, is large. As reported in Table 6.8, we cannot obtain its result even after 35 hours.

In addition, since the number of feature vectors (4096 per image) is now adequately larger than feature dimensions (43), we can compare the sensitivity of the two representations against the number of feature vectors. Brodatz data set and the RBF kernel are used. In Figure 6.1, the x-axis is the ratio of the number of feature vectors used to compute the kernel- or covariance-representations. The y-axis is the classification accu-

racy corresponding to the resulting representation. As shown, Ker-RP-RBF consistently outperforms Cov-RP, although both of them degrade with the decreasing ratio. In particular, the margin between them becomes even larger when the ratio is lower than $1/75$ (about 55 feature vectors), indicating the more significant advantage of Ker-RP-RBF when feature vectors are scarce. This suggests that modeling nonlinear feature relationship enhances the expressive power of SPD-matrix-based representation and benefits classification, especially in the case of a small number of feature vectors available.

Table 6.7: Comparison on object classification data sets.

Methods	Brodatz (texture)	FERET (face)	ETH80 (object)
Cov-RP [93]	81.2	81.0	94.0
Ker-RP-POL (proposed)	77.9	82.4	93.8
Ker-RP-RBF (proposed)	**84.9**	**85.4**	**94.8**

⋆The result of Cov-$J_{\mathcal{H}}$-SVM [32] is not obtained in 35 hours.

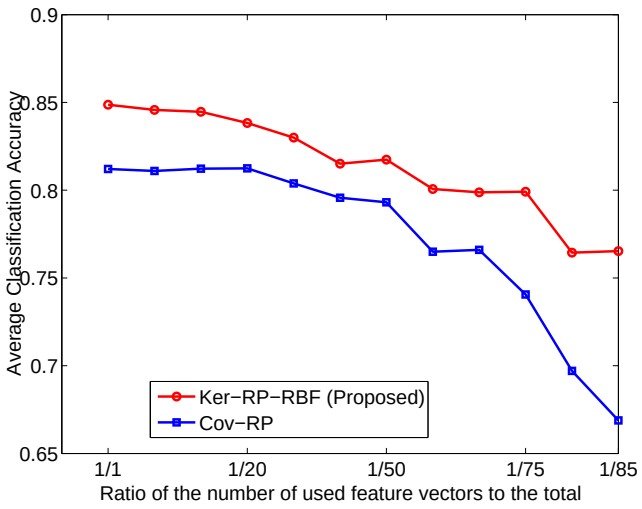


Figure 6.1: Comparison of the sensitivity (in terms of classification accuracy) of the kernel- and covariance-representation with respect to the number of feature vectors used to compute them.

6.3.4 Computation time

Table 6.8 compares the computation time of Cov-RP, Cov-$J_{\mathcal{H}}$-SVM, and Ker-RP-RBF on all the data sets. A desktop computer with 3.6 GHz CPU and 32GB memory is used. Recall that Cov-$J_{\mathcal{H}}$-SVM does not have an explicit representation. To make fair comparison, we compare the time for computing the whole kernel matrix $\mathbf{G}$ for a pair of training and test sets, which is needed by SVM classification. The value in brackets shows the time for computing the covariance or kernel representation. As seen, our kernel representation only slightly increases computation time (e.g., from 0.1 to 0.2 second), which is insignificant compared to the total time for computing $\mathbf{G}$. However, Cov-$J_{\mathcal{H}}$-SVM incurs much higher computational load, except on ETH80 which has a small number of samples, 41. In addition, on four data sets, we cannot obtain the matrix $\mathbf{G}$ by Cov-$J_{\mathcal{H}}$-SVM for a single pair of training and test sets even after 35 hours (and therefore the respective classification performance is not provided). This shows the computational efficiency of our kernel representation.

Table 6.8: Comparison of the time for computing the whole kernel matrix $\mathbf{G}$ used for SVM classifier training and test. The value in brackets is the time used to compute the covariance or the proposed kernel representation. (Unit: second)

Data set	Cov-RP	Cov-$J_{\mathcal{H}}$-SVM [32]	Ker-RP-RBF (Proposed)
MSR-A-3D	61.4 (0.1)	349	65.9 (0.2)
MSR-DA-3D	20.6 (0.2)	6.4×10^3	22.5 (0.3)
MSRC-12	1.3×10^3 (0.6)	3.3×10^4	1.3×10^3 (1.2)
HDM05(14)	11.4 (0.1)	1.8×10^3	15.6 (0.2)
HDM05	884.6 (0.9)	$>$ 35 hours	1037(1.6)
ETH80	28.0 (0.1)	6.5	27.9 (0.2)
CMU MoBo	21.6 (0.2)	74.0	20.4 (0.3)
YouTube	549 .0 (0.7)	898	546.9 (1.3)
Brodatz	1.4×10^3 (6.5)	$>$ 35 hours	1.4×10^3 (22.1)
FERET	109.6 (1.8)	$>$ 35 hours	110.7 (5.5)
ETH80	299.8 (2.9)	$>$ 35 hours	302.3 (9.1)

6.4 Conclusion

To address the new issues encountered by covariance representation, we propose to use kernel matrix as a generic feature representation. This new representation models more sophisticated feature relationship, is more robust against sample scarcity, and maintains computational efficiency. The significant improvement achieved by this representation is verified on a variety of tasks.

Chapter 7

Discussion

7.1 Contribution Summary

The main contribution of this thesis is developing advanced learning methods for covariance representations from several perspectives. Firstly, this thesis provides a novel perspective to learn a data-driven kernel function for covariance descriptors. It integrates class label information into the Stein kernel to adjust input covariance matrices to enhance its discriminative capability. As far as we know, we are the first to learn kernel functions through eigenvalue adjustment. This brings insight into the connection between eigenvalues and similarity measures of covariance descriptors. This novel perspective could probably inspire other researchers to develop advanced similarity measures for SPD matrices. From this work, we realize that two aspects play an important role in discovery of better kernels for SPD matrices: 1) A good distance measure should effectively take the underlying structure of data into account. As a specially designed distance measure, the (square-rooted) S-Divergence we used in this work well respects the Riemannian manifold where SPD matrices reside. 2) The class information should be effectively integrated into kernel functions to improve its quality in further. In this work, we achieve this by utilizing the class information to adjust the eigenvalues to

make Stein kernel better align with specific classification tasks.

Secondly, this thesis proposes a kernel-PCA based method to extract compact features from SICE matrices. The proposed method jointly considers the SPD geometry of SICE matrices and data distribution. When applied to functional brain network analysis, the features extracted by this method obtain higher classification accuracy in comparison with state-of-the-art methods. Furthermore, this thesis proposes to integrate multiple SICE matrices generated for the same subject to improve classification accuracy. Multiple SICE matrices at different sparsity levels present complementary information which has not been sufficiently explored in the literature previously. This thesis demonstrates that such information is useful and could be integrated to boost the classification accuracy. This work verifies that fully utilizing the information contained in the data is critical to achieve the state-of-the-art performance.

Besides, this thesis moves beyond covariance representation and significantly extends the SPD representations in visual recognition tasks from fixed covariance descriptor to general kernel matrices. This not only resolves the high dimensionality and small sample problems encountered by covariance representation, but can also take advantage of the capability of kernel matrix in modeling nonlinear relationship among features. This novel representation also expands our understanding of kernels. Kernels are previously often used to perform tasks such as statistical classification, regression analysis, and cluster analysis on data in an implicit space, however, this thesis shows that they can also be used as feature representations. This work well demonstrates that understanding the essence of the existing methods is a critical step to improve them.

7.2 Future Work

Research in this thesis can be extended in various ways in future. In terms of similarity measure for SPD matrices, we improve Stein kernel using supervised learning through

eigenvalue adjustment. On the one hand, the idea of adjusting eigenvalues could be used to improve other kernels. On the other hand, the proposed approach shall be extendable to the unsupervised case, which can be applied to better cluster the data represented by covariance descriptors. In that situation, how to incorporate cluster information to improve SPD kernels will also be an interesting topic to explore.

For the application of SPD matrices in medical image analysis, the future work could explore two directions. The first direction is to apply the proposed methods to analyze other brain diseases and the second direction is to explore a nonlinear integration of a set of networks.

Regarding the kernel matrix representation, the future work could gain more insight on the learned representations, for example, by visualizing them, and analyze the sensitivity of this representation to the number of samples in depth. Also, with the verified performance, several research issues on this new representation are worth exploring, including automatically choosing and designing appropriate kernels, its unsupervised learning methods, and the applications to more visual tasks. Another issue worth investigation is the combination between kernel representations and Convolutional Neural Networks (CNNs). The powerfulness of CNNs has recently been widely recognized. It incorporates feature extraction, feature representation, label prediction into a unified framework. Especially, CNNs have been considered as the state-of-the-art feature extraction approach. Our proposed kernel representation is evaluated using hand-crafted features. It can be expected that applying kernel representation to CNN features could generate more promising performance. Moreover, it is even possible to embed kernel representations into the training procedure of CNNs. For example, in image set classification tasks, multiple images should be jointly considered to do classification. Nevertheless, CNNs are mainly designed for single-image based classification. It could be promising to incorporate kernel representation as an add-in layer to fuse the features from multiple images in the same set. In doing so, the advantages of kernel representa-

tions and CNNs can be inherently combined to improve the classification performance.

Acknowledgments

First of all, I would like to express my most sincere and deepest gratitude to my supervisors, A/Prof. Lei Wang and Dr. Luping Zhou, for their constant guidance, support, encouragement and advice. What I gained from them has never been only about knowledge and skills. I also learned much about principles of conduct either as a researcher or person and attitude in front of difficulties, which will help me through my entire career and life. It is my pleasure to be their student and it was wonderful working with them during the last four years. Besides, I would like to thank the rest of my supervisory panel: A/Prof. Wanqing Li, for his insightful advice and kind support.

Secondly, I would like to sincerely thank my friends I have met here, Dr. Lingqiao Liu, Dr. Xinwang Liu, Dr. Chao Wang, Dr. Yan kong, Dr. Xing Su, Dr. Hongda Tian, Yan Zhao, Zhimin Gao, Weitao Zhou, Rongmao Chen, Hongyun Zhang, Yuyao Zhang, Chang Tang, Huangjing Ni, Lijuan Zhou, Pichao Wang et al. for their kind help, constructive discussions and also the enjoyable time they have brought to me.

I also would like to thank IT staff, Rui Yang, Jun Hu, Yuan Tian, Liju Dong and Ce Zhan for their technical support.

I would like to thank my parents for their spiritual support throughout the last four years.

The comments and advise from the reviewers of this thesis are appreciated.

Finally, China Scholarship Council (CSC) is gratefully acknowledged for providing scholarship.

Bibliography

[1] A. Alavi, M. T. Harandi, and C. Sanderson, "Relational divergence based classification on Riemannian manifolds," in *2013 IEEE Workshop on Applications of Computer Vision*. IEEE, 2013, pp. 111–116.

[2] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Log-euclidean metrics for fast and simple calculus on diffusion tensors," *Magnetic resonance in medicine*, vol. 56, no. 2, pp. 411–421, 2006.

[3] ——, "Log-euclidean metrics for fast and simple calculus on diffusion tensors," *Magnetic Resonance in Medicine*, vol. 56, no. 2, pp. 411–421, 2006. [Online]. Available: http://dx.doi.org/10.1002/mrm.20965

[4] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural computation*, vol. 12, no. 10, pp. 2385–2404, 2000.

[5] A. Ben-David and C. E. Davidson, "Eigenvalue estimation of hyperspectral wishart covariance matrices from limited number of samples," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4384–4396, 2012.

[6] J. W. Bohland, S. Saperstein, F. Pereira, J. Rapin, and L. Grady, "Network, anatomical, and non-imaging measures for the prediction of ADHD diagnosis in individual subjects," *Frontiers in systems neuroscience*, vol. 6, 2012.

[7] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998. [Online]. Available: http://dx.doi.org/10.1023/A:1009715923555

[8] V. Calhoun, T. Adali, G. Pearlson, and J. Pekar, "A method for making group inferences from functional MRI data using independent component analysis," *Human brain mapping*, vol. 14, no. 3, pp. 140–151, 2001.

[9] Y. Chao-Gan and Z. Yu-Feng, "DPARSF: a matlab toolbox for pipeline data analysis of resting-state fMRI," *Frontiers in systems neuroscience*, vol. 4, 2010.

[10] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine learning*, vol. 46, no. 1-3, pp. 131–159, 2002.

[11] G. Cheng and B. C. Vemuri, "A novel dynamic system in the space of spd matrices with applications to appearance tracking," *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 592–615, 2013.

[12] J. Damoiseaux, S. Rombouts, F. Barkhof, P. Scheltens, C. Stam, S. M. Smith, and C. Beckmann, "Consistent resting-state networks across healthy subjects," *Proceedings of the National Academy of Sciences*, vol. 103, no. 37, pp. 13 848–13 853, 2006.

[13] B. de Celis Alonso, S. H. Tobón, P. D. Suarez, J. G. Flores, B. de Celis Carrillo, and E. B. Pérez, "A multi-methodological MR resting state network analysis to assess the changes in brain physiology of children with ADHD," *PLoS ONE*, vol. 6, 9 2014.

[14] S. Dey, A. R. Rao, and M. Shah, "Exploiting the brain's network structure in identifying ADHD subjects," *Frontiers in systems neuroscience*, vol. 6, 2012.

[15] ——, "Attributed graph distance measure for automatic detection of attention deficit hyperactive disordered subjects," *Frontiers in neural circuits*, vol. 8, 2014.

[16] I. L. Dryden, A. Koloydenko, and D. Zhou, "Non-euclidean statistics for covariance matrices, with applications to diffusion tensor imaging," *The Annals of Applied Statistics*, pp. 1102–1123, 2009.

[17] B. Efron and C. Morris, "Multivariate empirical bayes and estimation of covariance matrices," *The Annals of Statistics*, pp. 22–32, 1976.

[18] G. E. Fasshauer, "Positive definite kernels: past, present and future," *Dolomites Research Notes on Approximation*, vol. 4, no. Special Issue on Kernel Functions and Meshless Methods, pp. 21–63, 2011.

[19] S. Fiori, "Extended Hamiltonian learning on Riemannian manifolds: Numerical aspects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 7–21, 2012.

[20] P. T. Fletcher and S. Joshi, "Principal geodesic analysis on symmetric spaces: Statistics of diffusion tensors," in *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis*. Springer, 2004, pp. 87–98.

[21] W. Förstner and B. Moonen, "A metric for covariance matrices," in *Geodesy-The Challenge of the 3rd Millennium*. Springer, 2003, pp. 299–309.

[22] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical Lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[23] S. Gao, I. W. Tsang, and L.-T. Chia, "Sparse representation with kernels," *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 423–434, 2013.

[24] M. G. Genton, "Classes of kernels for machine learning: a statistics perspective," *The Journal of Machine Learning Research*, vol. 2, pp. 299–312, 2002.

[25] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *The Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.

[26] K. Guo, P. Ishwar, and J. Konrad, "Action recognition from video using feature covariance matrices," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2479–2494, 2013.

[27] ——, "Action recognition using sparse representation on covariance manifolds of optical flow," in *IEEE Advanced Video and Signal-based Surveillance*. IEEE, 2010, pp. 188–195. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/AVSS.2010.71

[28] M. Ha Quang, M. San Biagio, and V. Murino, "Log-hilbert-schmidt metric between positive definite operators on hilbert spaces," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 388–396.

[29] M. Harandi, M. Salzmann, and F. Porikli, "Bregman divergences for infinite dimensional covariance matrices," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 1003–1010.

[30] M. T. Harandi, M. Salzmann, and R. Hartley, "From manifold to manifold: geometry-aware dimensionality reduction for spd matrices," in *European Conference on Computer Vision*. Springer, 2014, pp. 17–32.

[31] M. T. Harandi, C. Sanderson, R. Hartley, and B. C. Lovell, "Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach," in *European Conference on Computer Vision*. Springer, 2012, pp. 216–229.

[32] M. T. Harandi, M. Salzmann, and F. M. Porikli, "Bregman divergences for infinite dimensional covariance matrices," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 1003–1010. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.132

[33] M. Harandi, R. Hartley, B. Lovell, and C. Sanderson, "Sparse coding on symmetric positive definite manifolds using bregman divergences," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1–1, 2015.

[34] S. Haykin, *Neural Networks: A Comprehensive Foundation (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2007.

[35] J. P. Hedi Tabia, Hamid Laga and P.-H. Gosselin, "Covariance descriptors for 3D shape matching and retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014.

[36] A. Hendrikse, R. Veldhuis, and L. Spreeuwers, "Eigenvalue correction results in face recognition," in *29th Symp. on Information Theory in the Benelux*. WIC, 2008, pp. 27–35.

[37] S. Huang, J. Li, L. Sun, J. Liu, T. Wu, K. Chen, A. Fleisher, E. Reiman, and J. Ye, "Learning brain connectivity of alzheimers disease by exploratory graphical models," *NeuroImage*, vol. 50, pp. 935–949, 2010.

[38] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban, "Human action recognition using a temporal hierarchy of covariance descriptors on 3D joint locations," in *International Joint Conference on Artificial Intelligence*, 2013. [Online]. Available: http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6869

[39] C. Jack, R. C. Petersen, Y. C. Xu, P. C. OBrien, G. E. Smith, R. J. Ivnik, B. F. Boeve, S. C. Waring, E. G. Tangalos, and E. Kokmen, "Prediction of AD with MRI-based hippocampal volume in mild cognitive impairment," *Neurology*, vol. 52, no. 7, pp. 1397–1397, 1999.

[40] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, "Kernel methods on the Riemannian manifold of symmetric positive definite matrices," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[41] G. Kai, P. Ishwar, and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow," in *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance*. IEEE, 2010, pp. 188–195.

[42] M. Kaiser, "A tutorial in connectome analysis: topological and spatial features of brain networks," *Neuroimage*, vol. 57, no. 3, pp. 892–907, 2011.

[43] S. B. Katwal, J. C. Gore, R. Marois, and B. P. Rogers, "Unsupervised spatiotemporal analysis of fMRI data using graph-based visualizations of self-organizing

maps," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 9, pp. 2472–2483, 2013.

[44] S. S. Keerthi, "Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1225–1229, 2002.

[45] J. Kim, J. R. Wozniak, B. A. Mueller, and W. Pan, "Testing group differences in brain functional connectivity: using correlations or partial correlations?" *Brain connectivity*, vol. 5, no. 4, pp. 214–231, 2015.

[46] R. Kondor and T. Jebara, "A kernel between sets of vectors," in *International Conference on Machine Learning*, 2003, pp. 361–368. [Online]. Available: http://www.aaai.org/Library/ICML/2003/icml03-049.php

[47] J.-Y. Kwok and I.-H. Tsang, "The pre-image problem in kernel methods," *IEEE Transactions on Neural Networks*, vol. 15, no. 6, pp. 1517–1525, 2004.

[48] T. S. Lee, "Image representation using 2d gabor wavelets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 959–971, 1996.

[49] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2003, pp. II–409.

[50] C. Lenglet, M. Rousson, R. Deriche, O. Faugeras, *et al.*, "Statistics on multivariate normal distributions: A geometric approach and its application to diffusion tensor MRI." INRIA Research Report, 2004.

[51] N. Leonardi, J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, M. Schluep, P. Vuilleumier, and D. Van De Ville, "Principal components of functional connectivity: A new approach to study dynamic brain connectivity during rest," *NeuroImage*, vol. 83, pp. 937–950, 2013.

[52] P. Li, Q. Wang, W. Zuo, and L. Zhang, "Log-euclidean kernels for sparse representation and dictionary learning," in *IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 1601–1608.

[53] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *IEEE Conference on Computer Vision and Pattern Recognition workshop*. IEEE, 2010, pp. 9–14.

[54] J. Liu, S. Ji, and J. Ye, *SLEP: Sparse Learning with Efficient Projections*, Arizona State University, 2009. [Online]. Available: http://www.public.asu.edu/~jye02/Software/SLEP

[55] X. Liu, L. Wang, J. Yin, E. Zhu, and J. Zhang, "An efficient approach to integrating radius information into multiple kernel learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 557–569, 2013.

[56] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[57] Q. Mao and I. W. Tsang, "Parameter-free spectral kernel learning," *arXiv preprint arXiv:1203.3495*, 2012.

[58] X. Mestre, "Improved estimation of eigenvalues and eigenvectors of covariance matrices using their sample estimates," *IEEE Transactions on Information Theory*, vol. 54, no. 11, pp. 5113–5129, 2008.

[59] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," in *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*. Cambridge, MA, USA: MIT Press, 1999, pp. 536–542. [Online]. Available: http://dl.acm.org/citation.cfm?id=340534.340729

[60] R. P. Monti, P. Hellyer, D. Sharp, R. Leech, C. Anagnostopoulos, and G. Montana, "Estimating time-varying brain connectivity networks from functional mri time series," *Neuroimage*, vol. 103, pp. 427–443, 2014.

[61] T. Musha, H. Matsuzaki, Y. Kobayashi, Y. Okamoto, M. Tanaka, and T. Asada, "EEG markers for characterizing anomalous activities of cerebral neurons in nat (neuronal activity topography) method," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 8, pp. 2332–2338, 2013.

[62] B. Ng, G. Varoquaux, J. B. Poline, and B. Thirion, "A novel sparse group gaussian graphical model for functional connectivity estimation," in *Information Processing in Medical Imaging*. Springer, 2013, pp. 256–267.

[63] S. W. Nydick, "The wishart and inverse wishart distributions," *Journal Of Statistic*, 2012.

[64] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 716–723.

[65] S. Paisitkriangkrai, C. Shen, and J. Zhang, "Fast pedestrian detection using a cascade of boosted covariance features," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1140–1151, 2008.

[66] Y. Pang, Y. Yuan, and X. Li, "Effective feature extraction in high-dimensional space," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 38, no. 6, pp. 1652–1656, 2008. [Online]. Available: http://dx.doi.org/10.1109/TSMCB.2008.927276

[67] ——, "Gabor-based region covariance matrices for face recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 7, pp. 989–993, 2008.

[68] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian framework for tensor computing," *International Journal of Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.

[69] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The feret evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.

[70] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2006, pp. 728–735. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2006.94

[71] A. Qiu, A. Lee, M. Tan, and M. K. Chung, "Manifold learning on brain functional networks in aging," *Medical image analysis*, vol. 20, no. 1, pp. 52–60, 2015.

[72] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "Simplemkl," *Journal of Machine Learning Research*, vol. 9, pp. 2491–2521, 2008.

[73] M. Ramona, G. Richard, and B. David, "Multiclass feature selection with kernel gram-matrix-based criteria," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 10, pp. 1611–1623, 2012.

[74] T. Randen and J. H. Husoy, "Filtering for texture classification: A comparative study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 291–310, 1999.

[75] Y. Rathi, S. Dambreville, and A. Tannenbaum, "Statistical shape analysis using kernel PCA," in *Electronic Imaging 2006*. International Society for Optics and Photonics, 2006, pp. 60 641–60 641.

[76] J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, B. Greco, P. Hagmann, M. Schluep, P. Vuilleumier, and D. Van De Ville, "Classifying minimally disabled multiple sclerosis patients from resting state functional connectivity," *NeuroImage*, vol. 62, no. 3, pp. 2021–2033, 2012.

[77] M. J. Rosa, L. Portugal, T. Hahn, A. J. Fallgatter, M. I. Garrido, J. Shawe-Taylor, and J. Mourao-Miranda, "Sparse network-based models for patient classification using fmri," *NeuroImage*, vol. 105, pp. 493–506, 2015.

[78] R. Rosipal and N. Krämer, "Overview and recent advances in partial least squares," in *Subspace, latent structure and feature selection*. Springer, 2006, pp. 34–51.

[79] H. Salehian, G. Cheng, B. C. Vemuri, and J. Ho, "Recursive estimation of the stein center of spd matrices and its applications," in *IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 1793–1800.

[80] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[81] N. Shawe-Taylor and A. Kandola, "On kernel target alignment," *Advances in neural information processing systems*, vol. 14, p. 367, 2002.

[82] J. W. Silverstein, "Eigenvalues and eigenvectors of large dimensional sample covariance matrices," *Contemporary Mathematics*, vol. 50, pp. 153–159, 1986.

[83] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Tensor sparse coding for region covariances," in *European Conference on Computer Vision*. Springer, 2010, pp. 722–735.

[84] S. M. Smith, "The future of fMRI connectivity," *Neuroimage*, vol. 62, no. 2, pp. 1257–1266, 2012.

[85] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich, "Network modelling methods for fmri," *Neuroimage*, vol. 54, no. 2, pp. 875–891, 2011.

[86] S. M. Smith, D. Vidaurre, C. F. Beckmann, M. F. Glasser, M. Jenkinson, K. L. Miller, T. E. Nichols, E. C. Robinson, G. Salimi-Khorshidi, M. W. Woolrich, *et al.*, "Functional connectomics from resting-state fMRI," *Trends in cognitive sciences*, vol. 17, no. 12, pp. 666–682, 2013.

[87] O. Sporns, C. J. Honey, and R. Kötter, "Identification and classification of hubs in brain networks," *PloS one*, vol. 2, no. 10, pp. 1–13, 2007.

[88] S. Sra, "Positive definite matrices and the symmetric stein divergence," *arXiv preprint arXiv:1110.1773*, 2011.

[89] N. Städler and P. Bühlmann, "Missing values: sparse inverse covariance estimation and an extension to sparse regression," *Statistics and Computing*, vol. 22, no. 1, pp. 219–235, 2012.

[90] H. Tabia, H. Laga, D. Picard, and P. H. Gosselin, "Covariance descriptors for 3D shape matching and retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 4185–4192. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.533

[91] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in Neural Information Processing Systems*, 2006, pp. 1345–1352. [Online]. Available: http://papers.nips.cc/paper/3078-modeling-human-motion-using-binary-latent-variables

[92] T. Tokuda, B. Goodrich, I. Van Mechelen, A. Gelman, and F. Tuerlinckx, "Visualizing distributions of covariance matrices," *Columbia Univ., New York, USA, Tech. Rep*, pp. 18–18, 2011.

[93] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *European Conference on Computer Vision*. Springer, 2006, pp. 589–600.

[94] ——, "Human detection via classification on riemannian manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2007.383197

[95] ——, "Pedestrian detection via classification on riemannian manifolds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1713–1727, 2008.

[96] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot, "Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the MNI MRI single-subject brain," *Neuroimage*, vol. 15, no. 1, pp. 273–289, 2002.

[97] K. Ugurbil, "Magnetic resonance imaging at ultrahigh fields," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 5, pp. 1364–1379, May 2014.

[98] M. van den Heuvel, R. Mandl, and H. H. Pol, "Normalized cut group clustering of resting-state FMRI data," *PloS one*, vol. 3, no. 4, pp. 1–11, 2008.

[99] V. N. Vapnik and V. Vapnik, *Statistical learning theory*. Wiley New York, 1998, vol. 2.

[100] G. Varoquaux, F. Baronnet, A. Kleinschmidt, P. Fillard, and B. Thirion, "Detection of brain functional-connectivity difference in post-stroke patients using group-level covariance modeling," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2010*. Springer, 2010, pp. 200–208.

[101] A. Veeraraghavan, A. K. Roy-Chowdhury, and R. Chellappa, "Matching shape sequences in video with applications in human movement analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1896–1909, 2005.

[102] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D skeletons as points in a lie group," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 588–595.

[103] R. Vemulapalli, J. K. Pillai, and R. Chellappa, "Kernel learning for extrinsic classification of manifold features," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 1782–1789.

[104] C. Wang, Y. Wang, and A. L. Yuille, "An approach to pose-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 915–922.

[105] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Learning actionlet ensemble for 3D human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 914–927, 2014.

[106] L. Wang, "Feature selection with kernel class separability," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1534–1546, 2008.

[107] L. Wang, P. Xue, and K. L. Chan, "Two criteria for model selection in multiclass support vector machines," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 6, pp. 1432–1448, 2008.

[108] R. Wang, H. Guo, L. S. Davis, and Q. Dai, "Covariance discriminative learning: A natural and efficient approach to image set classification," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2496–2503. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2012.6247965

[109] ——, "Covariance discriminative learning: A natural and efficient approach to image set classification," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2496–2503.

[110] T. Wang, D. Zhao, and S. Tian, "An overview of kernel alignment and its applications," *Artificial Intelligence Review*, pp. 1–14, 2012.

[111] C.-Y. Wee, P.-T. Yap, K. Denny, J. N. Browndyke, G. G. Potter, K. A. Welsh-Bohmer, L. Wang, and D. Shen, "Resting-state multi-spectrum functional connectivity networks for identification of mci patients," *PloS one*, vol. 7, no. 5, pp. 1–11, 2012.

[112] C.-Y. Wee, P.-T. Yap, D. Zhang, L. Wang, and D. Shen, "Constrained sparse functional connectivity networks for MCI classification," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2012*. Springer, 2012, pp. 212–219.

[113] C. Williams and M. Seeger, "Using the Nystrom Method to Speed Up Kernel Machines," in *Advances in Neural Information Processing Systems*. MIT Press, 2001, pp. 682–688.

[114] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, Feb 2009.

[115] Y. Wu, B. Ma, and Y. Jia, "Differential tracking with a kernel-based region covariance descriptor," *Pattern Analysis and Applications*, vol. 18, no. 1, pp. 45–59, 2015. [Online]. Available: http://dx.doi.org/10.1007/s10044-014-0430-6

[116] H. Xiong, M. Swamy, and M. O. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Transactions on Neural Networks*, vol. 16, no. 2, pp. 460–474, 2005.

[117] X. Yang and Y. Tian, "Super normal vector for activity recognition using depth sequences," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 804–811.

[118] X. Yang, H. Kang, A. Newton, and B. Landman, "Evaluation of statistical inference on empirical resting state fMRI," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 4, pp. 1091–1099, April 2014.

[119] C. Yuan, W. Hu, X. Li, S. J. Maybank, and G. Luo, "Human action recognition under log-euclidean riemannian metric," in *Asian Conference on Computer Vision*, 2009, pp. 343–353. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12307-8_32

[120] M. Zanfir, M. Leordeanu, and C. Sminchisescu, "The moving pose: An efficient 3D kinematics descriptor for low-latency action recognition and detection," in *IEEE International Conference on Computer Vision*. IEEE, Dec 2013, pp. 2752–2759.

[121] S. Zhong, D. Chen, Q. Xu, and T. Chen, "Optimizing the gaussian kernel function with the formulated kernel target alignment criterion for two-class pattern classification," *Pattern Recognition*, vol. 46, no. 7, pp. 2045–2054, 2013.

[122] L. Zhou, R. Hartley, L. Wang, P. Lieby, and N. Barnes, "Identifying anatomical shape difference by regularized discriminative direction," *IEEE Transactions on Medical Imaging*, vol. 28, no. 6, pp. 937–950, 2009.

[123] L. Zhou, L. Wang, and P. Ogunbona, "Discriminative sparse inverse covariance matrix: Application in brain functional network classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 3097–3104.

[124] Y. Zhu, W. Chen, and G. Guo, "Fusing spatiotemporal features and joints for 3D action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition workshop*. IEEE, 2013, pp. 486–491.