Distributed Manifold Hashing for Image Set Classification and Retrieval

Xiaobo Shen[1], Peizhuo Song[1], Yun-Hao Yuan[2], Yuhui Zheng[3,4]*

[1]Nanjing University of Science and Technology
[2]Yangzhou University
[3]Qinghai Normal University
[4]Nanjing University of Information Science and Technology
njust.shenxiaobo@gmail.com, 18362995367@139.com, yyhzbh@163.com, zhengyh@vip.126.com

Abstract

Conventional image set methods typically learn from image sets stored in one location. However, in real-world applications, image sets are often distributed or collected across different positions. Learning from such distributed image sets presents a challenge that has not been studied thus far. Moreover, efficiency is seldom addressed in large-scale image set applications. To fulfill these gaps, this paper proposes Distributed Manifold Hashing (DMH), which models distributed image sets as a connected graph. DMH employs Riemannian manifold to effectively represent each image set and further suggests learning hash code for each image set to achieve efficient computation and storage. DMH is formally formulated as a distributed learning problem with local consistency constraint on global variables among neighbor nodes, and can be optimized in parallel. Extensive experiments on three benchmark datasets demonstrate that DMH achieves highly competitive accuracies in a distributed setting and provides faster classification and retrieval than state-of-the-arts.

Introduction

With the rapid advancement of multimedia technology, there has been a surge in the amount of images captured from cameras or surveillance videos. An image set is considered as a collection of images belonging to a specific object and offers more rich variability information about object than a single-shot image. Image set tasks, e.g., classification, retrieval have attracted increasing attention (Wang et al. 2018b), and showed its great potential (Wang et al. 2012; Huang et al. 2015b). These tasks utilize rich data available in image sets to improve the performance of various applications, from object recognition to information retrieval.

Image set modeling is a critical component in image set tasks, and various effective methods have been proposed, such as affine or convex hull (Cevikalp and Triggs 2010), linear subspace (Kim, Kittler, and Cipolla 2007), covariance matrix (Wang et al. 2012). Given an image set representation, it becomes crucial to measure distance between two sets. Existing image set methods can be broadly divided into four categories based on representation and measurement: linear subspace methods (Yamaguchi, Fukui, and Maeda 1998; Kim, Kittler, and Cipolla 2007), affine/convex hull methods (Cevikalp and Triggs 2010; Hu, Mian, and Owens 2012), kernel methods (Ham and Lee 2008; Wang et al. 2012), Riemannian metric learning methods (Huang et al. 2015a,b). Based on these well-studied Riemannian metrics (Arsigny et al. 2006; Huang et al. 2015a), a variety of image set methods (Huang et al. 2018; Wang et al. 2022; Chen et al. 2023; Wang, Wu, and Kittler 2022) have been proposed to obtain nonlinear feature. Despite the success of existing image set methods, they often perform in a continuous space, which in practice imposes severe challenges in terms of storage and computation cost, particular for large-scale datasets. Therefore, it is imperative to learn compact image set representation, which remains relatively underexplored and presents substantial challenges.

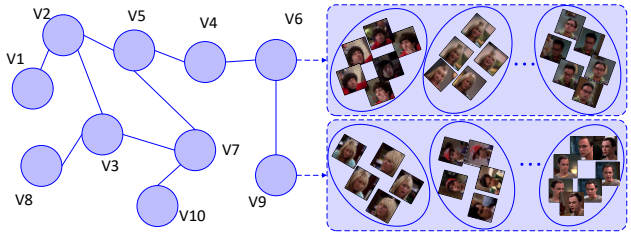


Figure 1: Illustration of an example distributed image set dataset, where a subset of image sets is stored in each node.

In real-world scenarios, e.g., surveillance applications, image sets are often distributed or stored in various locations (Yang et al. 2019; Verbraeken et al. 2021). The distributed image sets can be viewed as an undirected network, as depicted in Figure 1. In this network, each node stores a subset of image sets, and each edge denotes the neighborhood relationship between two nodes. A straightforward approach to handle such distributed image sets is to first gather them in one node and then apply conventional image set methods. However, data gathering inevitably results in prohibitive communication and storage costs. Therefore, it is imperative to develop distributed image set methods, which has not been studied thus before.

Hashing (Wang et al. 2018a; Gao et al. 2023; Kou et al. 2022; Wang et al. 2021) has gained increasing interests in many large-scale applications. Its primary goal is to encode

*Corresponding author.

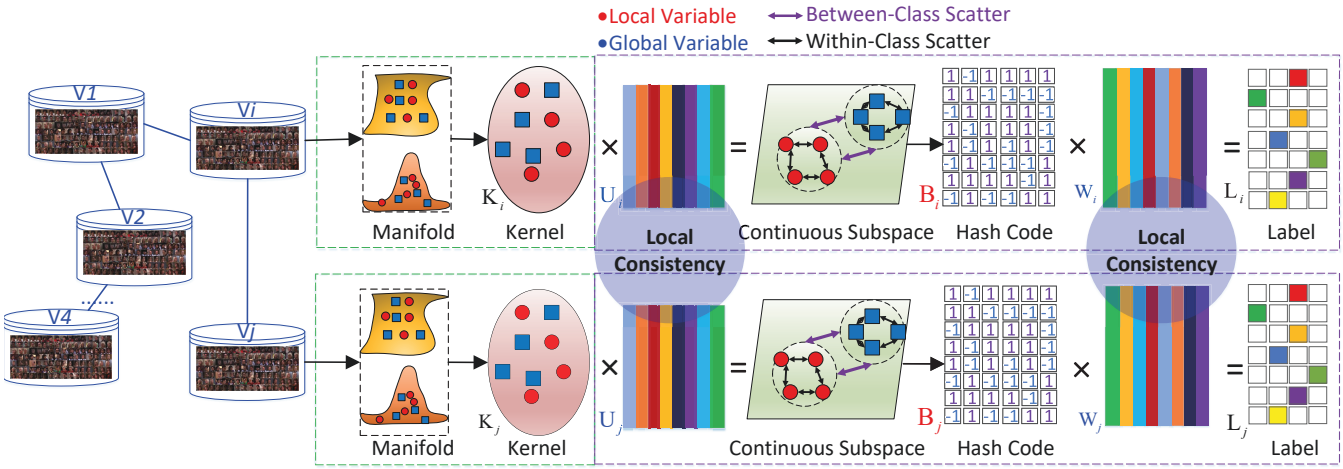


Figure 2: Illustration of the proposed DMH. DMH first employs Riemannian manifold to represent each image set, and further learns hash code on Riemannian manifold kernel to effectively capture intrinsic geometric structure of the image sets. DMH constructs within-class and between-class scatters and jointly learns hash code and classifier to improve discrimination. DMH imposes local consistency constraint on global variables among neighbor nodes, and is optimized in parallel across multiple nodes to improve computational efficiency.

data into a set of short hash codes while preserving similarity. The calculation and storage of hash codes are low in Hamming space, which motivates us to leverage the advantages of hashing to address concerns related to computational and storage cost. As such, we propose a new distributed manifold hashing method (DMH) method for image set classification and retrieval. The overview of DMH is shown in Figure 2. The contributions of this work are summarized as follows:

- We propose a novel distributed manifold hashing (DMH) method for compact image set representation. DMH represents each image set using hash code, significantly reducing computational and storage costs. To our knowledge, DMH is the first attempt at learning compact representation on distributed image sets.
- DMH is formulated as a distributed learning problem with local consistency constraint on global variables among neighbor nodes. It can be optimized in parallel across multiple nodes, leading to a reduction in training time.
- Extensive experiments on three benchmark image set datasets demonstrate the superiority of the proposed DMH over the state-of-the-arts in terms of accuracy and efficiency in a distributed setting.

Related Work

Image Set Classification Traditional image set classification methods can be generally categorized into four types: linear subspace methods, affine/convex hull methods, kernel methods, and Riemannian metric learning methods.

Affine/Convex hull based methods, e.g., Affine/Convex Hull Based Image Set Distance (AHISD/CHISD) (Cevikalp and Triggs 2010) employ the affine/convex hull to model image set. The similarity between two image sets is then determined by measuring the Euclidean distance between their hulls. These methods are computationally expensive due to redundant parameters and high complexity of optimization. Linear subspace methods model each image set as a linear subspace. For example, Mutual Subspace Method (MSM) (Yamaguchi, Fukui, and Maeda 1998) calculates Canonical Correlation (CC) between two linear subspaces as a measurement of distance. Later Discriminant-Analysis of CC (DCC) (Kim, Kittler, and Cipolla 2007) learns a linear mapping by maximizing CCs among within-class sets and minimizing the CCs among between-class sets. To capture nonlinear relationships, kernel methods embed Riemannian manifold into Hilbert space. Grassmann Discriminant Analysis (GDA) (Ham and Lee 2008) first maps the linear subspace on Grassmann manifold into Hilbert space using Grassmann kernel function based on Projection Metric (PM) (Huang et al. 2015a). Covariance Discriminative Learning (CDL) (Wang et al. 2012) utilizes covariance matrix on SPD manifold to model image set and employs Log-Euclidean metric (LEM) (Arsigny et al. 2006) based SPD kernel. Riemannian metric learning methods, e.g., Projection Metric Learning (PML) (Huang et al. 2015a) learn discriminative low-dimensional Riemannian manifold using metric learning.

Existing image set methods primarily focus on learning effective continuous image set representation. The efficiency issue is seldom considered, which pose challenges in terms of high computational and storage costs especially for large-scale image set applications. There have been a few preliminary explorations (Sun et al. 2023, 2022) on learning hash code for image sets, which warrant further research. Furthermore, all the image set methods have not considered cases where image sets are distributed across different positions.

Hashing Hashing (Wang et al. 2018a,a; Gao et al. 2023; Kou et al. 2022; Wang et al. 2021) aims to learn compact hash code in Hamming space while preserving similarity, which can be particularly useful for efficient large-scale search. Many existing hashing methods (Gong et al. 2013; Shen et al. 2015) employ machine learning technique to learn hash function and hash code. For instance, Iterative Quantization (ITQ) (Gong et al. 2013) learns a rotation matrix on data after PCA processing to minimize quantization error. Supervised Discrete Hashing (SDH) (Shen et al. 2015) uses semantics to enhance performance, and jointly learns hash code and a linear classifier. Most existing hashing methods are designed to learn hash code for single images. This work focuses on learning hash code on distributed image sets.

Distributed Manifold Hashing

Problem Setup

Suppose we are given a distributed image set dataset $\{\mathcal{S}_i, \mathcal{Y}_i\}_{i=1}^{P}$ that has N samples distributed in P nodes of a network, as shown in Figure 1. In the i-th node, local image set features and labels are denoted as $\mathcal{S}_i = \{\mathbf{S}_{ij} \in \mathbb{R}^{d\times n_{ij}}\}_{j=1}^{N_i}$ and $\mathcal{Y}_i = \{\mathbf{y}_{ij} \in \mathbb{R}^{C}\}_{j=1}^{N_i}$ respectively, where $\mathbf{S}_{ij}$, $\mathbf{y}_{ij}$, and n_{ij} are image feature matrix, one-hot label vector, and number of images of the j-th image set in the i-th node respectively, d is dimension of image feature, C is number of classes, and N_i is dataset size in the i-th node. The goal of this work is to learn a distributed image set hashing model and hash code $\{\mathbf{B}_i \in \{-1,1\}^{L\times N_i}\}_{i=1}^{P}$ from such distributed image set dataset, by which efficient image set classification and retrieval can be conducted, where L is hash code length.

Riemannian Manifold Representation

SPD manifold For all non-zero vector $\mathbf{v} \in \mathbb{R}^d$, a real Symmetric Positive Definite (SPD) matrix $\mathbf{C}$ has the property of $\mathbf{v}^T\mathbf{C}\mathbf{v} > 0$. $\mathbf{S}_+^d$ forms a convex cone in the $\frac{d(d+1)}{2}$-dimensional Euclidean space. Log-Euclidean Distance (LED) is widely used to measure the geodesic distance on S_+^d. The distance between $\mathbf{C}_i$ and $\mathbf{C}_j$ can be written as:

$$d(\mathbf{C}_i, \mathbf{C}_j) = \| \log \mathbf{C}_i - \log \mathbf{C}_j \|_F \tag{1}$$

where $\log$ is the matrix logarithm operator and $\|.\|_F$ represents matrix Frobenius norm. Under this metric, a Riemannian kernel function can be derived as:

$$k_{log}(\mathbf{C}_i, \mathbf{C}_j) = \mathrm{Tr}\Big(\log \mathbf{C}_i \log \mathbf{C}_j \Big) \tag{2}$$

Grassmann manifold A Grassmann manifold (Edelman, Arias, and Smith 1998) $\mathcal{G}(q,d)$ is spanned by a set of linear subspaces of $\mathbb{R}^{d\times q}$. Each linear subspace is spanned by an orthonormal basis matrix $\mathbf{Y}$ of size $d \times q$ with the constraint $\mathbf{Y}^\top\mathbf{Y} = \mathbf{I}_q$ and can be regarded as an element of $\mathcal{G}(q,d)$. As shown in (Harandi et al. 2013), each element in Grassmann manifold can be represented as a unique projection matrix $\mathbf{Y}\mathbf{Y}^\top$. The Projection Metric (Huang et al. 2015a) can approximate the true geodesic distance on $\mathcal{G}(q,d)$. For any pair of $\mathbf{Y}_i\mathbf{Y}_i^\top$, $\mathbf{Y}_j\mathbf{Y}_j^\top$, this metric can be written as:

$$d\left(\mathbf{Y}_i\mathbf{Y}_i^\top \mathbf{Y}_j\mathbf{Y}_j^\top\right) = \frac{1}{\sqrt{2}}\|\mathbf{Y}_i\mathbf{Y}_i^\top - \mathbf{Y}_j\mathbf{Y}_j^\top\|_F \tag{3}$$

where $\|\cdot\|_F$ represents the matrix Frobenius norm. A well-defined Grassmann kernel (Ham and Lee 2008) can be induced by:

$$k_{\mathrm{p}}\left(\mathbf{Y}_i\mathbf{Y}_i^\top, \mathbf{Y}_j\mathbf{Y}_j^\top\right) = \mathrm{Tr}\Big(\left(\mathbf{Y}_i\mathbf{Y}_i^\top\right)\left(\mathbf{Y}_j\mathbf{Y}_j^\top\right)\Big) = \|\mathbf{Y}_i^\top\mathbf{Y}_j\|_F^2 \tag{4}$$

where $\mathrm{Tr}(\cdot)$ denotes trace of square matrix.

Formulation

We employ Riemannian manifold representation to represent each image set due to superior capability of image set modeling of manifold. In the i-th node, $\mathbf{X}_i$ is denoted as general manifold representation matrix of all image sets, and $\mathbf{x}_{ij}$ is denoted as manifold representation vector of the j-th image set, where SPD and Grassmann manifolds can be chosen. With great success of kernel learning, we consider to map original Representation into a high dimensional Hilbert space, where subspace learning can be performed. Specifically, nonlinear mapping can be defined as $\phi : \mathbf{x} \to \phi(\mathbf{x})$, where $\phi(\mathbf{x})$ denotes representation of $\mathbf{x}$ of Hilbert space. Now we consider the case in the i-th node, and the distance between two image sets, i.e., i.e., $\mathbf{x}_{ip}$ and $\mathbf{x}_{iq}$ in the i-th node $d(\mathbf{x}_{ip}, \mathbf{x}_{iq})$ is defined as:

$$\mathrm{Tr}\left(\left(\phi(\mathbf{x}_{ip}) - \phi(\mathbf{x}_{iq})\right)^\top \mathbf{V}^\top\mathbf{V}\left(\phi(\mathbf{x}_{ip}) - \phi(\mathbf{x}_{iq})\right)\right) \tag{5}$$

where $\mathbf{V}$ denotes projection matrix defined in Hilbert space. As ϕ is nonlinear and implicit, it is difficult to compute $\mathbf{x}_i^w$ and $\mathbf{x}_i^b$. With idea of kernel learning, $\mathbf{V}$ can be regarded as linear combinations of the samples in the i node, i.e., $\mathbf{V} = \phi(\mathbf{X}_i)\mathbf{U}$, where $\mathbf{U}$ denotes linear coefficient matrix, and we have $\mathbf{V}^\top\phi(\mathbf{s}_{ip}) = \mathbf{U}^\top\phi(\mathbf{X}_i)^\top\phi(\mathbf{x}_{ip}) = \mathbf{U}^\top\mathbf{K}_{ip}$, where $\mathbf{K}_{ip}$ is the p-th column of kernel matrix $\mathbf{K}_i$ induced by specific in the i node.

To improve discrimination of the learned representation, a straightforward approach is to maximize distances of image sets in different classes and minimize distances of image sets in the same classes. With this idea, we have the following objective function using kernel trick:

$$\min_{\mathbf{U}} \sum_{i=1}^{P}\Big(\mathrm{Tr}(\mathbf{U}^\top\mathbf{R}_i^w\mathbf{U}) - \alpha\mathrm{Tr}(\mathbf{U}^\top\mathbf{R}_i^b\mathbf{U})\Big) \tag{6}$$

where α is a regularization parameter to balance two terms, $\mathbf{R}_i^w$ and $\mathbf{R}_i^b$ denote within-class and between-class scatters in the i-th node, which are specifically defined as:

$$\begin{cases} \mathbf{R}_i^w = \frac{1}{M_i^w}\sum\limits_{p=1}^{N_i}\sum\limits_{\mathbf{y}_{ip}=\mathbf{y}_{iq}}(\mathbf{K}_{ip} - \mathbf{K}_{iq})(\mathbf{K}_{ip} - \mathbf{K}_{iq})^\top \\ \mathbf{R}_i^b = \frac{1}{M_i^b}\sum\limits_{p=1}^{N_i}\sum\limits_{\mathbf{y}_{ip}\neq\mathbf{y}_{iq}}(\mathbf{K}_{ip} - \mathbf{K}_{iq})(\mathbf{K}_{ip} - \mathbf{K}_{iq})^\top \end{cases} \tag{7}$$

where M_i^w and M_i^b are numbers of sample pairs from the same class and different classes in node i respectively.

Existing image set methods perform classification or retrieval by first finding nearest neighbors in Euclidean space. However, this approach suffers from high storage and computational costs on large-scale datasets. To address these efficiency issues, we propose performing classification and retrieval in Hamming space, leveraging superior capability of hashing to handle large-scale data efficiently. We propose to employ compact hash code to represent each image set, and further minimize least square classification loss to improve discrimination. To this end, the hash code learning framework on the P nodes is defined as follows:

$$\begin{aligned}&\min \sum_{i=1}^{P}\left(\|\mathbf{Y}_i-\mathbf{W}^\top\mathbf{B}_i\|_F^2+\upsilon\|\mathbf{B}_i-\mathbf{U}^\top\mathbf{K}_i\|_F^2+\beta\|\mathbf{W}\|_F^2\right)\\&\text{s.t. }\mathbf{B}_i\in\{-1,1\}^{L\times N_i}\end{aligned}\tag{8}$$

where $\mathbf{B}_i$ and $\mathbf{Y}_i$ denote hash code and label of the i-th node, $\mathbf{W}$ denotes classification weight matrix, β and υ are two nonnegative regularization parameters. The first term is classification loss, the second term is quantization loss between hash code and continuous feature, and the third term is a regularizer. By Combining (6) and (8), we have the following loss function:

$$\begin{aligned}\min_{\mathbf{B}_i,\mathbf{U},\mathbf{W}}\ &\sum_{i=1}^{P}\Big(\mathrm{Tr}(\mathbf{U}^\top\mathbf{R}_i^w\mathbf{U})-\alpha\mathrm{Tr}(\mathbf{U}^\top\mathbf{R}_i^b\mathbf{U})\\&+\|\mathbf{Y}_i-\mathbf{W}^\top\mathbf{B}_i\|^2+\upsilon\|\mathbf{B}_i-\mathbf{U}^\top\mathbf{K}_i\|_F^2+\beta\|\mathbf{W}\|_F^2\Big)\\\text{s.t. }&\mathbf{B}_i\in\{-1,1\}^{L\times N_i}\end{aligned}\tag{9}$$

It is a distributed learning problem. The local variable $\mathbf{B}_i$ can be optimized individually in each node. However, global variables $\mathbf{U}$ and $\mathbf{W}$ are shared among all the nodes, and thus difficult to optimize. Motivated by block splitting strategy, we introduce a new set of local auxiliary variables, i.e., $\mathbf{U}_i$ and $\mathbf{W}_i$, and optimize the new local variables in parallel. To this end, (9) can be transformed to the following final distributed objective function:

$$\begin{aligned}\min_{\mathbf{B}_i,\mathbf{U}_i,\mathbf{W}_i}\ &\sum_{i=1}^{P}\Big(\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^w\mathbf{U}_i)-\alpha\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^b\mathbf{U}_i)\\&+\|\mathbf{Y}_i-\mathbf{W}_i^\top\mathbf{B}_i\|_F^2+\upsilon\|\mathbf{B}_i-\mathbf{U}_i^\top\mathbf{K}_i\|_F^2+\beta\|\mathbf{W}_i\|_F^2\Big)\\&\text{s.t. }\mathbf{B}_i\in\{-1,1\}^{L\times N_i},\mathbf{U}_i=\mathbf{U}_j,\ \mathbf{W}_i=\mathbf{W}_j,\ j\in\mathcal{N}(i)\end{aligned}\tag{10}$$

where $\mathcal{N}(i)$ represents the neighbors of the i-th node in the network. In (10), we impose consistency constraint on global variables *in neighbor nodes rather than all the nodes* by utilizing transitivity property of a connected graph.

Optimization

The optimization problem is non-convex with variables $\mathbf{B}_i$, $\mathbf{U}_i$, and $\mathbf{W}_i$. In this work, we divide it into several subproblems, and optimize one set of variables in one node while fixing variables in other nodes. We employ the well-known ADMM to solve (10) as ADMM can decompose variables and converge fast. The augmented Lagrangian of (10) can be written as follows:

$$\begin{aligned}\mathcal{J}=&\sum_{i=1}^{P}\Big(\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^w\mathbf{U}_i)-\alpha\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^b\mathbf{U}_i)\\&+\|\mathbf{Y}_i-\mathbf{W}_i^\top\mathbf{B}_i\|_F^2+\upsilon\|\mathbf{B}_i-\mathbf{U}_i^\top\mathbf{K}_i\|_F^2+\beta\|\mathbf{W}_i\|_F^2\\&+\sum_{j\in\mathcal{N}(i)}\mathrm{Tr}(\mathbf{\Lambda}_{i,j}^\top(\mathbf{U}_i-\mathbf{U}_j))+\frac{\rho_1}{2}\sum_{j\in\mathcal{N}(i)}\|\mathbf{U}_i-\mathbf{U}_j\|_F^2\\&+\sum_{j\in\mathcal{N}(i)}\mathrm{Tr}(\mathbf{\Gamma}_{i,j}^\top(\mathbf{W}_i-\mathbf{W}_j))+\frac{\rho_2}{2}\sum_{j\in\mathcal{N}(i)}\|\mathbf{W}_i-\mathbf{W}_j\|_F^2\Big)\end{aligned}\tag{11}$$

where $\mathbf{\Lambda}_{i,j}$ and $\mathbf{\Gamma}_{i,j}$ are Lagrangian multipliers for the constraint $\mathbf{U}_i=\mathbf{U}_j$ and $\mathbf{W}_i=\mathbf{W}_j$ respectively, ρ_1 and ρ_2 are penalty parameters of augmented Lagrangian.

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

$$\begin{aligned}\min_{\mathbf{B}_i}\ &\|\mathbf{Y}_i-\mathbf{W}_i^\top\mathbf{B}_i\|_F^2+\upsilon\|\mathbf{B}_i-\mathbf{U}_i^\top\mathbf{K}_i\|_F^2\\\text{s.t. }&\mathbf{B}_i\in\{-1,1\}^{L\times N_i}\end{aligned}\tag{12}$$

It is challenging to directly optimize $\mathbf{B}_i$ as discrete constraint is NP hard. Following the widely-used coordinate descent in optimization, We employ *discrete cyclic coordinate descent* (DCC) to optimize $\mathbf{B}_i$ bit by bit. At each iteration, DCC determines one bit as a coordinate, then iteratively minimizes a subproblem with respect to each bit, which admits a closed-form solution.

Theorem 1. *For the problem of optimizing hash code, i.e.,* (12)*, we can learn $\mathbf{B}_i$ by iteratively updating each bit using the following rule*

$$\mathbf{b}_i^l=sign\left(\mathbf{q}_i^l-\mathbf{B}_i^{l'\top}\mathbf{W}_i^{l'}\mathbf{w}_i^l\right)\tag{13}$$

where $\mathbf{b}_i^{l\top}$ is the l-th row of $\mathbf{B}_i$, $\mathbf{B}_i^{l'}$ is remaining matrix in $\mathbf{B}_i$ except $\mathbf{b}_i^{l\top}$, $\mathbf{Q}_i=\mathbf{W}_i\mathbf{Y}_i+\upsilon\mathbf{U}_i^\top\mathbf{K}_i$, $\mathbf{w}_i^{l\top}$ and $\mathbf{q}_i^{l\top}$ are the l-th row of $\mathbf{W}_i$ and $\mathbf{Q}_i$ respectively, and $\mathbf{W}_i^{l'}$ is remaining matrix in $\mathbf{W}_i$ except $\mathbf{w}_i^{l\top}$.

Proof. The solution of the problem is simple to obtain. The proof of this theorem can be adapted from (Shen et al. 2015). □

Update $\mathbf{U}_i$ After removing all the terms irrelevant to $\mathbf{U}_i$, (11) is reduced to:

$$\begin{aligned}\mathcal{J}_{\mathbf{U}_i}=&\sum_{i=1}^{P}\Big(\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^w\mathbf{U}_i)-\alpha\mathrm{Tr}(\mathbf{U}_i^\top\mathbf{R}_i^b\mathbf{U}_i)\\&+\upsilon\|\mathbf{B}_i-\mathbf{U}_i^\top\mathbf{K}_i\|_F^2+\sum_{j\in\mathcal{N}(i)}\mathrm{Tr}(\mathbf{\Lambda}_{i,j}^\top(\mathbf{U}_i-\mathbf{U}_j))\\&+\frac{\rho_1}{2}\sum_{j\in\mathcal{N}(i)}\|\mathbf{U}_i-\mathbf{U}_j\|_F^2\Big)\end{aligned}\tag{14}$$

ADMM solves (14) by repeating the following two steps:

$$\begin{cases}\mathbf{U}_i^{(t)}=\underset{\mathbf{U}_i}{\mathrm{argmin}}\,\mathcal{J}_{\mathbf{U}_i}^{(t-1)}\\\mathbf{\Lambda}_{i,j}^{(t)}=\mathbf{\Lambda}_{i,j}^{(t-1)}+\rho_1\left(\mathbf{U}_i^{(t-1)}-\mathbf{U}_j^{(t-1)}\right)\end{cases}\tag{15}$$

Datasets	#Samples	#Classes	#Training	#Testing	#Dim
BBT	4,667	15	3,268	1,399	512
PB	9,435	20	6,602	2,828	512
YTC	1,856	47	1,484	372	900

Table 1: Statistics of three datasets.

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

Lemma 1. *The augmented Lagrangian with respect to* $\mathbf{U}_i$*, i.e.,* (14) *is equivalent to the following form:*

$$\begin{aligned}\mathcal{J}_{\mathbf{U}_i} = \sum_{i=1}^{P} \Big(& \mathrm{Tr}(\mathbf{U}_i^\top \mathbf{R}_i^w \mathbf{U}_i) - \alpha \mathrm{Tr}(\mathbf{U}_i^\top \mathbf{R}_i^b \mathbf{U}_i) \\ & + \upsilon \|\mathbf{B}_i - \mathbf{U}_i^\top \mathbf{K}_i\|_F^2 + \mathrm{Tr}(\mathbf{\Lambda}_i^\top \mathbf{U}_i) \\ & + \frac{\rho_1}{2} \sum_{j \in \mathcal{N}(i)} \|\mathbf{U}_i - \mathbf{U}_j\|_F^2 \Big)\end{aligned} \tag{16}$$

where a new Lagrangian multiplier $\mathbf{\Lambda}_i = \sum_{j \in \mathcal{N}(i)} (\mathbf{\Lambda}_{i,j} - \mathbf{\Lambda}_{j,i})(i \in \{1, 2, ...P\})$.

In (14), there are totally P_κ Lagrangian multipliers if κ denotes the average number of neighbors of each node in the network. In (16), a Lagrangian multiplier is assigned to each node in the network, and thus there are only P Lagrangian multipliers. The number of Lagrangian multipliers has been reduced, and computational complexity of ADMM can be greatly reduced by optimizing (16). Based on Lemma 1, we have the following theorem to update $\mathbf{U}_i$:

Theorem 2. *For the new augmented Lagrangian, i.e.,* (16)*, we can update* $\mathbf{U}_i$ *by using the following updating rule:*

$$\begin{cases} \mathbf{U}_i^{(t)} = \Big(\left(\mathbf{R}_w + \mathbf{R}_w^\top\right) - \alpha \left(\mathbf{R}_b + \mathbf{R}_b^\top\right) + 2\upsilon \mathbf{K}_i \mathbf{K}_i^\top \\ \quad + \sum_{j \in \mathcal{N}(i)} \rho_1 \Big)^{-1} \Big(\sum_{j \in \mathcal{N}(i)} \rho_1 \mathbf{U}_j^{(t-1)} - \mathbf{\Lambda}_i^{(t-1)} + 2\upsilon \mathbf{K}_i \mathbf{B}_i^\top \Big) \\ \mathbf{\Lambda}_i^{(t)} = \mathbf{\Lambda}_i^{(t-1)} + 2\rho_1 \sum_{j \in \mathcal{N}(i)} \left(\mathbf{U}_i^{(t-1)} - \mathbf{U}_j^{(t-1)} \right) \end{cases} \tag{17}$$

Update $\mathbf{W}_i$ Similar to optimizing $\mathbf{U}_i$, we first define new Lagrangian multiplier $\mathbf{\Gamma}_i = \sum_{j \in \mathcal{N}(i)} (\mathbf{\Gamma}_{i,j} - \mathbf{\Gamma}_{j,i})$, and then repeat the following steps to optimize $\mathbf{W}_i$:

$$\begin{cases} \mathbf{W}_i^{(t)} = \Big(2\mathbf{B}_i \mathbf{B}_i^\top + 2\beta + \sum_{j \in \mathcal{N}(i)} \rho_2 \Big)^{-1} \\ \qquad \Big(\sum_{j \in \mathcal{N}(i)} \mathbf{W}_j^{(t-1)} - \mathbf{\Gamma}_i^{(t-1)} + 2\mathbf{B}_i \mathbf{Y}_i^\top \Big) \\ \mathbf{\Gamma}_i^{(t)} = \mathbf{\Gamma}_i^{(t-1)} + 2\rho_2 \sum_{j \in \mathcal{N}(i)} \left(\mathbf{W}_i^{(t-1)} - \mathbf{W}_j^{(t-1)} \right) \end{cases} \tag{18}$$

Testing Procedure

Once training of the proposed method is complete, we randomly select a local hash function trained on the i-th node. Given a testing image set, i.e., $\mathbf{S}_q$, we obtain hash code via $\mathbf{b}_q = \mathrm{sign}(\mathbf{U}_i \mathbf{K}_q)$, where $\mathbf{K}_q$ denotes its kernel representation. We calculate Hamming distances between testing image set $\mathbf{b}_q$ and database $\mathbf{B}$, and obtain its ranking list $\mathcal{L}_q$ by sorting these Hamming distances. For image set classification, we predict label by nearest neighbor method. For image set retrieval, we return ranking list $\mathcal{L}_q$. As Hamming distance computation is much efficient than Euclidean distance computation, the proposed DMH is theoretically faster than conventional continuous image set methods during classification and retrieval stages.

Methods	Type	Accuracy			mAP		
		BBT	PB	YTC	BBT	PB	YTC
MSM	[L, C, E]	87.56	82.00	70.92	63.82	32.38	26.43
DCC	[L, C, E]	93.85	69.76	69.89	61.60	29.00	22.26
AHISD	[A, C, E]	-	-	-	-	-	-
CHISD	[A, C, E]	-	-	-	-	-	-
GDA	[M, C, E]	97.71	92.72	66.13	94.73	63.22	72.91
CDL	[M, C, E]	87.56	52.55	70.70	88.49	57.75	14.92
PML	[M, C, E]	95.42	84.47	72.04	72.31	36.89	22.85
LEML	[M, C, E]	-	-	-	-	-	-
ITQ-GM	[M, H, E]	95.57	71.11	60.48	59.79	30.31	23.76
ITQ-SPD	[M, H, E]	95.21	72.21	67.47	63.57	34.76	25.13
SDH-GM	[M, H, E]	98.21	87.16	75.27	93.20	88.23	74.23
SDH-SPD	[M, H, E]	95.93	90.59	74.19	96.00	91.40	75.65
DMH-GM	[M, H, D]	96.38	96.19	73.39	**99.14**	89.02	63.97
DMH-SPD	[M, H, D]	**98.72**	**98.03**	**76.48**	97.30	**96.69**	**79.37**

Table 2: The accuracies and mAPs of all the methods on image set classification and retrieval tasks respectively. [L], [A], and [M] denote linear subspace, affine hull, and nonlinear manifold methods respectively; [C] and [H] denote continuous and hashing methods respectively; [E] and [D] denote centralized and distributed methods respectively. '-' denotes no available result in this case as running time of the method exceeds one week.

Experiments

Datasets

To our knowledge, three large-scale image set datasets, i.e., BBT (Li et al. 2015), PB (Li et al. 2015), YTC (Kim et al. 2008) are used for experiment. The statistics of the three datasets are summarized in Table 1. For YTC, each image is first resized into a 30×30 grayscale image, and histogram equalization is used to eliminate lighting effects as done in (Wang et al. 2012). For BBT and PB, a 512-dimensional deep feature for each image is extracted by a CNN (Schroff, Kalenichenko, and Philbin 2015) that is specifically designed for general still face recognition task.

Experiment Setting

Comparison Methods We compare the proposed method with various state-of-the-art image set classification methods that fall into four categories: Linear subspace methods, including MSM (Yamaguchi, Fukui, and Maeda 1998), DCC (Kim, Kittler, and Cipolla 2007); Affine/Convex hull methods, including AHISD/CHISD (Cevikalp and Triggs 2010);

Datasets	Original Feature		Manifold Feature		Kernel Matrix		Continuous Feature		Hash Code
	Mem.	Red.	Mem.	Red.	Mem.	Red.	Mem.	Red.	Mem.
BBT	1.81 GB	104110×	9.11 GB	524288×	27.54 MB	1547×	1.14 MB	64×	18.23 KB
PB	2.94 GB	83635×	18.42 GB	524288×	111.79 MB	3105×	2.30 MB	64×	36.86 KB
YTC	859.71 MB	121427×	11.19 GB	1620000×	6.02 MB	850×	0.45 MB	64×	7.25 KB

Table 3: Storage of several image set representation. 'Mem.' denotes memory usage, 'Red.' denotes memory reduction (×) of other representations over hash code.

Methods	BBT	PB	YTC
DCC/MSM	2875.74	11743.71	347.2
GDA/CDL	0.0730	0.2717	0.0126
PML	6.46	20.70	1.61
Hashing	0.0093	0.0352	0.0013

Table 4: Running time (in seconds) of distance calculation of several methods.

Nonlinear manifold methods, including GDA (Ham and Lee 2008), CDL (Wang et al. 2012), PML (Huang et al. 2015a), LEML (Huang et al. 2015b); Hashing methods, including ITQ (Gong et al. 2013), SDH (Shen et al. 2015). The implementations of the baseline methods are kindly provided by authors. As all the baseline methods can only handle centralized data, they are trained on data stored in single node. We apply the proposed DMH on SPD and Grassmann manifolds respectively, which are denoted as DMH-SPD and DMH-GM respectively. For the proposed method, we equally distribute the training set across all four nodes in the network to construct distributed data.

Evaluation Metrics We evaluate performance on two image set-related tasks, i.e., image set classification and retrieval. Accuracy and mean Average Precision (mAP) are used as evaluation metrics for classification and retrieval tasks respectively.

Performance Evaluation

Image Set Classification The classification accuracies of all the methods on the three datasets are summarized in Table 2. We find some interesting points:

- The proposed methods achieve the highest classification accuracies among all the cases, and DMH-SPD outperforms DMH-GM. DMH-SPD improves accuracies of the best baselines by 0.51%, 5.31%, and 1.21% on BBT, PB, and YTC respectively.
- Among nonlinear manifold baselines, GDA performs the best on BBT and PB, and CDL performs the best on YTC. Among conventional hashing baselines, SDH-GM outperforms ITQ, as SDH is supervised.
- AHISD, CHISD, and LEML cannot scale to large-scale datasets due to high computational complexity, and their accuracies are not reported.

Image Set Retrieval Table 2 reports mAP results of all the methods on image set retrieval of the three datasets. We observe that retrieval results are generally consistent with classification results. The proposed methods obtain the highest mAPs among all the cases, and DMH-SPD outperforms DMH-GM on PB and YTC, and DMH-GM outperforms DMH-SPD on BBT. Among all the baselines, SDH-SPD takes the first place, followed by SDH-GM and GDA. ITQ-GM and ITQ-SPD are inferior to the continuous baselines.

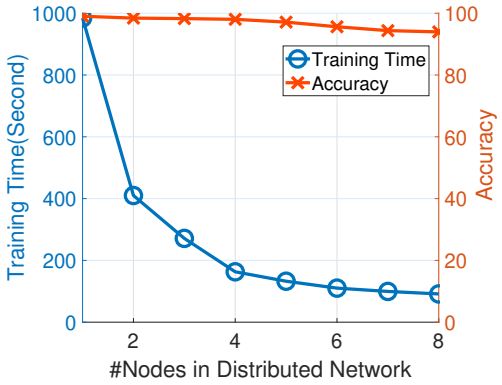
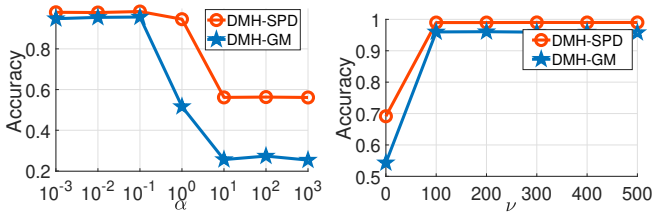


Figure 3: Training time (in seconds) and classification accuracies (%) of the proposed DMH on PB with respect to different numbers of nodes.

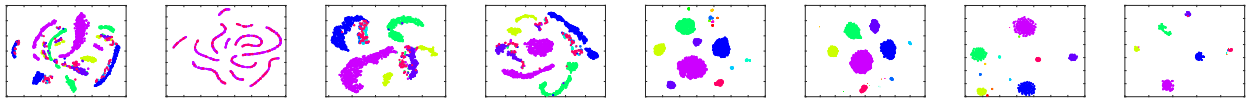


(a) α (b) υ

Figure 4: The performance of the proposed DMH with varying (a) α and (b) υ.

Efficiency Evaluation

Storage We compare storages of original feature, SPD manifold, kernel matrix, continuous feature, and hash code, where dimension of continuous feature and hash code is set to 32. Table 3 represents storage usages for these different representations. As can be clearly observed, storage usages using hash code are significantly lower than those using other representations. Particularly, the storage required using hash code is 64 times less than that required for contin-



(a) GDA (b) CDL (c) ITQ-SPD (d) ITQ-GM (e) SDH-SPD (f) SDH-GM (g) DMH-SPD (h) DMH-GM

Figure 5: The visualization of BBT by different methods, i.e., (a) GDA, (b) CDL, (c) ITQ-SPD, (d) ITQ-GM, (e) SDH-SPD, (f) SDH-GM, (g) DMH-SPD, (h) DMH-GM. Each point represents an image set, and each color indicates a class.

uous feature, and considerably less compared to other features. The learned hash code only requires less than 100 KB of storage for the three datasets. The empirical results demonstrate the superiority of the proposed method in terms of efficient storage usage.

Time Table 4 presents running time associated with distance calculation for several representative methods, where time of embedding generation is not included. As can be seen from Table 4, hashing methods clearly require less time than other methods. Specifically, hashing methods are nearly 8 times faster than GDA and CDL, 700 times faster than PML on BBT. The time reduction is more obvious compared to DCC and MSM, as the two methods require eigen decomposition to calculate distance, which is notably time-consuming. Hashing methods, including the proposed DMH, which calculate Hamming distance, are theoretically faster than conventional image set methods that calculate Euclidean distance. The above empirical results verify the theoretical superiority of the proposed method.

Further Analysis

Node Number Analysis This section analyzes the impact of number of nodes in the network on the performance of the proposed method. Figure 3 illustrates its running time of distance calculation and classification accuracies of the proposed method, varying from 1 to 8 nodes on PB with 32-bit hash code. As can be seen, training time sharply decreases while accuracy sightly drops as more nodes are utilized. Specifically, parallel training across 8 nodes is approximately 10 times faster than centralized training on just 1 node. This experiment confirms that parallel optimization scheme of the proposed method can improve training efficiency.

Trade-off Parameter Analysis The two trade-off parameters, i.e., α, υ are varied from $[10^{-3}, 10^{3}]$ and $[0, 500]$ respectively. Figure 4 reports accuracies and mAPs of the proposed DMH with respect to different values of the two parameters on BBT. We observe that the performance drops with large α. The performance improves with increase of υ. The good performance can be achieved when α is suggested to be less than 0.1 and υ is suggested to be larger than 100.

Discrete versus Continuous This section compares the performance of the proposed method and with its continuous variant that removes binary constraint. This variant is denoted by adding the suffix '-C'. The performances of the proposed method and its continuous variant in classification and retrieval tasks are reported in Table 5. From this table, we see that even with binary constraint, the proposed method still outperforms its variant in most cases, mainly due to its discrimination learning. The above empirical results suggest that does not degrade performance in image set classification and retrieval tasks.

Metric	Method	BBT	PB	YTC
Accuracy	DMH-SPD	98.72	98.03	76.48
	DMH-SPD-C	98.70	97.86	68.68
	DMH-GM	96.38	96.19	73.39
	DMH-GM-C	91.67	83.52	53.52
mAP	DMH-SPD	97.30	96.69	79.37
	DMH-SPD-C	94.99	97.50	72.00
	DMH-GM	99.14	89.02	63.97
	DMH-GM-C	94.49	81.09	58.41

Table 5: Performance of the proposed DMH using hash code and continuous representation.

Visualization

This section conducts qualitative empirical study and performs visualization. The widely-used t-SNE (der Maaten and Hinton 2008) is applied on learned representations of eight methods, and the visualization results on BBT are shown in Figure 5. From the above figure, we observe that compared to the baselines, the proposed DMH-SPD and DMH-GM generate much smaller clusters, and separate multiple clusters better. These qualitative results are consistent with previous quantitative results, intuitively reinforcing the effectiveness of the proposed DMH.

Conclusion

This paper presents the first attempt to study challenging hash code learning from distributed image sets, and proposes Distributed Manifold Hashing (DMH) for fast image set classification and retrieval. The basic idea is to model distributed image sets as a connected graph, and then to build a distributed model on this graph. The primary advantage of the proposed method lies in its capability of handling distributed image sets while maintaining competitive accuracy and high computational and storage efficiency. The empirical studies verify our theoretical findings.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 62176126, 62101268, U20B2065, 62272468, the Natural Science Foundation of Jiangsu Province, China under Grant No. BK20230095, BK20211539, and the Fundamental Research Funds for the Central Universities under Grant No. 30921011210.

References

Arsigny, V.; Fillard, P.; Pennec, X.; and Ayache, N. 2006. Geometric Means in a Novel Vector Space Structure on Symmetric Positive-Definite Matrices. *SIAM Journal on Matrix Analysis and Applications*, 29(1): 328–347.

Cevikalp, H.; and Triggs, B. 2010. Face recognition based on image sets. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2567–2573.

Chen, Z.; Xu, T.; Wu, X.; Wang, R.; Huang, Z.; and Kittler, J. 2023. Riemannian Local Mechanism for SPD Neural Networks. In *Proceedings of AAAI Conference on Artificial Intelligence*, 7104–7112.

der Maaten, L. V.; and Hinton, G. 2008. Visualizing data using t-SNE. *The Journal of Machine Learning Research*, 9(11).

Edelman, A.; Arias, T. A.; and Smith, S. T. 1998. The Geometry of Algorithms with Orthogonality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2): 303–353.

Gao, Z.; Wang, J.; Yu, G.; Yan, Z.; Domeniconi, C.; and Zhang, J. 2023. Long-Tail Cross Modal Hashing. In *Proceedings of AAAI Conference on Artificial Intelligence*, 7642–7650.

Gong, Y.; Lazebnik, S.; Gordo, A.; and Perronnin, F. 2013. 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.

Ham, J.; and Lee, D. D. 2008. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proceedings of International Conference on Machine Learning*, volume 307, 376–383.

Harandi, M. T.; Sanderson, C.; Shen, C.; and Lovell, B. C. 2013. Dictionary Learning and Sparse Coding on Grassmann Manifolds: An Extrinsic Solution. In *Proceedings of International Conference on Computer Vision*, 3120–3127.

Hu, Y.; Mian, A. S.; and Owens, R. A. 2012. Face Recognition Using Sparse Approximated Nearest Points between Image Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10): 1992–2004.

Huang, Z.; Wang, R.; Shan, S.; and Chen, X. 2015a. Projection Metric Learning on Grassmann Manifold with Application to Video based Face Recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 140–149.

Huang, Z.; Wang, R.; Shan, S.; Gool, L. V.; and Chen, X. 2018. 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.

Huang, Z.; Wang, R.; Shan, S.; Li, X.; and Chen, X. 2015b. Log-Euclidean Metric Learning on Symmetric Positive Definite Manifold with Application to Image Set Classification. In *Proceedings of International Conference on Machine Learning*, 720–729.

Kim, M.; Kumar, S.; Pavlovic, V.; and Rowley, H. A. 2008. Face tracking and recognition with visual constraints in real-world videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Kim, T.; Kittler, J.; and Cipolla, R. 2007. Discriminative Learning and Recognition of Image Set Classes Using Canonical Correlations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6): 1005–1018.

Kou, X.; Xu, C.; Yang, X.; and Deng, C. 2022. Attention-guided Contrastive Hashing for Long-tailed Image Retrieval. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1017–1023.

Li, Y.; Wang, R.; Shan, S.; and Chen, X. 2015. Hierarchical hybrid statistic based video binary code and its application to face retrieval in TV-series. In *International Conference on Automatic Face and Gesture Recognition*, 1–8.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 815–823.

Shen, F.; Shen, C.; Liu, W.; and Shen, H. T. 2015. Supervised Discrete Hashing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 37–45.

Sun, Y.; Peng, D.; Huang, H.; and Ren, Z. 2022. Feature and Semantic Views Consensus Hashing for Image Set Classification. In *Proceedings of ACM International Conference on Multimedia*, 2097–2105.

Sun, Y.; Wang, X.; Peng, D.; Ren, Z.; and Shen, X. 2023. Hierarchical Hashing Learning for Image Set Classification. *IEEE Transactions on Image Processing*, 32: 1732–1744.

Verbraeken, J.; Wolting, M.; Katzy, J.; Kloppenburg, J.; Verbelen, T.; and Rellermeyer, J. S. 2021. A Survey on Distributed Machine Learning. *ACM Computing Surveys*, 53(2): 30:1–30:33.

Wang, J.; Zhang, T.; Song, J.; Sebe, N.; and Shen, H. T. 2018a. A Survey on Learning to Hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4): 769–790.

Wang, R.; Guo, H.; Davis, L. S.; and Dai, Q. 2012. Covariance discriminative learning: A natural and efficient approach to image set classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2496–2503.

Wang, R.; Wu, X.; Chen, Z.; Xu, T.; and Kittler, J. 2022. Learning a discriminative SPD manifold neural network for image set classification. *Neural Networks*, 151: 94–110.

Wang, R.; Wu, X.; and Kittler, J. 2022. 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.

Wang, W.; Wang, R.; Huang, Z.; Shan, S.; and Chen, X. 2018b. Discriminant Analysis on Riemannian Manifold of Gaussian Distributions for Face Recognition With Image Sets. *IEEE Transactions on Image Processing*, 27(1): 151–163.

Wang, Y.; Xue, B.; Cheng, Q.; Chen, Y.; and Zhang, L. 2021. Deep Unified Cross-Modality Hashing by Pairwise Data Alignment. In *IJCAI*, 1129–1135.

Yamaguchi, O.; Fukui, K.; and Maeda, K. 1998. Face Recognition Using Temporal Image Sequence. In *International Conference on Automatic Face and Gesture Recognition*, 318–323.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2): 12:1–12:19.