Distributed Cascaded Manifold Hashing Network for Compact Image Set Representation

Xiaxin Wang[1] , **Haoyu Cai**[1] , **Xiaobo Shen**[1*] , **Xia Wu**[2]
[1]Nanjing University of Science and Technology
[2]Beijing Institute of Technology
wangxiaxin@njust.edu.cn, hycai@njust.edu.cn, njust.shenxiaobo@gmail.com, wuxia@bit.edu.cn

Abstract

Conventional image set methods typically learn from image sets stored in a single location. However, in real-world applications, image sets are often distributed across different locations. Learning from such distributed sets using deep neural networks poses challenges for efficient image set classification and retrieval. To address this, we propose Distributed Cascade Manifold Hashing Network (DCMHN) for compact image set representation. DCMHN represents each image set using an SPD manifold and utilizes a manifold hashing network to generate hash codes, enabling efficient classification and retrieval. The network is trained in a cascaded manner, where the bilinear mapping in the BiMap layer is learned first, followed by joint learning of the hash function and classifier in the hash layer. DCMHN enforces local consistency on global variables across neighboring nodes, allowing parallel optimization. Extensive experiments on three benchmark image set datasets demonstrate that the proposed DCMHN achieves competitive accuracies in distributed settings, and outperforms state-of-the-arts in terms of computation and storage efficiency.

1 Introduction

In recent years, the rapid development of multimedia technology has led to an explosion in the number of images captured by cameras and surveillance videos. As a result, many classification and retrieval tasks now focus on image sets, which consist of multiple images of a specific object [Zhu *et al.*, 2014; Huang *et al.*, 2015a; Huang *et al.*, 2017; Pigou *et al.*, 2018]. Image sets provide richer variability than single images, but due to factors like lighting, angle, posture, and background, the image sets from the same object often exhibit large intra-class variations, making image set classification a challenging task.

Image set modeling is crucial for image set tasks, and various effective methods have been proposed. Affine/convex hull [Cevikalp and Triggs, 2010] and linear subspace methods [Yang *et al.*, 2013; Kim *et al.*, 2007] model image sets but struggle to capture nonlinear structures. Manifold-based methods have gained attention for their strong nonlinear modeling capabilities. These methods can be categorized into three types: Riemannian kernel methods [Wang *et al.*, 2012b; Hamm and Lee, 2008; Wang *et al.*, 2018b; Shen *et al.*, 2024], Riemannian metric learning methods [Huang *et al.*, 2015b; Huang *et al.*, 2015a; Zhu *et al.*, 2018; Lu *et al.*, 2015], and deep manifold networks [Huang and Van Gool, 2017; Huang *et al.*, 2018; Chakraborty *et al.*, 2020; Wang *et al.*, 2021]. Riemannian kernel methods use kernel functions to embed manifolds into a Reproducing Kernel Hilbert Space (RKHS), but they ignore Riemannian geometry. Riemannian metric learning addresses this by using manifold similarity as the Riemannian metric. To enhance modeling, deep manifold networks construct neural networks directly on manifolds. However, these methods learn in continuous space, leading to high computational and storage costs on large-scale datasets, making learning of compact image set representations a continuing challenge.

Conventional image set methods primarily learn from image sets collected at a single location. However, in real-world applications, such as person re-identification [Cao *et al.*, 2022], image sets are often collected across multiple locations. One approach is to centralize the image sets before applying a centralized learning method. However, this incurs high communication costs and may exceed the computational capacity of a single machine [Zhai *et al.*, 2017].

Hashing [Wang *et al.*, 2018a] has attracted extensive interest in large-scale image retrieval due to its advantage of computation and storage. Hashing maps high-dimensional data into low-dimensional hash codes while preserving similarity structure among data. The advantage of hashing motivates us to extend to the application of image retrieval to more challenging image set classification and retrieval, and thus proposes Distributed Cascaded Manifold Hashing Network (DCMHN) to significantly reduce computation and storage costs. The illustration of the proposed DCMHN is shown in Figure 1. The main contributions of this work can be summarized as follows:

- We propose the Distributed Cascaded Manifold Hashing Network (DCMHN) to learn compact image set representations using hash codes, enabling efficient classifi-

*Corresponding author.

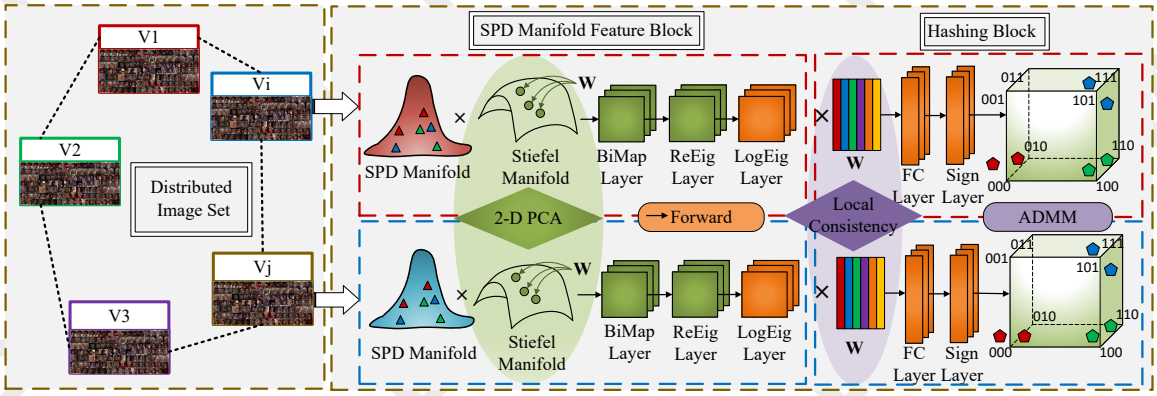


Figure 1: Illustration of the proposed DCMHN. The proposed DCMHN represents each image set using an SPD manifold and utilizes a manifold hashing network, consisting of BiMap, ReEig, and LogEig layers for manifold encoding, and a hash layer to generate hash codes.

cation and retrieval. DCMHN is the first deep neural network approach designed for compact representation learning on distributed image sets.

- The manifold hashing network is used to encode the manifold and is trained in a cascaded manner to reduce training time. DCMHN enforces local consistency across neighboring nodes, enabling parallel optimization.
- We evaluate DCMHN on three benchmark image set datasets, demonstrating competitive accuracy in distributed settings and superior performance in terms of computation and storage efficiency compared to state-of-the-art methods.

2 Related Work

Image Set Classification Conventional image set methods can be mainly divided into five types: affine/convex hull methods, linear subspace methods, Riemannian kernel methods, Riemannian metric learning methods, and deep manifold networks.

Affine/Convex hull methods, e.g., Affine/Convex Hull Based Image Set Distance (AHISD/CHISD) [Cevikalp and Triggs, 2010] characterize an image set by the convex geometric region, i.e., affine or convex hull, and measure the distance between two image sets by geometric distance, i.e., distance of closest approach between convex models. Linear subspace methods, e.g., Mutual Subspace Method (MSM) [Yang *et al.*, 2013] and Discriminant Canonical Correlations (DCC) [Kim *et al.*, 2007] represent an image set as a linear subspace. MSM uses the cosine of the minimum angle between two subspaces as the distance measurement, and DCC aims to seek the projection metric such that canonical correlations in the learned subspace are maximized. Riemannian kernel methods, e.g., Covariance Discriminative Learning (CDL) [Wang *et al.*, 2012b] and Grassmann Discriminant Analysis (GDA) [Hamm and Lee, 2008] map the manifold into a reproducing kernel Hilbert space(RKHS), and then employ the kernel discriminant analysis for discriminant learning. Riemannian metric learning methods, e.g., Log-Euclidean Metric Learning (LEML) [Huang *et al.*, 2015b] and Projection Metric Learning (PML) [Huang *et al.*, 2015a] map the manifold to lower-dimensional discriminative manifold through metric learning. Deep manifold networks, e.g., Symmetric Positive Definite Network (SPDNet) [Huang and Van Gool, 2017] and Grassmann Network (GrNet) [Huang *et al.*, 2018] generalize Euclidean network paradigm to manifold space and perform end-to-end training through matrix backpropagation. Later, [Wang *et al.*, 2022] propose two Riemannian operation modules, i.e., Riemannian batch regularization layer, Riemannian pooling layer to improve performance. In addition, SymNet [Wang *et al.*, 2021] proposes a simple SPD manifold deep learning network that learns multilevel connection matrices and reduces computational complexity.

Hashing Hashing [Liu *et al.*, 2011; Gionis *et al.*, 1999; Wang *et al.*, 2012a; Liu *et al.*, 2012; Norouzi and Fleet, 2011] projects the original high-dimensional data into a compact Hamming space while preserving the similarity structure. Locality Sensitive Hashing (LSH) [Datar *et al.*, 2004] generates hash functions in a random manner. Iterative Quantization (ITQ) [Gong *et al.*, 2012] uses Principal Component Analysis (PCA) to reduce dimension, and then employs orthogonal rotation to reduce quantization error. Supervised Discrete Hashing (SDH) [Shen *et al.*, 2015] assumes that high-quality hash codes are ideal for classification, and then are learned jointly by minimizing quantization losses under a linear classifier framework. Deep Hashing Network (DHN) [Zhu *et al.*, 2016] captures the image representation by a convolutional layer and a pooling layer, and uses cross-entropy loss and quantization loss supervision to generate a compact binary code, which addresses the quantization error issue.

3 Distributed Cascaded Manifold Hashing Network

3.1 Problem Setup

Suppose that we are given a distributed image set dataset $\{\boldsymbol{\mathcal{S}}_i, \mathbf{Y}_i\}_{i=1}^{P}$ that has N samples distributed in P nodes of a network, as shown in Figure 1. In the i-th node, image set features and labels are denoted as $\boldsymbol{\mathcal{S}}_i = \{\mathbf{S}_{ij} \in \mathbb{R}^{l \times n_{ij}}\}_{j=1}^{N_i}$

and $\mathbf{Y}_i \in \mathbb{R}^{C\times N_i}$ respectively, where $\mathbf{S}_{ij}$ and n_{ij} denote feature and number of images of the j-th image set in the i-th node respectively, l is the dimension of image feature, C is the number of classes, N_i denotes the number of image sets in the i-th node, and we have $N = \sum_{i=1}^{P} N_i$. The goal is to learn distributed hashing network and hash code $\boldsymbol{\mathcal{B}} = \{\mathbf{B}_i \in \mathbb{R}^{d\times N_i}\}_{i=1}^{P}$ from such distributed image set to perform efficient image set classification and retrieval, where d is the dimension of hash code.

3.2 Network Structure

SPD manifold Given an image set $\mathbf{S}$, its covariance matrix is defined as:

$$\mathbf{X} = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{s}_i - \mathbf{m})(\mathbf{s}_i - \mathbf{m})^\top \tag{1}$$

where $\mathbf{m}$, $\mathbf{s}_i$, and n denote the mean, the i-th image, and the number of images in $\mathbf{S}$ respectively. To ensure positive, we add a small regularization matrix $\frac{\mathrm{Tr}(\mathbf{X})}{\alpha}\mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix, and parameter α is empirically set to 10^3.

SPD Manifold and Hash Layer As shown in Figure 1, the proposed DCMHN constructs a manifold hashing network to encode manifold to generate hash code. Specifically, DCMHN first employs three SPD manifold layers [Huang and Van Gool, 2017], i.e., BiMap, ReEig, and LogEig layers to encode manifold, and further employs a hash layer to transform the continuous feature into hash code. We assume $\mathbf{X}_i^k \in \mathbb{R}^{d_k\times d_k}$ denotes the input of the k-th layer of the i-th node, and define the layers [Huang and Van Gool, 2017]:

BiMap Layer projects each input SPD matrix into a new one through a bilinear mapping $\mathbf{X}_i^{k+1} = \mathbf{W}_i^{k\top}\mathbf{X}_i^k\mathbf{W}_i^k$, where $\mathbf{W}_i^k \in \mathbb{R}^{d_k\times d_{k+1}}$ denotes transformation matrix, and we have $d_{k+1} < d_k$.

ReEig Layer injects nonlinearity by tuning-up the small positive eigenvalues of each input SPD matrix via $\mathbf{X}_i^{k+1} = \mathbf{U}\max(\epsilon\mathbf{I}, \mathbf{\Sigma})\mathbf{U}^\top$, where $\mathbf{U}$ and $\mathbf{\Sigma}$ are eigenvectors and eigenvalues of $\mathbf{X}_i^k$ respectively, ϵ is a predefined small activation threshold.

LogEig Layer embeds each input SPD matrix into a flat space via $\mathbf{X}_i^{k+1} = \mathbf{U}\log(\mathbf{\Sigma})\mathbf{U}^\top$.

Hash Layer transforms the input using a linear mapping matrix and a nonlinear sign function via $\mathbf{X}_i^{k+1} = \mathrm{sign}(\mathbf{W}_i^{k\top}\mathbf{X}_i^k)$, where sign denotes the sign function.

3.3 Formulation

As illustrated in Figure 1, the proposed DCMHN uses BiMap, ReEig, and LogEig layers for manifold encoding, with a Hash layer to generate hash codes. The ReEig and LogEig layers are parameter-free, while the BiMap and Hash layers contain learnable parameters. This section details the parameter learning process, as summarized in Algorithm 1.

Distributed Bilinear Mapping Learning

Conventional deep network often uses stochastic gradient descent, and further employs chain rule as backpropagation (BP). However, compared to calculating Euclidean gradient, calculating Riemannian gradients in deep manifold network is challenging. As such, a variant of SGD on Stiefel manifold [Huang and Van Gool, 2017] has been developed to train such network, however it leads to very high computational cost, and usually requires hundreds or even thousands iterations to converge. In this work, instead of using BP, we propose to learn bilinear mapping in BiMap layer from the perspective of effective preservation of SPD structure [Wang *et al.*, 2024b].

We consider the i-th node, and denote $\mathbf{X}_i^{n,k} \in \mathbb{R}^{d_k\times d_k}$ as the input SPD matrix of the k-level BiMap layer of the n-th image set, where d_k denotes feature dimension. As defined earlier, BiMap layer transforms the input SPD matrix into a new SPD matrix via the following function via a projection matrix $\mathbf{W}_i^k \in \mathbb{R}^{d_k\times d_{k+1}}$, and it is defined as follows:

$$\mathbf{X}_i^{n,k+1} = \mathbf{W}_i^{k\top}\mathbf{X}_i^{n,k}\mathbf{W}_i^k \tag{2}$$

We have the following theorem related to property of $\mathbf{W}_k$.

Theorem 1. *Given the input SPD matrix $\mathbf{X}_i^{n,k}$, the output matrix $\mathbf{X}_i^{n,k+1}$ is a SPD matrix if $\mathbf{W}_i^k$ is required to be a row full-rank matrix, i.e., $\mathbf{W}_i^{k\top}\mathbf{W}_i^k = \mathbf{I}$.*

Proof. It is easy to prove the output matrix is satisfied with the definition of SPD matrix if transformation matrix is full-rank using simple matrix computation. □

Motivated by the popular idea of Principle Component Analysis (PCA), we propose a simple yet effective solution to learn $\mathbf{W}_i^k$. Specifically, we aim to preserve SPD structure across BiMap layer [Wang *et al.*, 2024b], and employ the following reconstruction loss:

$$\begin{aligned}\min_{\mathbf{W}_i^k}\quad & \sum_{n=1}^{N_i}\left\|\mathbf{X}_i^{n,k} - \mathbf{X}_i^{n,k}\mathbf{W}_i^k\mathbf{W}_i^{k\top}\right\|_F^2 \\ \text{s.t.}\quad & \mathbf{W}_i^{k\top}\mathbf{W}_i^k = \mathbf{I}\end{aligned} \tag{3}$$

where $\mathbf{X}_i^{n,k}$ is first required to be centralized. The above problem can be equivalently transformed to the eigenvalue problem, and $\mathbf{W}_i^k$ is formed by the eigenvectors corresponding to the largest d_{k+1} eigenvalues of sample covariance matrix $\mathbf{C}_i^k = \sum_{n=1}^{N_i}\mathbf{X}_i^{n,k\top}\mathbf{X}_i^{n,k}$.

Note that we are given a distributed task, and we need to consider all the P nodes. To make full of all the distributed manifolds on the P nodes, we propose to calculate the global sample covariance matrix $\mathbf{C}^k = \sum_{i=1}^{P}\mathbf{C}_i^k$, and perform its eigenvalue decomposition. To achieve this, we introduce a parameter server to first receive the local covariance matrices on the P nodes, calculate the global covariance matrix by averaging the local ones. The parameter server finally returns the learned bilinear mapping matrices to the P nodes.

Distributed Hash Code Learning

On the i-th node, each SPD manifold is processed through BiMap, ReEig, and LogEig layers, and we have vectorized Euclidian features $\mathbf{X}_i \in \mathbb{R}^{m\times N_i}$, where m denotes the feature dimension. The feature is then feed into hash layer to obtain hash code. Existing researches [Shen *et al.*, 2015] show that optimal hash codes are expected to have good classification performance. In addition, quantization errors between continuous features and hash codes affect quality of

hash codes. Therefore, we consider all the P nodes, and propose to learn hash codes by jointly minimizing the following simple yet linear classification loss and quantization loss:

$$\min_{\mathbf{V},\mathbf{B}_i,\mathbf{U}} \sum_{i=1}^{P} \left(\|\mathbf{Y}_i - \mathbf{U}^\top \mathbf{B}_i\|_F^2 + \lambda \|\mathbf{U}\|_F^2 + \mu \|\mathbf{B}_i - \mathbf{V}^\top \mathbf{X}_i\|_F^2 \right)$$
$$\text{s.t.} \quad \mathbf{B}_i \in \{-1, 1\}^{d \times N_i} \tag{4}$$

where $\mathbf{U} \in \mathbb{R}^{d \times C}$ denotes the weighted matrix related to the linear classifier, $\mathbf{V} \in \mathbb{R}^{m \times d}$ denotes transformation matrix in hash layer, λ and μ denote regularization parameters. The first term is least square classification loss to improve discrimination of hash code, the second term is regularization term, and the third term is quantization loss between continuous feature and hash code. The binary constraint enforces the continuous feature to approximate hash code, which can handle large-scale applications efficiently.

Inspired by the brain's distributed memory processing [Wen *et al.*, 2018] and brain-inspired coordination strategies [Han *et al.*, 2025; Jia *et al.*, 2025; Wang *et al.*, 2024a], we formulate a distributed optimization framework. The local variable $\mathbf{B}_i$ can be optimized independently in each node, while the global variables $\mathbf{U}$ and $\mathbf{V}$ are shared among all the nodes, making them difficult to optimize. Motivated by block splitting strategy, we introduce a new set of local auxiliary variable in each node, i.e., $\mathbf{U}_i$, $\mathbf{V}_i$, and optimize these new local variables in parallel. To the end, (4) can be rewritten as the following distributed objective function:

$$\min_{\mathbf{V}_i,\mathbf{B}_i,\mathbf{U}_i} \sum_{i=1}^{P} \left(\|\mathbf{Y}_i - \mathbf{U}_i^\top \mathbf{B}_i\|_F^2 + \lambda \|\mathbf{U}_i\|_F^2 + \mu \|\mathbf{B}_i - \mathbf{V}_i^\top \mathbf{X}_i\|_F^2 \right)$$
$$\text{s.t.} \quad \mathbf{B}_i \in \{-1, 1\}^{d \times N_i}, \mathbf{V}_i = \mathbf{V}_j,\ \mathbf{U}_i = \mathbf{U}_j, j \in \mathcal{N}(i) \tag{5}$$

where $\mathcal{N}(i)$ represents the neighbors of the i-th node. In (5), we impose consistency constraint on global variables in neighbor nodes rather than all the nodes by utilizing transitivity property of a connected graph. This optimization problem is non-convex with respect to the variables $\mathbf{B}_i$, $\mathbf{U}_i$, and $\mathbf{V}_i$. Therefore, we divide it into multiple sub-problems and optimize the variables in one node while fixing the variables in other nodes. We use *alternating direction multiplier method* (ADMM) to solve (5). To the end, the augmented Lagrangian function of (5) can be defined as follows:

$$\begin{aligned}\min_{\mathbf{V}_i,\mathbf{B}_i,\mathbf{U}_i} \sum_{i=1}^{P} \Bigg(& \|\mathbf{Y}_i - \mathbf{U}_i^\top \mathbf{B}_i\|_F^2 + \lambda \|\mathbf{U}_i\|_F^2 + \mu \|\mathbf{B}_i - \mathbf{V}_i^\top \mathbf{X}_i\|_F^2 \\ & + \sum_{j \in \mathcal{N}(i)} \mathrm{Tr}\left(\mathbf{\Lambda}_{i,j}^\top (\mathbf{V}_i - \mathbf{V}_j) \right) + \frac{\alpha}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{V}_i - \mathbf{V}_j\|_F^2 \\ & + \sum_{j \in \mathcal{N}(i)} \mathrm{Tr}\left(\mathbf{\Gamma}_{i,j}^\top (\mathbf{U}_i - \mathbf{U}_j) \right) + \frac{\beta}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{U}_i - \mathbf{U}_j\|_F^2 \Bigg) \\ \text{s.t.} \quad & \mathbf{B}_i \in \{-1, 1\}^{d \times N_i}\end{aligned} \tag{6}$$

where $\mathbf{\Lambda}_{i,j}$ and $\mathbf{\Gamma}_{i,j}$ are Lagrangian multipliers for the constraints $\mathbf{V}_i = \mathbf{V}_j$ and $\mathbf{U}_i = \mathbf{U}_j$ respectively, α and β are penalty parameters corresponding to augmented Lagrangian.

Update $\mathbf{V}_i$ By fixing the other variables and removing irrelevant terms, the subproblem related to $\mathbf{V}_i$ is reduced as follows:

$$\mathcal{J}_{\mathbf{V}_i} = \sum_{i=1}^{P} \Bigg(\|\mathbf{B}_i - \mathbf{V}_i^\top \mathbf{X}_i\|_F^2 + \sum_{j \in \mathcal{N}(i)} \mathrm{Tr}\left(\mathbf{\Lambda}_{i,j}^\top (\mathbf{V}_i - \mathbf{V}_j) \right) + \frac{\alpha}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{V}_i - \mathbf{V}_j\|_F^2 \Bigg) \tag{7}$$

ADMM is applied to solve (7) by repeating the following two steps:

$$\begin{cases} \mathbf{V}_i^{(t)} := \arg\min_{\mathbf{V}_i} \mathcal{J}_{\mathbf{V}_i}^{(t-1)}, \\ \mathbf{\Lambda}_{i,j}^{(t)} := \mathbf{\Lambda}_{i,j}^{(t-1)} + \alpha (\mathbf{V}_i^{(t)} - \mathbf{V}_j^{(t)}). \end{cases} \tag{8}$$

where t and $t-1$ represent the t and $t-1$ iterations respectively. We provide the following theorem to simplify the Lagrangian multipliers.

Lemma 1. *The augmented Lagrangian function on $\mathbf{V}_i$, i.e.,* (7) *is equivalent to the following form:*

$$\mathcal{J}_{\mathbf{V}_i} = \sum_{i=1}^{P} \Bigg(\|\mathbf{B}_i - \mathbf{V}_i^\top \mathbf{X}_i\|_F^2 + \mathrm{Tr}(\mathbf{\Lambda}_i^\top \mathbf{V}_i) + \frac{\alpha}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{V}_i - \mathbf{V}_j\|_F^2 \Bigg) \tag{9}$$

where $\mathbf{\Lambda}_i = \sum_{j \in \mathcal{N}(i)} (\mathbf{\Lambda}_{i,j} - \mathbf{\Lambda}_{j,i})$ is the new Lagrangian multiplier.

Proof. The lemma can be proven with straightforward matrix computations. □

As can be seen, (9) has fewer Lagrange multipliers than (7), and computational complexity of ADMM can be greatly reduced. Based on Lemma 1, we have the following theorem to update $\mathbf{V}_i$:

Theorem 2. *For the new augmented Lagrangian, i.e.,* (9)*, we can update $\mathbf{V}_i$ using the following rule:*

$$\begin{cases} \mathbf{V}_i^{(t)} := \left(2\mathbf{X}_i \mathbf{X}_i^\top + \alpha \mathbf{I} \right)^{-1} \Bigg(2\mathbf{X}_i \mathbf{B}_i^\top - \mathbf{\Lambda}_i^{(t-1)} + \alpha \sum_{j \in \mathcal{N}(i)} \mathbf{V}_j^{(t-1)} \Bigg) \\ \mathbf{\Lambda}_i^{(t)} := \mathbf{\Lambda}_i^{(t-1)} + 2\alpha \sum_{j \in \mathcal{N}(i)} (\mathbf{V}_i^{(t)} - \mathbf{V}_j^{(t)}) \end{cases} \tag{10}$$

Proof. The theorem can be proven with straightforward matrix computations. □

Update $\mathbf{B}_i$ By fixing other variables, the subproblem with respect to $\mathbf{B}_i$ is defined as:

$$\min_{\mathbf{B}_i} \sum_{i=1}^{P} \left(\|\mathbf{Y}_i - \mathbf{U}_i^\top \mathbf{B}_i\|_F^2 + \mu \|\mathbf{B}_i - \mathbf{V}_i^\top \mathbf{X}_i\|_F^2 \right)$$
$$\text{s.t.} \quad \mathbf{B}_i \in \{-1, 1\}^{d \times N_i} \tag{11}$$

It is challenging to directly optimize $\mathbf{B}_i$ as discrete constraint is NP hard. Following the widely-used coordinate descent in optimization, *discrete cyclic coordinate descent* (DCC) [Shen *et al.*, 2015] is employed to optimize $\mathbf{B}_i$ bit by bit, and each bit admits a closed-form solution.

Algorithm 1 Distributed Cascade Manifold Hashing Network (DCMHN)

Input: Image sets $\boldsymbol{\mathcal{S}} = \{\boldsymbol{\mathcal{S}}_1, \boldsymbol{\mathcal{S}}_2, \ldots, \boldsymbol{\mathcal{S}}_P, \}$; Labels $\boldsymbol{\mathcal{Y}} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_P\}$; Hash code length d; parameter α, β;
Output: Hash code $\boldsymbol{\mathcal{B}} = \{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_P\}$;

1: Obtain SPD manifold of each image set via (1);
2: Calculate global covariance matrices of all the nodes and perform eigenvalue decomposition, and return the learned bilinear mapping matrix in BiMap layer on each node;
3: Perform eigenvalue correction to introduce non-linearity in ReEig layer;
4: Perform Riemannian computation in LogEig layer;
5: Randomly initialize Lagrangian multipliers $\{\mathbf{\Lambda}_i\}_{i=1}^P$, $\{\mathbf{\Gamma}_i\}_{i=1}^P$, $\{\mathbf{U}_i\}_{i=1}^P$, $\{\mathbf{V}_i\}_{i=1}^P$, and penalty parameters α, β;
6: **for** All nodes simultaneously **do**
7: Update $\{\mathbf{B}_i\}_{i=1}^P$ using DCC algorithm;
8: Update $\{\mathbf{V}_i\}_{i=1}^P$, $\{\mathbf{\Gamma}_i\}_{i=1}^P$ using (10);
9: Update $\{\mathbf{U}_i\}_{i=1}^P$, $\{\mathbf{\Lambda}\}_{i=1}^P$ using (13);
10: **end for**

Update $\mathbf{U}_i$ By fixing the other variables and removing irrelevant terms, the subproblem related to $\mathbf{U}_i$ is reduced as follows:

$$\min_{\mathbf{U}_i} \sum_{i=1}^{P} \Bigg(\|\mathbf{Y}_i - \mathbf{U}_i^\top \mathbf{B}_i\|_F^2 + \lambda \|\mathbf{U}_i\|_F^2 + \sum_{j \in \mathcal{N}(i)} \mathrm{Tr}\left(\mathbf{\Gamma}_{i,j}^\top (\mathbf{U}_i - \mathbf{U}_j)\right) + \frac{\beta}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{U}_i - \mathbf{U}_j\|_F^2 \Bigg) \tag{12}$$

Similar to optimizing $\mathbf{V}_i$, we repeat the following steps to optimize $\mathbf{U}_i$:

$$\begin{cases} \mathbf{U}_i^{(t)} := \left(2\mathbf{B}_i\mathbf{B}_i^\top + (\beta + 2\lambda)\mathbf{I}\right)^{-1} \left(2\mathbf{B}_i\mathbf{Y}_i^\top - \mathbf{\Gamma}_i^{(t-1)} + \beta \sum_{j \in \mathcal{N}(i)} \mathbf{U}_j^{(t-1)}\right) \\ \mathbf{\Gamma}_i^{(t)} := \mathbf{\Gamma}_i^{(t-1)} + 2\beta \sum_{j \in \mathcal{N}(i)} \left(\mathbf{U}_i^{(t)} - \mathbf{U}_j^{(t)}\right) \end{cases} \tag{13}$$

4 Experiments

4.1 Experiment Setting

Datasets Three benchmark image set datasets, i.e., FPHA [Garcia-Hernando *et al.*, 2018], AFEW [Wang *et al.*, 2012b], BBT [Li *et al.*, 2015] are used for experiment. The statistics of the three datasets are summarized in Table 1. For FPHA, a 63-dimensional image feature is extracted for each sequence [Wang *et al.*, 2021]. For AFEW, a 400-dimensional feature is extracted for each frame [Wang *et al.*, 2012b]. For BBT, a 512-dimensional deep feature for each frame is extracted by CNN [Schroff *et al.*, 2015]. The training set of each image set benchmark is equally distributed across all the nodes in the network to construct distributed data. The network includes 3 nodes for FPHA and AFEW, and 4 nodes for BBT.

Datasets	Type	#Samples	#Training	#Testing	#Dim
FPHA	sequence	1150	600	550	63×63
AFEW	video	2118	1747	371	400×400
BBT	video	4667	3268	1399	512×512

Table 1: Statistics of three datasets.

Method	Type	Accuracy				mAP			
		FPHA	AFEW	BBT	Avg	FPHA	AFEW	BBT	Avg
DCC	[L,C,E]	-	0.258	0.939	0.399	-	-	-	-
AHISD	[A,C,E]	-	-	-	-	-	-	-	-
CHISD	[A,C,E]	0.642	-	-	0.214	0.226	-	-	0.075
CDL	[M,C,E]	0.250	0.297	0.876	0.474	0.091	0.200	0.885	0.392
LEML	[M,C,E]	0.762	-	-	0.254	0.368	-	-	0.123
RMML-SPD	[M,C,E]	0.732	0.159	-	0.297	0.026	0.171	-	0.066
SPDNet	[M,C,E]	0.807	0.253	**0.972**	0.677	0.512	0.251	0.935	0.566
SymNet	[M,C,E]	0.810	0.288	0.950	0.682	**0.797**	0.347	0.793	0.646
ITQ-SPD	[M,H,E]	0.299	0.191	0.923	0.471	0.259	0.253	0.923	0.478
SDH-SPD	[M,H,E]	0.769	0.159	0.953	0.627	0.616	0.159	**0.953**	0.576
DCMHN	[M,H,D]	**0.812**	**0.315**	0.966	**0.697**	0.787	**0.415**	0.946	**0.716**

Table 2: The accuracies and mAPs of all the methods on image set classification and retrieval tasks respectively, where [A], [L] and [M] define convex cone, linear subspace and nonlinear manifold methods respectively. [C] and [H] define continuous and hash methods respectively. [E] and [D] define centralized and distributed methods respectively, where '-' indicates that the method is limited by computational efficiency and memory usage and cannot get results on large-scale data sets.

Comparison Methods We compare the proposed method with various advanced image set classification methods, which can be divided into the following four categories: Affine/Convex hull methods including AHISD/CHISD [Cevikalp and Triggs, 2010], Linear subspace methods including DCC [Kim *et al.*, 2007], Nonlinear manifold methods including CDL [Wang *et al.*, 2012b], LEML [Huang *et al.*, 2015b], RMML [Zhu *et al.*, 2018], SPDNet [Huang and Van Gool, 2017], SymNet [Wang *et al.*, 2021], Hashing methods including ITQ [Gong *et al.*, 2012], SDH [Shen *et al.*, 2015]. Specifically, ITQ-SPD and SDH-SPD apply ITQ and SDH to the SPD manifold respectively. The implementations of the baseline methods are kindly provided by the authors. The manifold network in the proposed DCMHN is constructed as $f_b, f_r, f_b, f_r, f_l, f_h$, where f_b, f_r, f_l, f_h denote BiMap, ReEig, LogEig, Hash layers respectively. The dimensions of the two BiMap layers are set to $[20, 10]$, $[70, 35]$, $[80, 40]$ for FPHA, AFEW, and BBT respectively.

Evaluation Metrics Accuracy (Acc) is used as evaluation metric for classification, and mean Average Precision (mAP) and Precision-Recall (PR) curves are used as evaluation metrics for retrieval.

4.2 Performance Evaluation

Acc and mAP The classification and retrieval performances of all the methods on the three datasets are summarized in Table 2. From Table 2, we have the following observations:

- The proposed DCMHN generally achieves the best performance among all the cases, and improves the best

Methods	SPD Manifold		Kernel Matrix		Continuous Feature		Hash Code
	Mem	Red	Mem	Red	Mem	Red	Mem
FPHA	35.71 MB	922×	5.38 MB	750×	4.49 MB	64×	71.88 KB
AFEW	2.53 GB	40000×	28.23 MB	750×	8.27 MB	64×	132.38 KB
BBT	9.11 GB	65536×	116.36 MB	750×	18.23 MB	64×	291.69 KB

Table 3: Storage of different image set representations. 'Mem' represents memory usage, and 'Red' represents the ratio of memory to hash code consumed by this feature.

Methods	FPHA	AFEW	BBT
CDL	0.14	0.25	1.85
RMML	39.24	4847.73	14364.62
SPDNet/SymNet	0.15	0.28	1.97
Hashing	0.01	0.01	0.01

Table 4: Time (in seconds) required for distance calculation by different methods.

Metric	Method	FPHA	AFEW	BBT
Acc	DCMHN-C	0.768	0.285	0.937
	DCMHN	**0.812**	**0.315**	**0.966**
mAP	DCMHN-C	0.564	0.298	0.828
	DCMHN	**0.787**	**0.410**	**0.946**

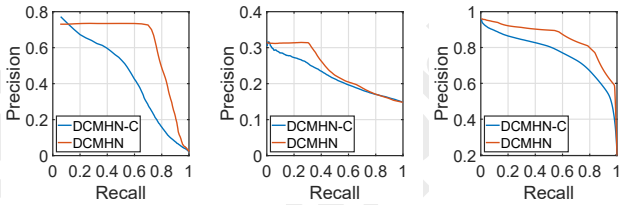
Table 5: The Acc and mAP of the proposed DCMHN and its continuous variant.

baseline averagely by 2% and 7% in terms of Acc and mAP respectively. The empirical results demonstrate the effectiveness of the proposed method on image set modeling.

- Among two deep Riemannian manifold baselines, SymNet outperforms SPDNet. The deep Riemannian manifold baselines outperform the shallow baselines, due to superior capability of the deep architecture. Among hashing baselines, SDH-SPD outperforms ITQ-SPD, as SDH-SPD is supervised.
- Some baselines, e.g., AHISD, CHISH and LEML cannot be well scaled to large-scale datasets due to high computational complexity.

Storage The storage usages for SPD manifold, kernel matrix, continuous feature, and hash code of three datasets are reported in Table 3. As can be observed, the hash code only requires less than 300 KB of storage for the three datasets, while manifold feature requires significantly expensive storage, thus manifold methods cannot be applied to large-scale image set datasets. Particularly, the storage required using hash code is 64 times less than that required for continuous feature, and considerably less compared to other features. The empirical studies demonstrate the significant advantage of hashing in terms of efficient storage usage.

Time The running time associated with distance calculation of different methods on three benchmarks is reported in Table 4, and time of embedding generation is not included. We can find that the running time of hashing is significantly lower than that of the other methods. Specifically, on the AFEW dataset, hashing methods are nearly 280 times faster than those of continuous feature, and significantly faster than manifold methods. Hashing methods, including the proposed DCMHN, which calculate Hamming distance, are theoretically faster than conventional image set methods that calculate Euclidean distance. The empirical studies verify the superiority of the proposed method in terms of efficient distance calculation.



(a) FPHA (b) AFEW (c) BBT

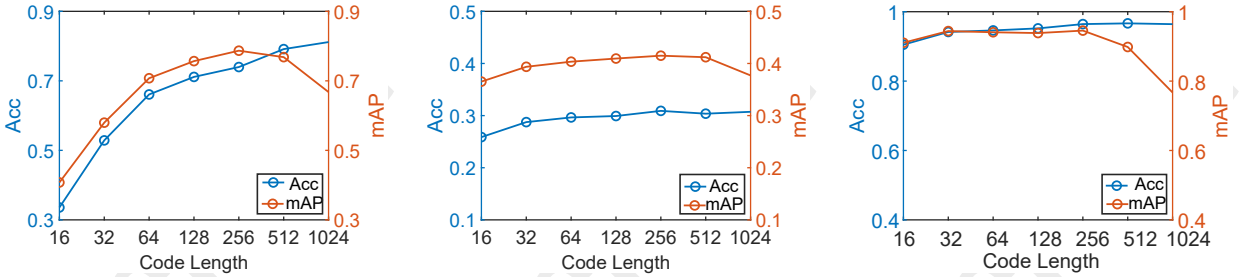
Figure 2: The PR curves of the proposed DCMHN and its continuous variant.

4.3 Further Analysis

Discrete versus Continuous This section compares the performance of the proposed method with its continuous variant that removes binary constraint. The variant is denoted by adding the suffix '-C'. The Acc and mAP of the two methods on the three datasets are shown in Table 5, and the PR curves are shown in Figure 2. We can clearly see that even with binary constraint, the proposed DCMHN outperforms its continuous variant in all the cases, indicating that binary constraint does not degrade performance image set tasks. The empirical studies demonstrate superiority of hash code on modeling image set, and verify the effectiveness of the proposed optimization strategy.

Code Length Analysis This section analyzes the impact of hash code length on the performance of the proposed method. Specifically, the Acc and mAP of the proposed method with respect to different code lengths varying from 16 to 1024 are illustrated in Figure 3. As can be seen, the performance of the proposed method improves as code length increases, and the best performance is generally achieved when code length is set to 256. The above empirical results suggest the optimal hash code length that can achieve good performance.

Node Number Analysis This section analyzes the impact of number of nodes in the network on the performance of the proposed method. The Acc, mAP, and running time of the proposed method with 32 bits on AFEW varying from 2 to 8 nodes are illustrated in Figure 4. As observed, increasing



(a) FPHA (b) AFEW (c) BBT

Figure 3: The Acc and mAP of the proposed DCMHN with respect to different code lengths.

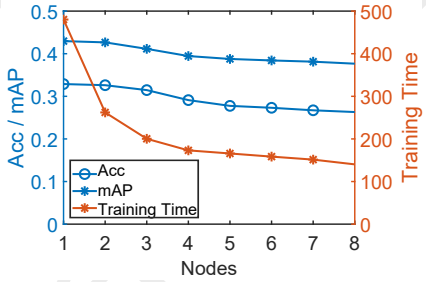


Figure 4: The Acc, mAP, and training time of the proposed DCMHN on AFEW with respect to different numbers of nodes.

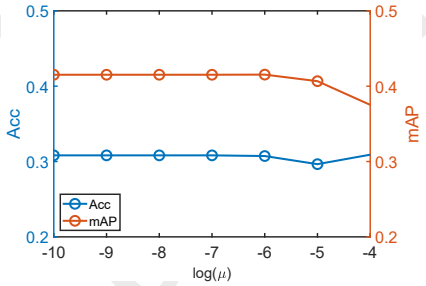
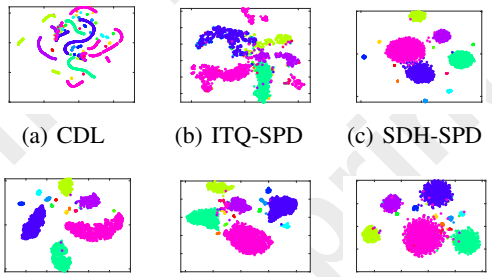


Figure 5: The Acc and mAP of the proposed DCMHN on AFEW with varying μ.

the number of nodes significantly reduces training time, with only a slight decline in accuracy and mAP. Using approximately 3 nodes strikes a balance between performance and training time. The empirical studies confirm that parallel optimization can improve training efficiency.

Trade-off Parameter Analysis This section empirically analyzes the sensitivity of a trade-off parameter, i.e., μ in the proposed DCMHN, where μ is used to balance the relative importance of quantization loss compared to classification loss. The Acc and mAP of the proposed method on AFEW with respect to varying from $[10^{-10}, 10^{-4}]$ are shown in Figure 5. It can be observed that with the decrease of μ, the performance remains stable. The good performance can be achieved when μ is suggested to be less than 10^{-6}.

Visualization This section conducts qualitative empirical study and performs visualization. The visualization results



(a) CDL (b) ITQ-SPD (c) SDH-SPD

(d) SPDNet (e) SymNet (f) DCMHN

Figure 6: The visualization of BBT by different methods. Each point represents an image set, and each color indicates a class.

of six representative methods on BBT are shown in Figure 6, and The widely-used t-SNE is applied to the learned representations. It can be observed that SDH-SPD, SPDNet, and the proposed DCMHN can separate multiple clusters better than the other methods. These qualitative results are consistent with previous quantitative results, intuitively reinforcing the effectiveness of the proposed DCMHN.

5 Conclusion

This paper introduces, for the first time, a Distributed Cascade Manifold Hashing Network (DCMHN) designed to construct a manifold-based network for learning hash codes in a distributed manner. The core idea is to model distributed image sets as a connected graph and build a distributed deep manifold hashing network on this graph, with the network trained in a cascaded fashion. Compared to existing image set methods, the proposed approach offers significant advantages in handling distributed image sets while enhancing computational and storage efficiency. The empirical studies validate our theoretical findings.

Acknowledgments

This work was supported by the National Science Fund for Distinguished Young Scholars of China under Grant No. 62325601, the National Natural Science Foundation of China under Grant No. 62472226, 62176126, the Natural Science Foundation of Jiangsu Province, China under Grant No. BK20230095.

References

[Cao *et al.*, 2022] Yuan Cao, Junwei Liu, Heng Qi, Jie Gui, Keqiu Li, Jieping Ye, and Chao Liu. Scalable distributed hashing for approximate nearest neighbor search. *IEEE Transactions on Image Processing*, 31:472–484, 2022.

[Cevikalp and Triggs, 2010] Hakan Cevikalp and Bill Triggs. Face recognition based on image sets. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2567–2573, 2010.

[Chakraborty *et al.*, 2020] Rudrasis Chakraborty, Jose Bouza, Jonathan H Manton, and Baba C Vemuri. Manifoldnet: A deep neural network for manifold-valued data with applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(2):799–810, 2020.

[Datar *et al.*, 2004] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of Symposium on Computational Geometry*, pages 253–262, 2004.

[Garcia-Hernando *et al.*, 2018] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 409–419, 2018.

[Gionis *et al.*, 1999] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Proceedings of International Conference on Very Large Data Bases*, pages 518–529, 1999.

[Gong *et al.*, 2012] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2012.

[Hamm and Lee, 2008] Jihun Hamm and Daniel D Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proceedings of International Conference on Machine Learning*, pages 376–383, 2008.

[Han *et al.*, 2025] Yuyang Han, Xiuxing Li, Qixin Wang, Tianyuan Jia, and Xia Wu. A brain-inspired distributed long-term memory guided online continual learning method. In *Human Brain and Artificial Intelligence*, pages 331–343, 2025.

[Huang and Van Gool, 2017] Zhiwu Huang and Luc Van Gool. A riemannian network for spd matrix learning. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 2036–2042, 2017.

[Huang *et al.*, 2015a] Zhiwu Huang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Projection metric learning on grassmann manifold with application to video based face recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 140–149, 2015.

[Huang *et al.*, 2015b] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *Proceedings of International Conference on Machine Learning*, pages 720–729, 2015.

[Huang *et al.*, 2017] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Luc Van Gool, and Xilin Chen. Cross euclidean-to-riemannian metric learning with application to face recognition from video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2827–2840, 2017.

[Huang *et al.*, 2018] Zhiwu Huang, Jiqing Wu, and Luc Van Gool. Building deep networks on grassmann manifolds. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 3279–3286, 2018.

[Jia *et al.*, 2025] Tianyuan Jia, Chaoqiong Fan, Qing Li, Ziyu Li, Li Yao, and Xia Wu. A brain-inspired harmonized learning with concurrent arbitration for enhancing motion planning in fuzzy environments. *IEEE Transactions on Fuzzy Systems*, 33(2):631–643, 2025.

[Kim *et al.*, 2007] Tae-Kyun Kim, Josef Kittler, and Roberto Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1005–1018, 2007.

[Li *et al.*, 2015] Yan Li, Ruiping Wang, Shiguang Shan, and Xilin Chen. Hierarchical hybrid statistic based video binary code and its application to face retrieval in tv-series. In *Proceedings of IEEE International Conference and Workshops on Automatic Face and Gesture Recognition*, volume 1, pages 1–8, 2015.

[Liu *et al.*, 2011] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Hashing with graphs. In *Proceedings of International Conference on Machine Learning*, pages 1–8, 2011.

[Liu *et al.*, 2012] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. Supervised hashing with kernels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081, 2012.

[Lu *et al.*, 2015] Jiwen Lu, Gang Wang, Weihong Deng, Pierre Moulin, and Jie Zhou. Multi-manifold deep metric learning for image set classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1137–1145, 2015.

[Norouzi and Fleet, 2011] Mohammad Norouzi and David J Fleet. Minimal loss hashing for compact binary codes. In *Proceedings of International Conference on Machine Learning*, pages 353–360, 2011.

[Pigou *et al.*, 2018] Lionel Pigou, Aäron Van Den Oord, Sander Dieleman, Mieke Van Herreweghe, and Joni Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, 126:430–439, 2018.

[Schroff *et al.*, 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[Shen *et al.*, 2015] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 37–45, 2015.

[Shen *et al.*, 2024] Xiaobo Shen, Wei Wu, Xiaxin Wang, and Yuhui Zheng. Multiple riemannian kernel hashing for large-scale image set classification and retrieval. *IEEE Transactions on Image Processing*, 33:4261–4273, 2024.

[Wang *et al.*, 2012a] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for large-scale search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2393–2406, 2012.

[Wang *et al.*, 2012b] Ruiping Wang, Huimin Guo, Larry S Davis, and Qionghai Dai. Covariance discriminative learning: A natural and efficient approach to image set classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2496–2503, 2012.

[Wang *et al.*, 2018a] Jingdong Wang, Ting Zhang, Jingkuan Song, Nicu Sebe, and Heng Tao Shen. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):769–790, 2018.

[Wang *et al.*, 2018b] Rui Wang, Xiao-Jun Wu, Kai-Xuan Chen, and Josef Kittler. Multiple manifolds metric learning with application to image set classification. In *Proceedings of International Conference on Pattern Recognition*, pages 627–632, 2018.

[Wang *et al.*, 2021] Rui Wang, Xiao-Jun Wu, and Josef Kittler. Symnet: A simple symmetric positive definite manifold deep learning method for image set classification. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):2208–2222, 2021.

[Wang *et al.*, 2022] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler. Learning a discriminative spd manifold neural network for image set classification. *Neural networks*, 151:94–110, 2022.

[Wang *et al.*, 2024a] Qixin Wang, Chaoqiong Fan, Tianyuan Jia, Yuyang Han, and Xia Wu. ND-MRM: neuronal diversity inspired multisensory recognition model. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Proceedings of AAAI Conference on Artificial Intelligence*, pages 15589–15597, 2024.

[Wang *et al.*, 2024b] Rui Wang, Xiaojun Wu, Ziheng Chen, Cong Hu, and Josef Kittler. SPD manifold deep metric learning for image set classification. *IEEE Transactions on Neural Networks and Learning Systems*, 35(7):8924–8938, 2024.

[Wen *et al.*, 2018] Xiaotong Wen, Hailing Wang, Zhenghao Liu, Chenghua Liu, Kang Li, Mingzhou Ding, and Xia Wu. Dynamic top-down configuration by the core control system during working memory. *Neuroscience*, 391:13–24, 2018.

[Yang *et al.*, 2013] Meng Yang, Pengfei Zhu, Luc Van Gool, and Lei Zhang. Face recognition based on regularized nearest points between image sets. In *Proceedings of IEEE International Conference and Workshops on Automatic Face and Gesture Recognition*, pages 1–7, 2013.

[Zhai *et al.*, 2017] Deming Zhai, Xianming Liu, Xiangyang Ji, Debin Zhao, Shin'ichi Satoh, and Wen Gao. Supervised distributed hashing for large-scale multimedia retrieval. *IEEE Transactions on Multimedia*, 20(3):675–686, 2017.

[Zhu *et al.*, 2014] Pengfei Zhu, Wangmeng Zuo, Lei Zhang, Simon Chi-Keung Shiu, and David Zhang. Image set-based collaborative representation for face recognition. *IEEE Transactions on Information Forensics and Security*, 9(7):1120–1132, 2014.

[Zhu *et al.*, 2016] Han Zhu, Mingsheng Long, Jianmin Wang, and Yue Cao. Deep hashing network for efficient similarity retrieval. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 2415–2421, 2016.

[Zhu *et al.*, 2018] Pengfei Zhu, Hao Cheng, Qinghua Hu, Qilong Wang, and Changqing Zhang. Towards generalized and efficient metric learning on riemannian manifold. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 3235–3241, 2018.