An Interface between Grassmann manifolds and vector spaces

Lincon S. Souza[1] Naoya Sogi[1] Bernardo B. Gatto[2]
Takumi Kobayashi[3] Kazuhiro Fukui[1,2]
[1]Graduate School of Science and Technology, University of Tsukuba
[2]Center for Artificial Intelligence Research (C-AIR), University of Tsukuba
[3] National Institute of Advanced Industrial Science and Technology (AIST)
{lincons, sogi, bernardo}@cvlab.cs.tsukuba.ac.jp takumi.kobayashi@aist.go.jp
kfukui@cs.tsukuba.ac.jp

Abstract

In this paper, we propose a method to map data from a Grassmann manifold to a vector space while maximizing discrimination capability for classification. Subspaces are a practical and robust representation for image set recognition. However, as they exist on a Grassmann manifold, machine learning tools constructed on Euclidean geometry cannot be promptly utilized. Recently, methods to construct end-to-end learnable models for subspaces are starting to be explored, but they require multiple matrix decompositions and can be hard to compute and extend. Therefore we introduce a layer to map Grassmann manifold-valued data to vector space, in such a way that it can be seamlessly used as a layer along with other powerful tools defined on Euclidean space. The key idea of our method is to formulate the manifold logarithmic map (log) as a learnable model, where we seek to learn a tangency point that minimizes a loss function with respect to the data. The log effectively transforms a manifold point into a tangent vector. This log model can be learned with Riemannian stochastic gradient descent on the target manifold. We demonstrate the effectiveness of our proposed method on the applications of hand shape recognition, face identification and facial emotion recognition.

1. Introduction

In this paper, we propose a method to map data from a Grassmann manifold to a vector space while maximizing discrimination capability for classification. The Grassmann manifold represents the set of subspaces of a vector space, and as such, is a significant foundation for various types of machine learning tools using subspace representation. It has been well known as a practical and robust representation, especially for image set recognition. Despite its usefulness, most standard machine learning methods cannot be promptly utilized on the Grassmann manifold, since they are constructed on Euclidean space. Moreover, it is hard to directly link the Grassmann manifold to deep neural network architectures. To fill this serious gap and exploit both the compact representation of Grassmann manifold and the handiness of Euclidean space, we propose a method named Grassmann log model to connect those two representations.

The key idea of our method is to formulate the manifold logarithmic map (log) as an end-to-end learnable model working as an interface between the Grassmann manifold and Euclidean space. It can be seamlessly followed by discriminative tools defined on Euclidean space, to provide a discriminative representation for subspaces so that Euclidean methods can perform classification well. Also, the proposed model can be learned in an end-to-end manner, being embedded as a single module in larger network systems.

The motivation of proposing this method is three-fold: 1) a subspace is a robust representation and has become a central research topic in computer vision, being applied to numerous problems such as image set recognition [6, 36, 18, 33, 10, 38], fine-grained classification [42] and action recognition [32, 35, 25]. 2) The image set recognition, where the goal is to model a set of input images and classify it, has been shown to provide significant recognition stability, but the set representation in the deep learning framework remains largely unexplored. On the other hand, 3) there is an abundance of well-established network layers and other end-to-end processing, which work in Euclidean space, e.g. fully-connected, batch normalization dropout layers, activation functions. We would like to connect the useful subspace representation to these Euclidean tools.

In the Grassmann manifold, one of the most popular methods that works by mapping manifold data to a vector space is the Grassmann discriminant analysis (GDA) [11]. GDA along with its extensions, e.g. Grassmannian Graph-Embedding GDA (GGDA) [12] and enhanced GDA (eGDA) [34] employ a Grassmann kernel to map sub-

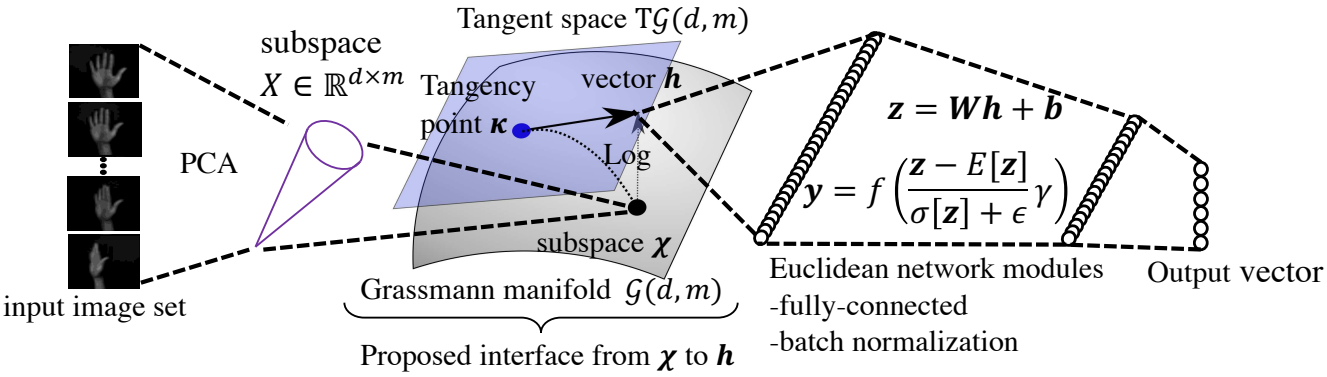


Figure 1: Conceptual diagram of the proposed Grassmann log model. A subspace $\boldsymbol{\chi}$ is computed by PCA, represented by an orthogonal basis matrix. Our proposed interface is to log map $\boldsymbol{\chi}$ into a tangent vector $\boldsymbol{h}$; then Euclidean network modules are applied. The first equation indicates a fully-connected layer, where $\boldsymbol{W}$ and $\boldsymbol{b}$ denote the weights and bias, respectively. The second equation indicates a batch normalization where E and σ denote expectation and variance, ϵ, γ are batch normalization hyperparameters, and f is a non-linear activation function.

spaces onto vectors in a reproducing kernel Hilbert space (RKHS), where then kernel discriminant analysis is performed. However, these methods are limited as they use vectors in RKHS defined by the kernel function, without learning a manifold-aware discriminant mechanism. Additionally, the dimension of the Grassmann kernel is directly dependent on the number of samples in the dictionary, which could possibly lead to an insufficiently small space to represent data, or lead to a curse of dimensionality.

Considering end-to-end methods, one could develop a simple method of concatenating a subspace basis vectors and then using Euclidean layers such as a fully-connected layer directly. However, such an approach cannot learn stably, since this maneuver would break the inherent subspace structure, impairing its capabilities. Additionally, a cross-entropy loss would not converge in this setting. Therefore, an interface is necessary to learn the Euclidean layers properly.

Our proposed Grassmann log model then contains two main stages, which can be seen in Fig. 1: 1) a mapping from manifold to vector space, and then 2) Euclidean network modules such as fully connected layers. More concretely, given a manifold data point $\boldsymbol{\chi}$ as input data, the log maps a manifold data point into a tangent vector $\boldsymbol{h}$ in a tangent space parameterized by a tangency point $\boldsymbol{\kappa}$. Then $\boldsymbol{h}$ is transformed through Euclidean layers and the cross-entropy loss function is applied. To obtain a discriminant interface between manifold and vector data, we learn both the tangent space representation and the discriminant Euclidean layers in an end-to-end manner. The log model is a general framework that can be learned with Riemannian stochastic gradient descent [4] on any Riemannian manifold with a defined closed-form log map. In the Grassmannian case, the parameter $\boldsymbol{\kappa}$ is learned as a point constrained to the Grassmann manifold, and the Euclidean layers are learned in conventional Euclidean space. In the following, we refer to tangency point as "anchor point" with emphasis on this point.

Most classification problems have a complex distribution with a wide variance where a single tangent vector may yield a suboptimal representation. Therefore, we extend the log model to learn more than one anchor point, obtaining a set of tangent vectors that are concatenated to output a single feature vector. From a geometry viewpoint, this idea can be interpreted as a wider atlas of tangent spaces, covering a broader neighborhood of the manifold and diminishing distortion. From the perspective of computer vision, this idea is similar to that of popular image descriptors that combine multiple residual vectors, such as Fisher vectors [30, 29], VLAD [3], and super-vector coding [43], and the log model can be seen as a kind of extension of these ideas to represent an image set rather than just a single image.

Through experiments, we demonstrate that learning a tangent space is important for finding a discriminative map, by making a comprehensive comparison of a learned log model against a fixed tangent space at the manifold data's Karcher mean. We demonstrate the flexibility and scalability of our Grassmann log model as an interface between deep network layers involving subspace data, by also evaluating the log model as a middle stage within larger networks containing convolution layers, and Resnet blocks along with PCA. We show the effectiveness of both in artificial subspace data and in real data for the applications of face identification, facial expression and hand shape recognition.

2. Theoretical Background

In this section, we review the operations we can perform on the Grassmann manifold. The following notation is used: lowercase plain letters for scalars, lowercase/uppercase plain letters for functions, lowercase bold for vectors, uppercase

bold for matrices, lowercase bold Greek letters for Grassmann manifold points and calligraphic letters for manifolds.

2.1. Grassmann Manifold

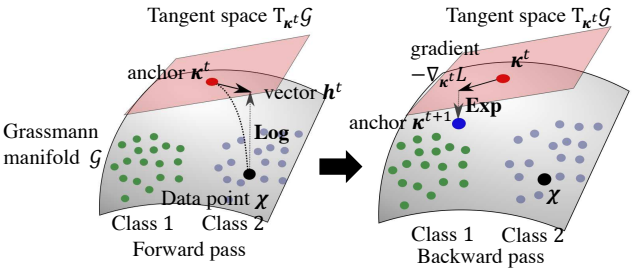


Figure 2: Conceptual diagram of the log model learning a tangent space for a binary class problem. The log is used to map point $\boldsymbol{\chi}$ to vector $\boldsymbol{h}^t$ in the tangent space (in red). Then, a loss function can be applied and the gradient with respect to the anchor point (tangency point) $\boldsymbol{\kappa}^t$ can be used to move towards a more optimal position $\boldsymbol{\kappa}^{t+1}$ which defines a new tangent space (in blue).

The *Grassmann manifold* $\mathcal{G}(d, m)$ is defined as the set of m-dimensional linear subspaces of $\mathbb{R}^d$. It is an $m(d-m)$-dimensional compact manifold and can be written as a quotient space of orthogonal groups $\mathcal{G}(d, m) = \mathcal{O}(d)/\mathcal{O}(m) \times \mathcal{O}(d-m)$, where $\mathcal{O}(m)$ is the group of $m \times m$ orthonormal matrices.

A point $\boldsymbol{\chi}$ in $\mathcal{G}(d, m)$, i.e., a m-dimensional subspace of $\mathbb{R}^d$, can be represented extrinsically by an orthogonal basis matrix $\boldsymbol{X} \in \mathcal{S}(d, m)$, where the columns of the matrix form a basis such that $\boldsymbol{\chi}$ is the set of all their linear combinations. $\mathcal{S}(d, m)$ denotes the manifold of tall orthogonal matrices, called compact Stiefel manifold.

2.2. Tangent spaces

A tangent space $T_{\boldsymbol{\kappa}}\mathcal{G}$ at $\boldsymbol{\kappa} \in \mathcal{G}(d, m)$ can be seen intuitively as a subspace of $\mathbb{R}^{d \times m}$, since $T_{\boldsymbol{\kappa}}\mathcal{G}$ can be derived as a horizontal space. Given a point $\boldsymbol{K} \in \mathcal{S}(d, m)$, such that $\text{span}(\boldsymbol{K}) = \boldsymbol{\kappa}$, a horizontal space $\mathcal{H}$ is a subspace of the tangent space $T_{\boldsymbol{K}}\mathcal{S}$ (a copy of $\mathbb{R}^{d \times m}$) which contains the directions of infinitesimal variation of $\boldsymbol{K}$ that modify its span. In this work we always work with orthogonal bases for subspaces, so it is intuitive to imagine a tangent space from the derivative of the orthogonal property $\boldsymbol{K}^\top \boldsymbol{K} = \boldsymbol{I}$, given by $\boldsymbol{K}^\top \dot{\boldsymbol{K}} + \dot{\boldsymbol{K}}^\top \boldsymbol{K} = \boldsymbol{0}$. Any tangent vector $\dot{\boldsymbol{K}}$ to a subspace $\boldsymbol{\kappa}$ must satisfy this constraint for some $\boldsymbol{K}$ that spans $\boldsymbol{\kappa}$.

2.3. Exponential and Logarithmic maps

The exponential map $\text{Exp} : \mathcal{G} \times T\mathcal{G} \times \mathbb{R} \to \mathcal{G}$ can be used to calculate an specific point on a geodesic $\gamma(t)$, given a point $\gamma(0)$, a direction $\dot{\gamma}(t)$ and a length t. It is denoted by $\gamma(t) = \text{Exp}_{\boldsymbol{\kappa}} \boldsymbol{H}$, meaning the point $\gamma(t)$ in the geodesic emanating from $\boldsymbol{\kappa} = \gamma(0)$ in the direction of $\boldsymbol{H} = t\frac{\dot{\gamma}(0)}{\|\dot{\gamma}(0)\|} \in T_\kappa\mathcal{G}$.

The inverse of the exponential map is the logarithmic map (or log map) $\text{Log} : \mathcal{G} \times \mathcal{G} \to T\mathcal{G}$, denoted by $\boldsymbol{H} = \text{Log}_{\boldsymbol{\kappa}} \boldsymbol{\chi}$. Given two points on the manifold $\boldsymbol{\chi}$ and $\boldsymbol{\kappa}$, one wants to find the tangent vector $\boldsymbol{H}$ at $\boldsymbol{\kappa}$ pointing towards $\boldsymbol{\chi}$. Note that $\text{Log}_{\boldsymbol{\chi}} \boldsymbol{\kappa} \neq \text{Log}_{\boldsymbol{\kappa}} \boldsymbol{\chi}$. In other words, the log outputs a tangent vector to the shortest path curve between $\boldsymbol{\kappa}$ and $\boldsymbol{\chi}$.

3. Proposed Grassmann Log model

In this section, we describe the algorithm to the proposed Grassmann Log model. First, we define the Log layer more generally and then introduce its numerical algorithm. Next, we extend it to work with multiple tangent spaces.

3.1. Basic Idea

Our learning problem is defined as follows: given training subspaces $\{\boldsymbol{\chi}_i\}_{i=1}^N \in \mathcal{G}$ paired with respective labels c_i, we want to learn a model that maps them to vectors $\boldsymbol{h}_i$ in Euclidean space, so that the class distributions are separable. In the following discussion, we consider a minibatch $\{\boldsymbol{\chi}_i\}_{i=1}^n$, as is common in neural networks, but the whole process can be repeated iteratively to use all N training subspaces. When convenient, we omit the i index, and explain the process for a single instance i.e. $\boldsymbol{\chi}$.

Our key idea is to cast the log map $\text{Log}_{\boldsymbol{\kappa}} \boldsymbol{\chi}$ as a learnable model by 1) defining the anchor point $\boldsymbol{\kappa}$ as a parameter and $\boldsymbol{\chi}$ as input, and 2) seeking an anchor point $\boldsymbol{\kappa}$ such that a tangent vector $\text{Log}_{\boldsymbol{\kappa}} \boldsymbol{\chi}$ of the manifold data point $\boldsymbol{\chi}$ can be as discriminant as possible in the tangent space.

3.2. Learning the Log layer

The learning process of the log module is given as follows. Figure 2 shows the conceptual diagram of learning the tangent space with the anchor $\boldsymbol{\kappa}^t$ on a binary classification, where t is indexing the update iteration. The whole learning process consists of forward and backward passes. At the iteration $t = 0$, we initialize our anchor parameter as a random manifold point.

3.2.1 Forward pass

The forward pass is processed as follows: we map a point $\boldsymbol{\chi}$ into a vector $\boldsymbol{h}^t$ by:

$$\boldsymbol{h}^t = \text{vec}(\text{Log}_{\boldsymbol{\kappa}^t} \boldsymbol{\chi}). \tag{1}$$

As the Grassmann manifold is a matrix manifold, we utilize the vectorization of matrices vec to turn the tangent vectors from matrix to simple vectors, such that $\boldsymbol{h}^t \in \mathbb{R}^{dm}$. This does not affect the distance structure. The equation above corresponds to the log map projecting a point $\boldsymbol{\chi}$ on the manifold to vector $\boldsymbol{h}^t$ in the red tangent space as shown

in the first figure in Fig.2. After the proposed log layer, Euclidean modules and a discriminative loss function, here abstracted as a loss function $L(\boldsymbol{h}_i, c_i) : \mathbb{R}^{dm} \rightarrow \mathbb{R}$ can be applied to the tangent vectors.

3.2.2 Backward pass

Then, the backward pass is processed as follows: the Euclidean gradient $\nabla_{\boldsymbol{h}^t} L$ with respect to the tangent vectors is computed, and then the log layer gradient is computed through the backpropagation algorithm. We write the chain gradient including the log as:

$$\nabla_{\boldsymbol{\kappa}^t} L(\boldsymbol{\chi}_i, c_i, \boldsymbol{\kappa}^t) = \nabla_{\boldsymbol{h}^t} L \frac{d}{d\boldsymbol{\kappa}^t}(\mathrm{Log}_{\boldsymbol{\kappa}^t} \boldsymbol{\chi}_i), \tag{2}$$

where $\frac{d}{d\boldsymbol{\kappa}^t}(\mathrm{Log}_{\boldsymbol{\kappa}^t} \boldsymbol{\chi}_i)$ represents the derivative of the log map.

Given the gradient $\nabla_{\boldsymbol{\kappa}^t} L$, we perform the update of the anchor $\boldsymbol{\kappa}^t$ by Riemannian stochastic gradient descent (RSGD) [5, 4]. This manifold aware update enforces the updated point $\boldsymbol{\kappa}^{t+1}$ to be a member of the Grassmann manifold, i.e., avoiding the parameter to leave the manifold. The RSGD update consists of two steps: 1) transforming the Euclidean gradient $\nabla_{\boldsymbol{\kappa}^t} L$ to a Riemannian gradient $\mathrm{grad}_{\boldsymbol{\kappa}^t} L$, that is, the closest vector to $\nabla_{\boldsymbol{\kappa}^t} L$ that is also tangent to the manifold at $\boldsymbol{\kappa}^{\boldsymbol{t}}$. Then 2) updating the anchor point by:

$$\boldsymbol{\kappa}^{t+1} = \mathrm{Exp}_{\boldsymbol{\kappa}^t} -\lambda \nabla_{\boldsymbol{\kappa}^t} L. \tag{3}$$

Here, λ is a learning rate.

The above update equation is illustrated in the second part of Figure 2. The exponential map can be seen as a line walk towards the opposite direction to the Riemannian gradient (direction of descent), landing on a more optimal point $\boldsymbol{\kappa}^{t+1}$ (in blue). This new anchor point defines a new tangent space, which should be more optimal in the sense that the corresponding log map yields tangent vectors with higher class separability as shown in the last figure. This iterative process is repeated until either a maximum number of iterations is achieved, or the gradient becomes too small, that is, the separability cannot be improved much further.

3.3. Numerical algorithm

In the previous section, we have established the framework in general while abstracting the computations. In this section, we describe the equations used in this framework to compute the exp and log maps in matrix form. Then, we and derive the log backward updates.

To compute the Grassmann exponential map, we utilize the following extrinsic function derived by [1], written in terms of orthonormal matrix representation. Recall our anchor parameter $\boldsymbol{\kappa}$ is a subspace, so given its basis matrix $\boldsymbol{K} \in \mathbb{R}^{d \times m}$, and given a tangent vector $\boldsymbol{H} \in \mathbb{R}^{d \times m}$:

$$\mathrm{Exp}_{\boldsymbol{K}} \lambda \boldsymbol{H} = \mathrm{orth}(\boldsymbol{K}\boldsymbol{Q}(\cos \boldsymbol{\Sigma}\lambda)\boldsymbol{Q}^\top + \boldsymbol{J}(\sin \boldsymbol{\Sigma}\lambda)\boldsymbol{Q}^\top), \tag{4}$$

where $\boldsymbol{J}\boldsymbol{\Sigma}\boldsymbol{Q}^\top = \boldsymbol{H}$ is the compact singular value decomposition (SVD) of the tangent vector $\boldsymbol{H}$. Note that $\boldsymbol{H}$ is written in upper case as it is a matrix; yet it is still is a vector in the sense that is a member of a tangent vector space. Here, $\boldsymbol{J}, \boldsymbol{K}, \boldsymbol{Q}$ and $\mathrm{Exp}_{\boldsymbol{K}} \lambda \boldsymbol{H}$ are orthogonal matrices, and $\boldsymbol{\Sigma}$ is a diagonal matrix. λ is the geodesic parameter, and can be seen as a step value to control the magnitude of the movement towards the direction $\boldsymbol{H}$.

As for the log map, in this work, given two basis matrices $\boldsymbol{X}$ and $\boldsymbol{K}$ for input subspace $\boldsymbol{\chi}$ and anchor $\boldsymbol{\kappa}$, we utilize the following three equations to calculate the log map [2]:

$$\boldsymbol{B} = (\boldsymbol{K}^\top \boldsymbol{X})^{-1}(\boldsymbol{K}^\top - \boldsymbol{K}^\top \boldsymbol{X}\boldsymbol{X}^\top), \tag{5}$$

$$\boldsymbol{W}\boldsymbol{\Theta}\boldsymbol{Z}^\top = \boldsymbol{B}^\top, \tag{6}$$

$$\mathrm{Log}_{\boldsymbol{K}} \boldsymbol{X} = \boldsymbol{H} = \boldsymbol{W}^* \arctan(\boldsymbol{\Theta}^*) \boldsymbol{Z}^{*\top}, \tag{7}$$

where $\boldsymbol{W}^*, \boldsymbol{\Theta}^*, \boldsymbol{Z}^*$ represent the matrices with the first m columns of $\boldsymbol{W}, \boldsymbol{\Theta}$ and $\boldsymbol{Z}^*$ respectively.

To perform learning by RSGD, we derived the expressions of the derivative for the log on the Grassmann manifold, using conventional techniques to operate differential forms [27] and based on the derivative of SVD [37]. For the detailed procedures, see the supplementary material. We omit the iteration t for simplicity.

Given the gradient of all next layers after the log layer, up to the loss $\dot{\boldsymbol{H}} = \nabla_{\boldsymbol{H}} L$, we compute the gradients with respect to the anchor $\dot{\boldsymbol{K}} = \nabla_{\boldsymbol{\kappa}} L$ and input subspace $\dot{\boldsymbol{X}} = \nabla_{\boldsymbol{\chi}} L$. $\dot{\boldsymbol{K}}$ is used to update the anchor according to equation 3, and $\dot{\boldsymbol{X}}$ is used if there are layers previous to the log layer, to compute their respective backward steps.

We provide the update formulation for the Grassmann log equations 5, 6 and 7 in Figure 3. Since the log is determined from the composition of three functions, the chain rule can be automatically used to compute the final gradients.

In Fig. 3, $\boldsymbol{\Omega}$ is defined as a diagonal matrix where the diagonal elements are $\boldsymbol{\Omega}_i = 1/(1 + \boldsymbol{\Theta}_i^*)$. $\circ$ represents the Hadamard product, and $\boldsymbol{I}$ represents the identity matrix. $\boldsymbol{F}$ is a matrix of the form:

$$\boldsymbol{F}_{ij} = \begin{cases} 1/(\arctan^2(\boldsymbol{\Theta}_j) - \arctan^2(\boldsymbol{\Theta}_i)), & i \neq j \\ 0, & i = j. \end{cases} \tag{8}$$

For the gradient of equation 7, the derivatives $\dot{\boldsymbol{W}}^*$, $\dot{\boldsymbol{\Theta}}^*$ and $\dot{\boldsymbol{Z}}^*$ are m-leftmost matrices, so to continue back to the square matrix gradients $\dot{\boldsymbol{W}}$, $\dot{\boldsymbol{\Theta}}$ and $\dot{\boldsymbol{Z}}$, we can fill in columns of zeros (no gradient update in these variables) until the matrices become square. In the gradient of equation 6, $\boldsymbol{F}$ is similar to equation 8, but the non-diagonals are instead defined as $1/(\boldsymbol{\Theta}_j^2 - \boldsymbol{\Theta}_i^2)$. The most important part is in the gradient of 5 (upper box with equations), we obtain two update rules, one to backpropagate $\boldsymbol{X}$ in case a gradient-based pre-processing needs it, and one to update the anchor point $\boldsymbol{K}$.

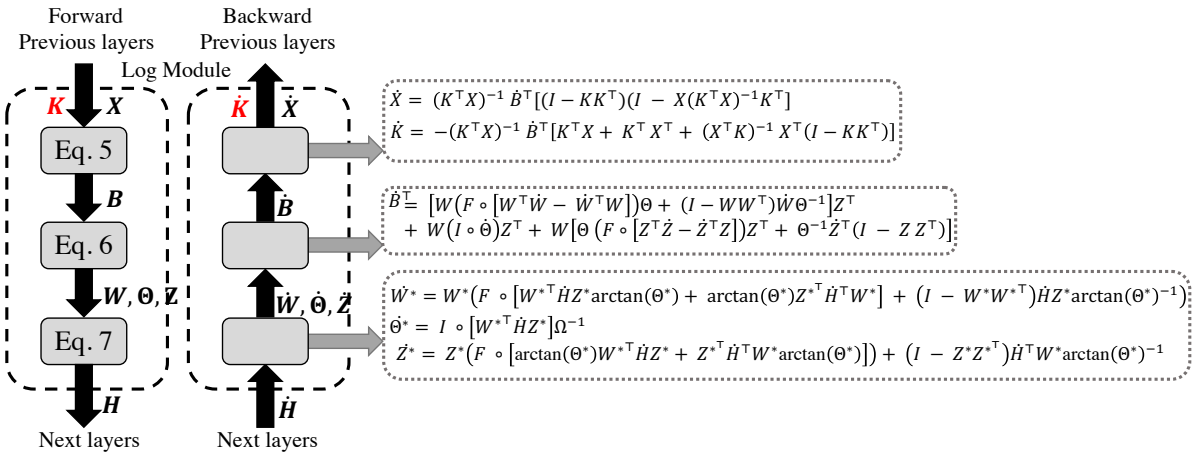


Figure 3: Update equations of the Grassmann log module backward phase. The objective is to compute the gradient $\dot{\boldsymbol{K}}$ of the anchor parameter $\boldsymbol{K}$ (in red) and the gradient $\dot{\boldsymbol{X}}$ of input subspace $\boldsymbol{X}$. $\dot{\boldsymbol{K}}$ is used to update the anchor according to the RSGD update, and $\dot{\boldsymbol{X}}$ is used if there are layers previous to the log layer, to compute their respective backward steps.

3.4. Extension to multiple tangent spaces

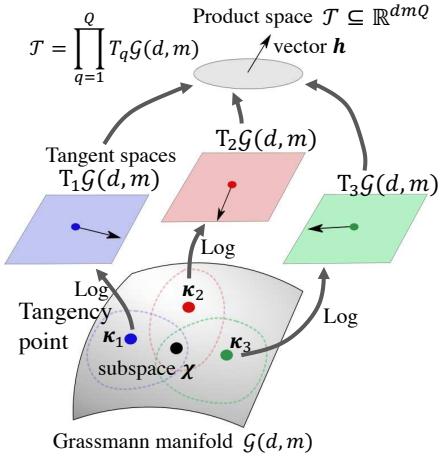


Figure 4: Diagram of the log module consisting of multiple tangent spaces.

We further extend the proposed log layer to learn multiple tangent spaces at the same time, by defining the layer parameters as a set of anchors $\{\boldsymbol{\kappa}_q\}_{q=1}^{Q}$. We map the point $\boldsymbol{\chi}$ into a vector $\boldsymbol{h}$ in a space $\mathcal{T}$ as follows:

$$\boldsymbol{h}_q = \mathrm{vec}(\mathrm{Log}_{\boldsymbol{\kappa}_q} \boldsymbol{\chi}) \tag{9}$$

$$\boldsymbol{h} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_q, \ldots, \boldsymbol{h}_Q]. \tag{10}$$

Here, the brackets denote the concatenation of vectors such that the output feature vector $\boldsymbol{h}$ is in a $\mathcal{T} \subseteq \mathbb{R}^{dmQ}$. Optimizing for a discriminative space $\mathcal{T}$ can be seen as a search in a product space of the Grassmann tangent bundle $T\mathcal{G}(d,m)$, i.e. $\mathcal{T} = \prod_{q=1}^{Q} T_q\mathcal{G}(d,m)$. An example diagram for the case of $Q = 3$ is shown in Fig. 4.

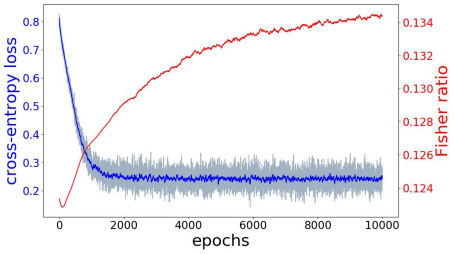
The basic intuition about introducing multiple tangent spaces is to cover well a larger neighborhood of the manifold. As it maps a non-flat manifold onto a Euclidean space, there cannot be a single map that is perfectly distance preserving between any two points of the manifold. In general, the log map is a good approximation of the manifold in a local neighborhood of $\boldsymbol{\kappa}_q$ and its representation power decreases to points too far from it. However, we desire a good representation for the data distribution rather than for all points in the manifold. The core idea here is that each anchor point $\boldsymbol{\kappa}_q$ is treated as an independent learnable parameter. By having several $\boldsymbol{\kappa}_q$ and learning them from a random initialization without assumptions, we may find the tangent spaces that produce discriminant vectors for the given data samples. In the case $Q = 1$, the equations above reduce to conventionally using the log once.

As described in Sec. 1, this extension of the log model is similar to that of popular image descriptors such as Fisher vectors [30, 29], VLAD [3], and super-vector coding [43]. A Fisher vector consists of the log-likelihood gradients of data descriptors with respect to a Gaussian mixture. The log model can be seen as a kind of extension of this idea of using multiple tangent vectors, but to represent an image set rather than just a single image.

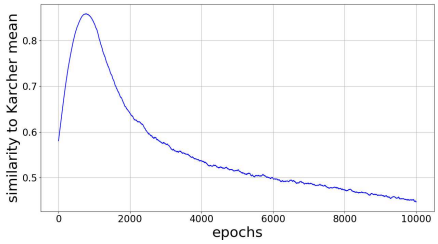
4. Experiments

4.1. Experiments on Artificial Data

We have trained a Grassmann log model using 3D artificial data to visualize its mapped data and verify its effectiveness in a very simple case, while obtaining some intuition about its mechanism. The artificial data contains two classes of points on the $\mathcal{G}(3,1)$, that is, the lines on 3-space crossing the origin. The data is represented by 3D vectors, and



(a) loss and Fisher ratio of the tangent vectors.



(b) anchor point/ Karcher mean similarity.

Figure 5: Plots of a log model trained for 10 thousand epochs in artificial data: (a) the cross-entropy loss (in blue), and the Fisher ratio between the tangent vectors, before they have been projected on a discriminant space (in red). (b) similarity between the anchor point and the Karcher mean at each epoch. Note that the Karcher mean is fixed while the anchor point moved.

each class is generated by a tangent Gaussian distribution as developed in [39]. The model we utilised is composed of the log with 1 anchor, and two fully-connected layers, one 3×3 and another 3×2.

We have trained this model for 1000 epochs and plotted the anchor's behavior.Figure 5a shows the cross-entropy loss of the network and the Fisher ratio of the tangent vectors of the log map, for each epoch during training.

Here, one can observe that while the loss is minimized, the Fisher ratio of the tangent vectors raises, suggesting the advantageous effects of using a learnable anchor point. Therefore, the choice of a tangent space appears to contribute to the data separability. Moreover, Figure 5b shows the similarity between the data's Karcher mean and the anchor point for every epoch.

4.2. Experiments on Hand Shape Recognition

We conducted an experiment with the Tsukuba hand shape dataset. This dataset contains 30 hand classes $\times$ 100 subjects, each of which contains 210 hand shape images, consisting of 30 frames $\times$ 7 different viewpoints. For each subject, we randomly created 6 sets with 5 frames from each viewpoint, so that each subject has 6 image sets of 35 images. In summary, there are a total of 18000 image sets in this dataset, each set containing image information from 7

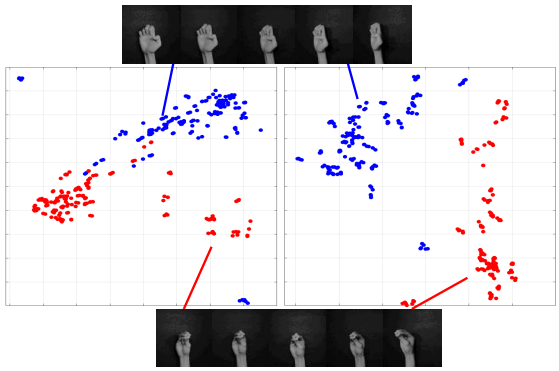


Figure 6: tSNE visualizations of 2 classes of hand shapes. The left plot denotes the tangent vectors at the Karcher mean, while the right plot shows the tangent vectors at the log model learned tangent space. It shows a representation of the vectors as 2D points based on their euclidean distances. It can be seen that the tangent vectors at the learned tangent space provide a more discriminative representation.

camera viewpoints. In the experiments, all the images were resized to 24×24 pixels. We compute the subspaces by PCA and extract 6 components, so for the Log model the input subspaces are on the $\mathcal{G}(576, 6)$.

The Grassmann log model used in this section is formed by of a log map followed by 4 fully-connected (F.C) layers, with batch normalization, dropout and number of anchor points is 6.

4.2.1 Effectiveness of learning tangent spaces

First, to verify the effectiveness of learning a tangent space, we trained two log models in the classification problem of 30 classes of hand shapes: one was trained normally, with random initialization and iterative updating of the tangent space according to the gradient. The other model was trained while freezing the tangent space always at the Grassmann Karcher mean of all training subspaces, a version we refer to as Karcher log model.

After training, we have extracted the log map tangent vectors, and measured their Fisher ratio (FR), i.e. the ratio between the average inter-class and intra-class Euclidean distances of the tangent vectors. The FR of the log model was 0.0171, while the FR at the Karcher was 0.0159. The FR is higher when we allow the model to learn the anchor point, which indicates the choice of tangent space works towards increasing separation capability of the subspace data. To further exemplify this property, we have selected two classes at random and plotted their distance structure using tSNE [26], which can be seen in Fig. 6. The plot shows a representation where each feature vector is shown as a point. The left plot denotes the tangent vectors at the Karcher mean, while the right plot shows the tangent vectors

at the log model learned tangent space. The log model mapping seems to have made the samples more discriminant simply by finding suitable tangent spaces to project the data.

	Accuracy (%)
Karcher log model	70.65
Log model	81.90
Conv+log model	91.90
Resnet18+log model	99.40

Table 1: Results on the Tsukuba hand shape dataset.

As a second experiment, we evaluated the performance of the proposed log model in the classification problem of 30 types of hand shapes and compared it to the Karcher log model. We used the image sets of 70 subjects as training sets, holding a subset of 15 subjects for validation. The remaining 30 subjects were used as testing sets.

Table 1 shows the results. The proposed Log model equipped with several anchor points provided to be efficient in modeling the complexity of the hand shapes, while the Karcher log model achieved an inferior performance.

4.2.2 Scalability of the Log model

We evaluated the scalability in performance of the proposed log model in the same hand shape classification problem, extending the log model with two variants: the Conv+log and Resnet18+log models.

Conv+log model refers to an architecture with 3 convolutions layers with batch normalization and pooling, where the convolutional filter size is 3, the number of filters of the first layer is 8, while the second and third are set to 2. Each layer has a zero-padding of 1 pixel and a pooling mask size of 2. After the convolutions, we have PCA and a log map, where the number of anchors of the log map is 2. After that, we utilize 4 F.C blocks.The Conv+log architecture was learned end-to-end including PCA, and entirely from scratch, using only randomly initialized weights. The method named

Results on the CMU MoBo		Results on the AFEW	
Method	Accuracy (%)	Method	Accuracy (%)
DCC[20]	88.89 ± 2.45	STM-ExpLet[22]	31.73
MMD[41]	92.50 ± 2.87	RSR-SPDML[13]	30.12
CHISD[7]	96.52 ± 1.18	DCC[20]	25.78
MMDML[24]	97.80 ± 1.00	GDA[11]	29.11
ADNT[14]	97.92 ± 0.73	GGDA[12]	29.45
PLRC[9]	93.74 ± 4.30	PML[17]	28.98
DRM[31]	98.33 ± 1.27	DeepO2P[19]	28.54
Resnet vote	98.61 ± 1.52	SPDNet[16]	34.23
Log model	98.19 ± 1.31	GrNet-1[18]	32.08
——	——	GrNet-2[18]	34.23
——	——	Log model	32.61

Table 2: Results on the CMU MoBo and AFEW datasets.

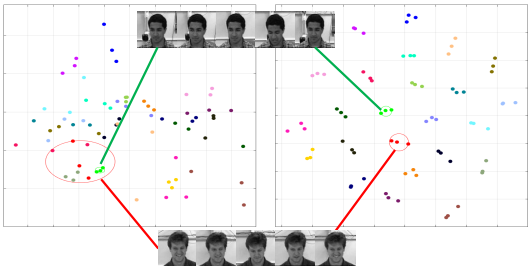


Figure 7: tSNE visualizations of 24 individuals of face image sets of the CMU Mobo dataset. The left plot denotes face image sets processed as subspaces. The plot shows a representation of the subspaces relative distances based on their similarities. The right plot shows mapped tangent vectors (output of the log layer). The colors represent each individual (each class). Visually, the tangent vectors provide a more discriminative representation.

Resnet18+log model uses a Resnet18 network pre-trained on ImageNet and fine-tuned on the hand shape data, similar to the conventional methods. We replaced the final F.C. layer of Resnet by a log model with PCA, log, and 4 F.C blocks, and trained this architecture while freezing the weights of the fine-tuned Resnet.

The results for these architectures can be seen in Table 1. When the learning framework is equipped with a convolutional layer, the log model achieves considerable accuracy improvement, strengthening the concept that the proposed interface is flexible enough to be incorporated in general neural network architectures. Following this pattern, the resnet18 architecture also benefits from the proposed interface.

4.3. Experiment on Face Identification

We conducted an experiment on the CMU Mobo dataset, consisting of footage videos of 25 people walking on a treadmill. This dataset was originally utilized for research on human gait analysis, but recently it has been used to compare the performance of image set based face classification methods [31, 14, 7].

We first detected face region from each video frame by the Viola and Jones detection algorithm [40]. A set of face images extracted from one video was considered as an image set. This dataset has four walking patterns (videos) of each person, except for one person. We evaluated the classification performance 10 times with the videos of 24 people with all walking patterns. For the evaluation, one video randomly selected from each person was used for training, while the remaining three videos were used for testing.

In this experiment, the subspaces are generated from CNN features vectors. As the feature extractor, we used the ResNet-50 [15], which was fine-tuned to classify each face image of training data. For the fine-tuning, we added

two fully connected (FC) layers after the last global average pooling layer in the network. The first FC layer outputs a 1024 dimension vector through the ReLU [28] function, and the second layer outputs a 24 (the number of classes) dimension vector through softmax function. We used the cross-entropy loss and Adam optimizer [21]. Hyperparameters of the optimizer were used as suggested by the original paper. We repeated the training to 100 epochs. Then, we extracted 1024 feature vectors by the first FC layer and for each set we computed a subspace with PCA by extracting 10 components. Therefore, a subspace input to the log module is on the $\mathcal{G}(1024, 10)$. We trained a Grassmann log model consisting of a log map followed by 1 fully-connected (F.C) layers, with number of anchor points 2.

First, we visualize the subspaces obtained through PCA in Fig. 7 by using tSNE. The left plot shows a representation where each subspace is a point, and subspace similarities correspond to Euclidean distances on the plane. We also visualized the log tangent vectors of our trained model in the right plot. Since there are 2 anchor points, the dimension of the Euclidean feature space (product of the tangent spaces) is $2 \times 1024 \times 10$. Each point corresponds to one tangent vector, and their distances correspond to Euclidean distances on the tangent spaces. The output vectors of the log model exhibits compact clusters, while the ones on the Grassmann manifold are visually more dispersed. Additionally, the log model seems to discriminate data from different classes more appropriately, suggesting that the tangent spaces learned by the model provide a more suitable space for classification.

We have also evaluated the log model against various manifold methods from previous works. Resnet vote [15] is a baseline consisting of a Resnet50 fine-tuned to this dataset, where each image of a set is classified independently and a majority voting strategy is used to select a single class prediction. Table 2 shows the results. The proposed model overperforms the classic methods, and achieves a result on par with various deep methods.

4.4. Experiments on Emotion Recognition

We utilize the Acted Facial Expression in Wild (AFEW) [8] dataset. The dataset contains $1,345$ sequences of facial expressions acted by 330 actors in close to real world setting. We follow the experiment protocol established by [22, 16] to present the results on the validation set. The training videos are split into $1,747$ small subvideos augmenting the numbers of sets. For the evaluation, each facial frame is normalised to an image of size 20×20. For representation, following various works [23, 22, 18], we express the facial expression sequences with a set of linear subspaces of dimension 10, which exist on a Grassmann manifold $\mathcal{G}(400, 10)$. The Grassmann log model used in this section was composed of a log map followed by 3 fully-connected (F.C) layers, with batch normalization and dropout, in addition to a final F.C. with softmax, with 4 anchor points.

We compare the proposed log model with a number of methods for classification of manifold-valued data. Grassmann net (GrNet) [18], that proposes a block of manifold layers for subspace data, is denoted by GrNet-1 for the architecture with 1 block and GrNet-2 for the one with 2 blocks.

The results can be seen in Table 2. The proposed method achieved competitive results, even though it uses simple Euclidean operations such as fully-connected layers and a cross-entropy loss. First, it overperforms popular methods such as GDA and GGDA, that have a similar purpose of mapping Grassmann manifold data into a vector representation. This may be likely attributed to the fact that the log model learns both the representation and the Euclidean discriminant plane in an end-to-end manner, and the use of multiple F.C. layers allow a higher level of non-linearity for separation. In contrast, GDA uses a fixed kernel function and learns the discriminant independently. Methods such as SPDNet and GrNet are composed of many complex layers involving SVD, QR decompositions, and Gram–Schmidt orthogonalization and its derivatives are utilized as well. They increase in complexity as the number of layers increase by repeating these operations, which are not easily scalable in GPUs. In contrast, the proposed method offers competitive results employing a smaller set of parameters, benefiting a broader range of applications. By exploiting the tangent space properties, several practical advantages arise. For instance, the proposed model is naturally parallelizable. Also, it presents greater interpretability, providing a tool to understand its decisions by using the tangent space.

5. Conclusion

We proposed in this paper the Grassmann log model to map data from a Riemannian manifold to a vector space while maximizing discrimination capability for classification. The key idea is to formulate the Grassmann log map as a learnable model in such a way that it approximates well the manifold around the neighborhood of the data distribution. The proposed log model can be learned with Riemannian stochastic gradient descent; therefore it can be learned together with other powerful features such as cascaded convolutional layers. We performed classification experiments on multi-view hand shape recognition, face identification and and facial expression classification. Future works include the extension of this idea to other Riemannian manifolds; and to other applications, such as the modeling of matrices in signal processing and text modeling.

Acknowledgments

This work was partly supported by the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) scholarship.

References

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematica*, 80(2):199–220, 2004. 4

[2] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 4

[3] Relja Arandjelovic and Andrew Zisserman. All about vlad. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1578–1585, 2013. 2, 5

[4] Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*, 2019. 2, 4

[5] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013. 4

[6] L. E. Carvalho and A. von Wangenheim. 3d object recognition and classification: a systematic literature review. *Pattern Analysis and Applications*, Feb 2019. 1

[7] Hakan Cevikalp and Bill Triggs. Face recognition based on image sets. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2567–2573. IEEE, 2010. 7

[8] Abhinav Dhall, Roland Goecke, Jyoti Joshi, Karan Sikka, and Tom Gedeon. Emotion recognition in the wild challenge 2014: Baseline, data and protocol. In *Proceedings of the 16th international conference on multimodal interaction*, pages 461–466. ACM, 2014. 8

[9] Qingxiang Feng, Yicong Zhou, and Rushi Lan. Pairwise linear regression classification for image set retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4865–4872, 2016. 7

[10] Kazuhiro Fukui and Atsuto Maki. Difference subspace and its generalization for subspace-based methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(11):2164–2177, 2015. 1

[11] Jihun Hamm and Daniel D Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proceedings of the 25th international conference on Machine learning*, pages 376–383. ACM, 2008. 1, 7

[12] Jihun Hamm and Daniel D Lee. Extended grassmann kernels for subspace-based learning. In *Advances in neural information processing systems*, pages 601–608, 2009. 1, 7

[13] Mehrtash T Harandi, Mathieu Salzmann, and Richard Hartley. From manifold to manifold: Geometry-aware dimensionality reduction for spd matrices. In *European conference on computer vision*, pages 17–32. Springer, 2014. 7

[14] Munawar Hayat, Mohammed Bennamoun, and Senjian An. Deep reconstruction models for image set classification. *IEEE transactions on pattern analysis and machine intelligence*, 37(4):713–727, 2014. 7

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 7, 8

[16] Zhiwu Huang and Luc Van Gool. A riemannian network for spd matrix learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 7, 8

[17] 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 the IEEE conference on computer vision and pattern recognition*, pages 140–149, 2015. 7

[18] Zhiwu Huang, Jiqing Wu, and Luc Van Gool. Building deep networks on grassmann manifolds. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 1, 7, 8

[19] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Training deep networks with structured layers by matrix backpropagation. *arXiv preprint arXiv:1509.07838*, 2015. 7

[20] 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. 7

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015. 8

[22] Mengyi Liu, Shiguang Shan, Ruiping Wang, and Xilin Chen. Learning expressionlets on spatio-temporal manifold for dynamic facial expression recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1749–1756, 2014. 7, 8

[23] Mengyi Liu, Ruiping Wang, Zhiwu Huang, Shiguang Shan, and Xilin Chen. Partial least squares regression on grassmannian manifold for emotion recognition. In *Proceedings of the 15th ACM on International conference on multimodal interaction*, pages 525–530. ACM, 2013. 8

[24] Jiwen Lu, Gang Wang, Weihong Deng, Pierre Moulin, and Jie Zhou. Multi-manifold deep metric learning for image set classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1137–1145, 2015. 7

[25] Yui Man Lui, J Ross Beveridge, and Michael Kirby. Action classification on product manifolds. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 833–839. IEEE, 2010. 1

[26] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 6

[27] Thomas P Minka. Old and new matrix algebra useful for statistics. *See www. stat. cmu. edu/minka/papers/matrix. html*, 2000. 4

[28] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, 2010. 8

[29] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007. 2, 5

[30] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010. 2, 5

[31] Syed AA Shah, Uzair Nadeem, Mohammed Bennamoun, Ferdous Sohel, and Roberto Togneri. Efficient image set classification using linear regression based image reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 99–108, 2017. 7

[32] Naoya Sogi and Kazuhiro Fukui. Action recognition method based on sets of time warped arma models. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1773–1778. IEEE, 2018. 1

[33] Lincon S. Souza, Bernardo Bentes Gatto, and Kazuhiro Fukui. Enhancing discriminability of randomized time warping for motion recognition. In *Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on*, pages 77–80. IEEE, 2017. 1

[34] Lincon S. Souza, Hideitsu Hino, and Kazuhiro Fukui. 3d object recognition with enhanced grassmann discriminant analysis. In *Asian Conference on Computer Vision*, pages 345–359. Springer, 2016. 1

[35] Chendra Hadi Suryanto, Jing-Hao Xue, and Kazuhiro Fukui. Randomized time warping for motion recognition. *Image and Vision Computing*, 54:1–11, 2016. 1

[36] Hengliang Tan, Ying Gao, and Zhengming Ma. Regularized constraint subspace based method for image set classification. *Pattern Recognition*, 76:434–448, 2018. 1

[37] James Townsend. Differentiating the singular value decomposition, 2016. 4

[38] Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2273–2286, 2011. 1

[39] Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 33(11):2273–2286, 2011. 6

[40] Paul Viola and Michael J Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004. 7

[41] Ruiping Wang, Shiguang Shan, Xilin Chen, and Wen Gao. Manifold-manifold distance with application to face recognition based on image set. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 7

[42] Xing Wei, Yue Zhang, Yihong Gong, Jiawei Zhang, and Nanning Zheng. Grassmann pooling as compact homogeneous bilinear pooling for fine-grained visual classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 355–370, 2018. 1

[43] Xi Zhou, Kai Yu, Tong Zhang, and Thomas S Huang. Image classification using super-vector coding of local image descriptors. In *European conference on computer vision*, pages 141–154. Springer, 2010. 2, 5