Riemannian Coding and Dictionary Learning: Kernels to the Rescue

Mehrtash Harandi
Australian National University & NICTA*
Canberra, Australia
mehrtash.harandi@nicta.com.au

Mathieu Salzmann
Australian National University & NICTA*
Canberra, Australia
mathieu.salzmann@nicta.com.au

Abstract

While sparse coding on non-flat Riemannian manifolds has recently become increasingly popular, existing solutions either are dedicated to specific manifolds, or rely on optimization problems that are difficult to solve, especially when it comes to dictionary learning. In this paper, we propose to make use of kernels to perform coding and dictionary learning on Riemannian manifolds. To this end, we introduce a general Riemannian coding framework with its kernel-based counterpart. This lets us **(i)** *generalize beyond the special case of sparse coding;* **(ii)** *introduce efficient solutions to two coding schemes;* **(iii)** *learn the kernel parameters;* **(iv)** *perform unsupervised and supervised dictionary learning in a much simpler manner than previous Riemannian coding methods. We demonstrate the effectiveness of our approach on three different types of non-flat manifolds, and illustrate its generality by applying it to Euclidean spaces, which also are Riemannian manifolds.*

1. Introduction

Over the years, coding -in its broadest definition- has proven a crucial step in visual recognition systems [4, 9, 26, 44]. Many techniques have been investigated, such as bag of words [32, 15, 34, 1, 51, 38], sparse coding [43, 55], collaborative coding [58] and locality-based coding [56, 52]. All these techniques follow a similar flow: Given a dictionary of codewords, a query is associated to one or multiple dictionary elements with different weights. These weights, or *codes*, act as the new representation for the query and serve as input to a classifier.

This paper addresses the problem of coding and dictionary learning on Riemannian manifolds. Many powerful image and video descriptors, such as covariance descriptors [50, 21, 36, 7], normalized histograms [48], linear subspaces [53, 20, 39] and 2D shape outlines [29, 49, 18, 25], are known to lie on Riemannian manifolds. While it may therefore seem natural to extend coding techniques to such manifold-valued data, the nonlinear structure of Riemannian manifolds makes this task significantly more complicated than in Euclidean space.

Recently, a few approaches have been proposed to tackle the special case of sparse coding on Riemannian manifolds [57, 16, 47, 21, 23, 36, 20, 7, 6]. However, these techniques either are designed for specific manifolds [47, 21, 20, 7] and thus do not generalize well, or rely on the computation of the logarithm map [57, 16, 23, 36, 6], which makes coding and dictionary learning complicated, if tractable at all, for arbitrary Riemannian manifolds.

In this paper, we propose to formulate Riemannian coding and dictionary learning in Reproducing Kernel Hilbert Space (RKHS). With the rapidly growing number of known positive definite kernels on Riemannian manifolds [17, 24, 36, 25, 22], our approach generalizes to many manifolds, such as the manifold of Symmetric Positive Definite (SPD) matrices, the Grassmann manifold, the unit hypersphere and the shape manifold. Furthermore, since an RKHS is a linear space, this lets us derive simple, yet effective solutions for both coding and dictionary learning on Riemannian manifolds. Last but not least, as usual with kernel-based algorithms, the high dimensionality of the RKHS typically yields a more discriminative representation than the original data space, which translates into codes and dictionaries potentially better suited for visual recognition.

In contrast with existing nonlinear methods that tackle the special case of sparse coding in Euclidean space [12, 35, 42, 30], here we derive a general coding formulation, together with its kernel-based counterpart, for Riemannian manifolds, which Euclidean spaces are specific instances of. Our general formulation lets us study different coding strategies, such as sparse coding and locality-constrained coding. In this context, we introduce an approach to learning the kernel parameters, thus avoiding the need to tune them manually. Finally, we show how the dictionary can

*NICTA is funded by the Australian Government as represented by Department of Broadband, Communications and the Digital Economy, as well as by the ARC through the ICT Centre of Excellence program. This research was supported under Australian Research Councils Discovery Projects funding scheme (project DP150104645).

be learned in both an unsupervised and a supervised manner. In the latter case, we introduce an algorithm to jointly learn a classifier and the dictionary, thus effectively tuning the dictionary to the recognition problem at hand. Importantly, both the dictionary and the classifier updates can be achieved in closed-form.

We demonstrate the benefits of our method over existing Riemannian coding schemes on several manifold-valued datasets, including SPD matrices, linear subspaces and 2D shapes. Our experiments reveal that our general approach outperforms existing methods even when dedicated to the specific manifold of interest. We also illustrate the generality of our approach by evaluating it in Euclidean space.

2. Coding on Riemannian Manifolds

In this section, we derive a general formulation of Riemannian coding that encompasses many different coding schemes, and present an intrinsic version of this general formulation. To this end, we start by studying the case of Euclidean space.

In Euclidean space, let $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^N,\ \boldsymbol{d}_i \in \mathbb{R}^n$, be a given dictionary of N atoms, and $\boldsymbol{x} \in \mathbb{R}^n$ a query point. The problem of coding the query point can be expressed in a general manner as

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \Big\| \boldsymbol{x} - \sum\nolimits_{j=1}^{N} \alpha_j \boldsymbol{d}_j \Big\|_2^2 + \lambda \gamma(\boldsymbol{\alpha}; \boldsymbol{x}, \mathcal{D}) \qquad (1) \\ &\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C}, \end{aligned}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_N]^T \in \mathbb{R}^N$ is the vector of codes, γ is a prior on the codes $\boldsymbol{\alpha}$ and $\mathcal{C}$ is a set of constraints on $\boldsymbol{\alpha}$. Note that this formulation allows the prior to be dependent on both the query $\boldsymbol{x}$ and the dictionary $\mathcal{D}$. Although not explicitly written, this is also true for $\mathcal{C}$. In short, (1) tries to best approximate the query as a linear combination of dictionary elements while taking into account prior knowledge and constraints on the codes.

Typical special cases of this general formulation include the following examples:

Example 1. *The popular Bag of Words (BoW) model can be derived from* (1) *by defining* $\gamma(\alpha; \boldsymbol{x},\ \mathcal{D}) = 0$ *and* $\mathcal{C} = \{\boldsymbol{\alpha} \mid \alpha_j \in \{0, 1\}, \boldsymbol{\alpha}^T \mathbf{1}_N = 1\}$.

Example 2. *Sparse coding (via a Lasso formulation) can be obtained from* (1) *by defining* $\gamma(\boldsymbol{\alpha}; \boldsymbol{x}, \mathcal{D}) = \|\boldsymbol{\alpha}\|_1$ *and* $\mathcal{C} = \emptyset$.

Example 3. *Collaborative coding [58] is a special case of* (1) *where* $\gamma(\alpha; \boldsymbol{x},\ \mathcal{D}) = \|\boldsymbol{\alpha}\|_2^2$ *and* $\mathcal{C} = \emptyset$.

Example 4. *Locality-Constrained Linear Coding [52] can be derived from* (1) *by defining* $\gamma(\alpha; \boldsymbol{x},\ \mathcal{D}) = \sum_j \left(\exp\left(\sigma \|\boldsymbol{x} - \boldsymbol{d}_j\|_2\right) \alpha_j\right)^2$ *and* $\mathcal{C} = \{\boldsymbol{\alpha} \mid \sum_j \alpha_j = 1\}$.

Inspired by the Euclidean formulation (1), we can now derive a general formulation for coding on Riemannian manifolds. In this case, $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^N,\ \boldsymbol{d}_i \in \mathcal{M}$, becomes a dictionary on a Riemannian manifold $\mathcal{M}$, and similarly the point $\boldsymbol{x} \in \mathcal{M}$ is on the manifold. Riemannian coding can then be defined as

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \delta^2\Big(\boldsymbol{x}, \biguplus\nolimits_{j=1}^{N} \alpha_j \boldsymbol{d}_j\Big) + \lambda \gamma(\boldsymbol{\alpha}; \boldsymbol{x}, \mathcal{D}) \qquad (2) \\ &\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C}, \end{aligned}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^N$ is the vector of Riemannian codes, and $\delta : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+$ is a metric on $\mathcal{M}$. Moreover, $\biguplus : \mathcal{M} \times \cdots \times \mathcal{M} \times \mathbb{R} \times \mathbb{R} \cdots \times \mathbb{R} \to \mathcal{M}$ is an operator that combines multiple dictionary atoms $\{\boldsymbol{d}_j \in \mathcal{M}\}$ with weights $\{\alpha_j\}$ and generates a point $\hat{\boldsymbol{x}}$ on $\mathcal{M}$.

An interesting special case of (2) is its corresponding intrinsic formulation, obtained by defining δ as the geodesic distance[1] δ_g on the manifold. In this scenario, coding can be achieved by solving

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \Big\| \sum\nolimits_j \alpha_j \log_{\boldsymbol{x}}(\boldsymbol{d}_j) \Big\|_{\boldsymbol{x}}^2 + \lambda \gamma(\boldsymbol{\alpha}; \boldsymbol{x}, \mathcal{D}) \qquad (3) \\ &\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C}, \end{aligned}$$

where $\log_{\boldsymbol{x}}(\boldsymbol{d}) : \mathcal{M} \to T_{\boldsymbol{x}}\mathcal{M}$ is the logarithm map operator that maps a point $\boldsymbol{d} \in \mathcal{M}$ to the tangent space of $\mathcal{M}$ at another point $\boldsymbol{x} \in \mathcal{M}$, and $\|\cdot\|_{\boldsymbol{x}}$ is the norm induced by the Riemannian metric at $T_{\boldsymbol{x}}\mathcal{M}$. To understand how the reconstruction term in (3) was derived, we note that $\log_{\boldsymbol{x}}(\boldsymbol{x}) = \mathbf{0}$ and $\delta_g^2(\boldsymbol{x}, \boldsymbol{d}_j) = \|\log_{\boldsymbol{x}}(\boldsymbol{d}_j)\|_{\boldsymbol{x}}^2$. Moreover, since the tangent space at $\boldsymbol{x}$ is a vector space, one can simply choose vector space operators (i.e., addition and scalar product) to compute the $\biguplus$ operation. Therefore (3) is an instance of (2).

To the best of our knowledge, the reconstruction term employed in (3) was first introduced in [14] for the purpose of clustering on Riemannian manifolds. Subsequently, this reconstruction term was exploited independently in [23] and [6] to perform sparse coding on Riemannian manifolds. It is important to note, however, that (3) suffers from the trivial solution $\boldsymbol{\alpha} = \mathbf{0}$ if no constraint is imposed on the codes. As a consequence, the Riemannian sparse coding formulations of [23] and [6] both enforced the additional constraint $\mathcal{C} = \{\boldsymbol{\alpha} \mid \sum_j \alpha_j = 1\}$, thus making them diverge from their original Euclidean counterpart.

2.1. Intrinsic Locality Constrained Coding

While [23, 6] have studied an intrinsic formulation of sparse coding, our formulation in (3) allows us to consider other coding schemes. In particular, here, we focus on the case of Locality-Constrained Linear Coding (LLC) [52].

As mentioned in Example 4, LLC can be obtained by defining $\gamma(\boldsymbol{\alpha}; \boldsymbol{x}, \mathcal{D}) = \sum_{j=1}^N \left(\exp\left(\sigma \|\boldsymbol{x} - \boldsymbol{d}_j\|_2\right) \alpha_j\right)^2$, and $\mathcal{C} = \{\boldsymbol{\alpha} \mid \sum_j \alpha_j = 1\}$. In [52], however, it was shown that this formulation could be well approximated by replacing the dictionary $\mathcal{D}$ with a local dictionary $\mathcal{B}$ formed by the $N_{\mathcal{B}}$ nearest dictionary elements to the query. Here, we make use

[1] The length of the shortest curve between two points on the manifold.

of this approximate formulation[2].

More specifically, let $\mathcal{M}$ be a Riemannian manifold. Given a dictionary $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^{N}$, $\boldsymbol{d}_i \in \mathcal{M}$, we define intrinsic Locality Constrained Coding (*int-LCC*) as the solution of

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \Big\| \sum\nolimits_{j}^{N_{\mathcal{B}}} \alpha_j \log_{\boldsymbol{x}}(\boldsymbol{b}_j) \Big\|_{\boldsymbol{x}}^2 \\ &\text{s.t.} \quad \boldsymbol{\alpha}^T \mathbf{1} = 1 \ , \end{aligned} \tag{4}$$

where $\{\boldsymbol{b}_j\}_{j=1}^{N_{\mathcal{B}}}$ denotes the set of $N_{\mathcal{B}}$ atoms of $\mathcal{D}$ closest to the query $\boldsymbol{x}$, which can be identified using the geodesic distance on $\mathcal{M}$. In contrast to intrinsic sparse coding, int-LCC has a closed-form solution, which is unique as long as $N_{\mathcal{B}}$ is less than (or equal to) the dimensionality of $\mathcal{M}$. More specifically, let $\boldsymbol{B}$ be the matrix obtained by stacking the $\log_{\boldsymbol{x}}(\boldsymbol{b}_j)$ vectors, and $\boldsymbol{G}$ be the Riemannian metric tensor. The codes can then be obtained by solving the linear system $\boldsymbol{B}^T\boldsymbol{G}\boldsymbol{B}\widehat{\boldsymbol{\alpha}} = \mathbf{1}$ and normalizing the result to have unit ℓ_1 norm, *i.e.*, $\boldsymbol{\alpha} = \widehat{\boldsymbol{\alpha}} / \sum_{j=1}^{N_{\mathcal{B}}} |\widehat{\alpha}_j|$ (see the supplementary material for details).

Importantly, dictionary learning in either the intrinsic sparse coding formulation of [23, 6], or our int-LCC formulation above is complicated by the fact that the dictionary elements appear inside the logarithm map, which may be highly nonlinear, or not even have an analytic solution. In the following section, we introduce our kernel-based Riemannian coding formulation, whose sparse coding version does not require additional constraints, and, as discussed in Section 4, allows us to derive an efficient solution to dictionary learning.

3. Kernel-Based Riemannian Coding

To obtain a general formulation of Riemannian coding, but overcome the weaknesses of the intrinsic solution discussed in the previous section, we propose to perform coding in RKHS. This has the twofold advantage of yielding simple solutions to several popular coding techniques and of resulting in a potentially better representation than standard coding techniques due to the nonlinearity of the approach.

Let $\phi : \mathcal{M} \rightarrow \mathcal{H}$ be a mapping to an RKHS induced by the kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x})^T\phi(\boldsymbol{y})$. Coding in $\mathcal{H}$ can then be formulated as

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \Big\| \phi\big(\boldsymbol{x}\big) - \sum\nolimits_{j=1}^{N} \alpha_j \phi\big(\boldsymbol{d}_j\big) \Big\|_2^2 + \lambda \gamma(\boldsymbol{\alpha}; \phi\big(\boldsymbol{x}\big), \phi\big(\mathcal{D}\big)) \\ &\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C}. \end{aligned} \tag{5}$$

Expanding the reconstruction term in (5) yields

$$\begin{aligned} &\Big\| \phi(\boldsymbol{x}) - \sum\nolimits_{j=1}^{N} \alpha_j \phi(\boldsymbol{d}_j) \Big\|_2^2 = \phi(\boldsymbol{x})^T\phi(\boldsymbol{x}) \\ &- 2\sum\nolimits_{j=1}^{N} \alpha_j \phi(\boldsymbol{d}_j)^T\phi(\boldsymbol{x}) + \sum\nolimits_{i,j=1}^{N} \alpha_i\alpha_j \phi(\boldsymbol{d}_i)^T\phi(\boldsymbol{d}_j) \\ &= k(\boldsymbol{x}, \boldsymbol{x}) - 2\boldsymbol{\alpha}^T\boldsymbol{k}(\boldsymbol{x}, \mathcal{D}) + \boldsymbol{\alpha}^T\boldsymbol{K}(\mathcal{D}, \mathcal{D})\boldsymbol{\alpha}, \end{aligned} \tag{6}$$

where $\boldsymbol{k}(\boldsymbol{x}, \mathcal{D}) \in \mathbb{R}^N$ is the kernel vector evaluated between $\boldsymbol{x}$ and the dictionary atoms, and $\boldsymbol{K}(\mathcal{D}, \mathcal{D}) \in \mathbb{R}^{N \times N}$ is the kernel matrix evaluated between the dictionary atoms.

This shows that the reconstruction term in (5), common to most coding techniques, can be kernelized. More importantly, after kernelization, this term remains quadratic, convex and similar to its counterpart in Euclidean space. Next, we discuss the special cases of two popular coding techniques (*i.e.*, sparse coding and locality-constrained coding) and derive efficient solutions to their kernel extensions[3].

3.1. Kernel Sparse Coding (kSC)

As mentioned in Example 2, sparse coding can be obtained from our general formulation by not using any constraints and employing the prior $\gamma(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_1$. Since this prior only depends on $\boldsymbol{\alpha}$, the only step required to kernelize sparse coding is given in Eq. 6. Note that this also applies to any structured or group sparsity prior.

To derive an efficient solution to kernel sparse coding, we introduce the following theorem.

Theorem 3.1. *Consider the least-squares problem in an RKHS $\mathcal{H}$*

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \ \Big\| \phi(\boldsymbol{x}) - \sum\nolimits_{j=1}^{N} \alpha_j \phi(\boldsymbol{d}_j) \Big\|_2^2 \Leftrightarrow \\ &\min_{\boldsymbol{\alpha}} \ \boldsymbol{\alpha}^T\boldsymbol{K}(\mathcal{D}, \mathcal{D})\boldsymbol{\alpha} - 2\boldsymbol{\alpha}^T\boldsymbol{k}(\boldsymbol{x}, \mathcal{D}) + f(\boldsymbol{x}) \ , \end{aligned} \tag{7}$$

*where $f(\boldsymbol{x})$ is a constant function (*i.e.*, independent of $\boldsymbol{\alpha}$). Let $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$ be the SVD of the symmetric positive definite matrix $\boldsymbol{K}(\mathcal{D}, \mathcal{D})$. Then* (7) *is equivalent to the least-squares problem in $\mathbb{R}^N$*

$$\min_{\boldsymbol{\alpha}} \big\| \tilde{\boldsymbol{x}} - \tilde{\boldsymbol{D}}\boldsymbol{\alpha} \big\|_2^2 \ , \tag{8}$$

with $\tilde{\boldsymbol{D}} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{U}^T$ and $\tilde{\boldsymbol{x}} = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{U}^T\boldsymbol{k}(\boldsymbol{x}, \mathcal{D})$.

Proof. In supplementary material. □

This theorem lets us write kernel sparse coding as

$$\min_{\boldsymbol{\alpha}} \big\| \tilde{\boldsymbol{x}} - \tilde{\boldsymbol{D}}\boldsymbol{\alpha} \big\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \ , \tag{9}$$

which is a standard linear sparse coding problem. As a consequence, any efficient sparse solver such as SLEP [37] or SPAMS [40] can be employed to solve kernel sparse coding.

3.2. Kernel Locality-Constrained Coding (kLCC)

Following the intrinsic formulation introduced in Section 2.1, we make use of the local dictionary approximation of LLC, but this time in Hilbert space.

In Hilbert space, a local dictionary $\mathcal{B}$ can be obtained by performing kernel nearest neighbor between the original

[2] We could also derive the exact version of LLC on the manifold by replacing the ℓ_2 norm in the exponential with δ_g. However, as in Euclidean space, the approximate formulation can be solved more efficiently.

[3] From the examples in the previous section, it can easily be verified that other techniques, such as Bag of Words and collaborative coding, can also be kernelized in a similar manner.

dictionary elements and the query. This lets us write LLC in Hilbert space as

$$\min_{\boldsymbol{\alpha}} \|\phi(\boldsymbol{x}) - \phi(\mathcal{B})\boldsymbol{\alpha}\|_2^2 \quad \text{s.t.} \quad \boldsymbol{\alpha}^T\mathbf{1} = 1, \tag{10}$$

which has a form similar to (5) with no prior. This can then be directly kernelized by making use of Eq. 6.

The solution to kernel LCC can still be obtained in closed-form by solving the linear system $\big(\boldsymbol{K}(\mathcal{B},\mathcal{B}) - (\mathbf{1}^T \otimes \boldsymbol{k}(\boldsymbol{x},\mathcal{B}))\big)\boldsymbol{\alpha} = \mathbf{1}$ and normalizing this solution by its ℓ_1 norm to satisfy the constraint (see the supplementary material for details).

Note that while we considered the approximate version of LLC, the exact one can also be kernelized. To this end, we observe that

$$\exp\left(\sigma \left\|\phi(\boldsymbol{x}) - \phi(\boldsymbol{d}_j)\right\|_2\right) = \exp\left(\sigma\sqrt{k(\boldsymbol{x},\boldsymbol{x}) - 2k(\boldsymbol{x},\boldsymbol{d}_j) + k(\boldsymbol{d}_j,\boldsymbol{d}_j)}\right). \tag{11}$$

Thus, $\gamma(\boldsymbol{\alpha};\boldsymbol{x},\mathcal{D}) = \sum_j \left(\exp\left(\sigma\|\phi(\boldsymbol{x}) - \phi(\boldsymbol{d}_j)\|_2\right)\alpha_j\right)^2$ can be expressed solely in terms of kernel values, and so does the exact version of LLC. In practice, however, we favor the approximate version, which we refer to as kLCC, due to the simplicity of its solution.

3.3. Existence of Riemannian Kernels

While the kernel-based Riemannian coding formulation described above is indeed general, it is only useful if valid positive definite (p.d.) kernels can be defined on the manifold of interest. In this section, we discuss the existence of such Riemannian kernels.

Lemma 1. *Positive definite kernels exist on any Riemannian manifold.*

Proof. This lemma can be proved using the Nash-Kuiper embedding theorem [3], which states that every smooth m-dimensional manifold admits an isometric embedding into $\mathbb{R}^n$ with $n \geq m+1$. Let us denote such an embedding by $f : \mathcal{M} \to \mathbb{R}^n$. If a kernel $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, is *p.d.* in $\mathbb{R}^n$, then $k(f(\cdot), f(\cdot)) : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is a *p.d.* kernel on the manifold. □

Remark 1. *While the previous lemma ensures the existence of* p.d. *kernels on any Riemannian manifold, the form of the embedding is unknown in general. This makes the design of valid* p.d. *kernels on Riemannian manifolds an interesting and important problem.*

Alternatively, to design *p.d.* Riemannian kernels, one could try to exploit the tangent bundle of a Riemannian manifold. For example, for $\boldsymbol{p} \in \mathcal{M}$, the function

$$k(\boldsymbol{x},\boldsymbol{y}) = e^{-\sigma\|\log_{\boldsymbol{p}}(\boldsymbol{x}) - \log_{\boldsymbol{p}}(\boldsymbol{y})\|_{\boldsymbol{p}}^2} \tag{12}$$

is *p.d.* This is due to the fact that the tangent space at $\boldsymbol{p}$ is a vector space and hence the Gaussian kernel on it is *p.d.*

Remark 2. *While kernels obtained in this manner might prove effective in specific scenarios, we do* not *advocate their use in general. The first reason is that they depend on the logarithm map, which may be difficult to compute for some manifolds. The second reason can be understood by the following example.*

Example 5. *Let us consider* $\mathcal{S}^2$, i.e.*, the unit sphere, which is a Riemannian manifold with positive curvature. For the sake of argument, let* $\boldsymbol{p}$ *be the north pole, and* $\boldsymbol{x}$ *and* $\boldsymbol{y}$ *be two mirrored points very close to the south pole. After mapping to the tangent space* $T_{\boldsymbol{p}}\mathcal{M}$*, the distance between* $\log_{\boldsymbol{p}}(\boldsymbol{x})$ *and* $\log_{\boldsymbol{p}}(\boldsymbol{y})$ *will be very large, and thus the Gaussian kernel of Eq. 12 will be small. In other words, according to the kernel in Eq. 12,* $\boldsymbol{x}$ *and* $\boldsymbol{y}$ *are very dissimilar, which contradicts the fact that they are very close on* $\mathcal{S}^2$.

Remark 3. *According to Toponogov's theorem [3], for manifolds with negative curvature (*e.g.*, SPD manifolds), the distance on tangent spaces is bounded above* locally *by the geodesic distance. Therefore, the kernel in Eq. 12 will behave much better than in the previous example. We conjecture that this is the reason why the kernels introduced in [36] perform well in practice.*

Importantly, the number of known p.d. kernels on the manifolds that are most common in computer vision has recently been growing. Many kernels that do not rely on the logarithm map are now available for SPD manifolds [24, 36], Grassmann manifolds [17, 22] and the shape manifold [25]. The existence of such kernels therefore makes our approach practical in many scenarios.

3.4. Learning Gaussian Kernels

Many of the above-mentioned Riemannian kernels have the form of a Gaussian kernel[4], *i.e.*, $k(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\sigma\delta^2(\boldsymbol{x},\boldsymbol{y})\right)$, where δ is some chosen metric. As such, they depend on a parameter σ, whose value will influence the quality of the codes. In this section, we therefore introduce an approach to learning this parameter σ from training data.

To this end, let us assume that we are given a set of M training samples $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^M$. Following our general formulation, σ can be learned by solving the optimization problem

$$\min_{\sigma,\{\boldsymbol{\alpha}_i\}} \frac{1}{M}\sum_{i=1}^M L_\phi(\sigma,\boldsymbol{\alpha}_i,\boldsymbol{x}_i,\mathcal{D}) \quad \text{s.t.} \quad \boldsymbol{\alpha}_i \in \mathcal{C},\ \forall i \in [1,M], \tag{13}$$

where $\boldsymbol{\alpha}_i$ is the vector of sparse codes for the i^{th} training sample $\boldsymbol{x}_i$, and $L_\phi(\cdot)$ is the kernelized objective function defined in (5).

[4]Gaussian kernels are probably the most popular choice in learning methods due to their universality (*i.e.*, their induced space is rich and can approximate any target function arbitrarily close).

Note that (13) is not jointly convex in σ and $\{\boldsymbol{\alpha}_i\}_{i=1}^M$. Therefore, we follow the standard alternating minimization strategy that consists of iteratively fixing one variable (*i.e.*, either σ, or the $\boldsymbol{\alpha}_i$s) and solving for the other. With a fixed σ, the solution for each $\boldsymbol{\alpha}_i$ can be obtained as the solution of the chosen coding scheme.

Unfortunately, with a Gaussian kernel, (13) is ill-posed in terms of σ. More specifically, $\sigma = 0$ is a minimum of (13). This is due to the fact that, if $\sigma \to 0$, all samples in the induced Hilbert space $\mathcal{H}$ collapse to one point, *i.e.*, $\|\phi(\boldsymbol{x}) - \phi(\boldsymbol{y})\|^2 = k(\boldsymbol{x},\boldsymbol{x}) - 2k(\boldsymbol{x},\boldsymbol{y}) + k(\boldsymbol{y},\boldsymbol{y}) = 0$. To avoid this trivial solution, we propose to search for a σ that not only minimizes the kernel coding cost, but also maximizes a measure of discrepancy between the dictionary atoms in $\mathcal{H}$. In other words, we search for a Hilbert space $\mathcal{H}$ that simultaneously yields a diverse dictionary and a good representation of the data. To this end, we define the discrepancy between the dictionary atoms in $\mathcal{H}$ as

$$J_\phi(\mathcal{D},\sigma) = \frac{1}{N^2}\sum\nolimits_{(r,s)=1}^{N} \|\phi(\boldsymbol{d}_r) - \phi(\boldsymbol{d}_s)\|_2^2 \tag{14}$$
$$= \frac{1}{N^2}\sum\nolimits_{(r,s)=1}^{N} \left(k(\boldsymbol{d}_r,\boldsymbol{d}_r) - 2k(\boldsymbol{d}_r,\boldsymbol{d}_s) + k(\boldsymbol{d}_s,\boldsymbol{d}_s)\right).$$

Given $\{\boldsymbol{\alpha}_i\}$, this lets us obtain σ by solving the optimization problem

$$\min_{\sigma} \frac{\frac{1}{M}\sum_{i=1}^{M} L_\phi(\sigma,\boldsymbol{\alpha}_i,\boldsymbol{x}_i,\mathcal{D})}{J_\phi(\mathcal{D},\sigma)} \quad \text{s.t.} \quad \boldsymbol{\alpha}_i \in \mathcal{C},\ \forall i \in [1,M]\ . \tag{15}$$

Since the objective function of (15) is not convex in σ, we utilize a gradient-based trust-region method to obtain a local minimum of this problem. The gradients of L_ϕ and J_ϕ (as required by the trust-region method) are related to $\partial k(\boldsymbol{x},\boldsymbol{y})/\partial\sigma = -\delta^2(\boldsymbol{x},\boldsymbol{y})k(\boldsymbol{x},\boldsymbol{y})$. Note that, for kSC and kLCC, the prior $\gamma(\cdot)$ does not depend on the kernel and can thus be omitted when updating σ. This is also the case of the constraint set $\mathcal{C}$.

4. Riemannian Dictionary Learning

In this section, we discuss how a dictionary can be learned in our Riemannian coding framework. In particular, given a set of M training samples $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^M$, we investigate dictionary learning in the unsupervised and supervised scenarios.

4.1. Unsupervised Dictionary Learning

In many cases, it is beneficial not only to compute the codes for a given dictionary, but also to optimize the dictionary to best suit the problem at hand. Here, we show how this can be done in our general formulation. Given training samples, we follow an alternating optimization strategy to update the codes and the dictionary. Since obtaining the codes with a given dictionary was discussed in the previous section, here we focus on the dictionary update.

With fixed codes for the training data (and a fixed kernel parameter), learning the dictionary can be expressed as solving the optimization problem

$$\min_{\mathcal{D}} \frac{1}{M}\sum\nolimits_{i=1}^{M} L_\phi(\mathcal{D};\boldsymbol{\alpha}_i,\boldsymbol{x}_i) \quad \text{s.t.} \quad \boldsymbol{\alpha}_i \in \mathcal{C},\ \forall i \in [1,M]. \tag{16}$$

Here, we make use of the *Representer theorem* [45] which enables us to express the dictionary as a linear combination of the training samples in RKHS. That is

$$\phi(\boldsymbol{d}_j) = \sum\nolimits_{i=1}^{M} \boldsymbol{v}_{i,j}\phi(\boldsymbol{x}_i), \tag{17}$$

where $\{\boldsymbol{v}_{i,j}\}$ is the set of weights, now corresponding to our new unknowns. By stacking these weights for the M samples and the N dictionary elements in a matrix $\boldsymbol{V}_{M\times N}$, we have

$$\phi(\mathcal{D}) = \phi(\mathcal{X})\boldsymbol{V}\ . \tag{18}$$

For kSC and kLCC, the only term that depends on the dictionary is the reconstruction error (*i.e.*, the first term in the objective of (5)). Given the matrix of sparse codes $\boldsymbol{A}_{N\times M} = [\boldsymbol{\alpha}_1|\boldsymbol{\alpha}_2|\cdots|\boldsymbol{\alpha}_M]$, this term can be expressed as

$$R(\boldsymbol{V}) = \left\|\phi(\mathcal{X}) - \phi(\mathcal{X})\boldsymbol{V}\boldsymbol{A}\right\|_F^2 \tag{19}$$
$$= \mathrm{Tr}\left(\phi(\mathcal{X})(\mathbf{I}_M - \boldsymbol{V}\boldsymbol{A})(\mathbf{I}_M - \boldsymbol{V}\boldsymbol{A})^T\phi(\mathcal{X})^T\right)$$
$$= \mathrm{Tr}\left(\boldsymbol{K}(\mathcal{X},\mathcal{X})(\mathbf{I}_M - \boldsymbol{V}\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{V}^T + \boldsymbol{V}\boldsymbol{A}\boldsymbol{A}^T\boldsymbol{V}^T)\right)\ .$$

The new dictionary, fully defined by $\boldsymbol{V}$, can then be obtained by zeroing out the gradient of $R(\boldsymbol{V})$ w.r.t. $\boldsymbol{V}$. This yields

$$\nabla R(\boldsymbol{V}) = 0 \Leftrightarrow \boldsymbol{V} = (\boldsymbol{A}\boldsymbol{A}^T)^{-1}\boldsymbol{A} = \boldsymbol{A}^\dagger\ . \tag{20}$$

In the case of kLCC, we update the full dictionary at once. To this end, for each training sample i, we simply set to 0 the codes corresponding to the elements that do not belong to the local dictionary $\mathcal{B}_i$ specific to the i^{th} sample.

4.2. Supervised Dictionary Learning

In the context of recognition, where labeled data is available, one would typically like to learn a dictionary that not only yields accurate reconstruction of the training samples, but also generates discriminative codes. To this end, a few methods have proposed to jointly learn a classifier with the codes and the dictionary for the special case of linear sparse coding [41, 28]. Here, we show how supervised data can be efficiently taken into account in our general Riemannian kernel coding framework.

To this end, given data belonging to S different classes, let $\boldsymbol{l}_i$ be the S-dimensional binary vector encoding the label of training sample $\boldsymbol{x}_i$, *i.e.*, the j^{th} element of $\boldsymbol{l}_i$ is set to 1 if sample i belongs to class j and 0 otherwise. We then make use of a linear classifier acting on the codes, whose prediction can thus be written as $\hat{\boldsymbol{l}} = \boldsymbol{W}\boldsymbol{\alpha}$, where $\boldsymbol{W}$ is the decision hyperplane[5]. By employing the square loss,

[5]Note that in practice, we truly use $\hat{\boldsymbol{l}} = \boldsymbol{W}\boldsymbol{\alpha} + \boldsymbol{b}$, but, with a slight abuse of notation, the bias $\boldsymbol{b}$ can be included in $\boldsymbol{W}$ by adding one column

learning in Hilbert space can then be written as

$$\min_{\boldsymbol{W},\mathcal{D},\{\boldsymbol{\alpha}_i\}} \frac{1}{M}\sum\nolimits_{i=1}^{M} L_\phi(\mathcal{D},\boldsymbol{\alpha}_i;\boldsymbol{x}_i) + \frac{\eta}{M}\sum\nolimits_{i=1}^{M}\|\boldsymbol{l}_i-\boldsymbol{W}\boldsymbol{\alpha}_i\|_2^2+\rho\|\boldsymbol{W}\|_F^2\ , \quad \text{s.t.}\ \ \boldsymbol{\alpha}_i\in\mathcal{C},\ \forall i\in[1,M], \tag{21}$$

where we utilize a simple regularizer on the parameters $\boldsymbol{W}$.

Following an alternating minimization approach, the classifier parameters $\boldsymbol{W}$ can be obtained by solving the linear system

$$\boldsymbol{W}=\left(\sum\nolimits_{i=1}^{M}\boldsymbol{l}_i\boldsymbol{\alpha}_i^T\right)\left(\frac{\eta}{M}\sum\nolimits_{i=1}^{M}\boldsymbol{\alpha}_i\boldsymbol{\alpha}_i^T+\rho\mathbf{I}_N\right)^{-1} \tag{22}$$

arising from zeroing out the gradient of the second and third terms in the objective function. Being unaffected by the discriminative term, the dictionary remains encoded by $\boldsymbol{V}=\boldsymbol{A}^\dagger$ (see Eq. 20). Computing the codes, however, must be modified by taking the discriminative term into account. Due to the least-squares form of this term, this can effectively be achieved by replacing the reconstruction term of Eq. 6 with

$$\boldsymbol{\alpha}_i^T\boldsymbol{R}\boldsymbol{\alpha}_i-2\boldsymbol{\alpha}_i^T\boldsymbol{q}+\tilde{f}(\boldsymbol{x}_i,\boldsymbol{l}_i)\ , \tag{23}$$

where $\tilde{f}$ is a function independent of the codes, and

$$\boldsymbol{R}=\boldsymbol{V}^T\boldsymbol{K}(\mathcal{X},\mathcal{X})\boldsymbol{V}+\eta\boldsymbol{W}^T\boldsymbol{W},\qquad \boldsymbol{q}=\boldsymbol{V}^T\boldsymbol{k}(\boldsymbol{x}_i,\mathcal{X})+\eta\boldsymbol{W}^T\boldsymbol{l}_i\ . \tag{24}$$

Due to the very similar form of this term compared to the reconstruction term, it can easily be verified that the solutions kSC and kLCC can still be obtained efficiently in the same manner as in Sections 3.1 and 3.2. The pseudocode of our algorithm is given in supplementary material.

5. Related Work

In light of our method, we now discuss related coding techniques, both Riemannian and nonlinear ones. Note that these methods have only considered the special case of sparse coding, which, as will be shown in our experiments, is typically outperformed by locality-constrained coding.

In the context of Riemannian coding, several methods have been designed for specific manifolds. In this scenario, one possible approach consists of embedding the Riemannian manifold into a vector space using a manifold-specific transformation, followed by coding and dictionary learning in the resulting vector space [47, 20]. In [7], another strategy that exploits the specific form of the geodesic distance on the SPD manifold was introduced to perform sparse coding. While effective in their context, the manifold-specific methods are typically difficult to generalize to arbitrary manifolds.

Nevertheless, some methods that apply to general Riemannian manifolds have also been proposed. A simple, yet natural idea to address this general scenario is to flatten the Riemannian manifold via a fixed tangent space at a chosen point $\boldsymbol{p}$ on the manifold, called the center of projection. This can be achieved via the logarithm map $\log_{\boldsymbol{p}}(\cdot)$. This idea was exploited in [57] and [16] (although only demonstrated on SPD manifolds) for the special case of sparse coding, which can then be expressed as

$$\min_{\boldsymbol{\alpha}}\left\|\log_{\boldsymbol{p}}(\boldsymbol{x})-\sum_{j=1}^{N}\boldsymbol{\alpha}_j\log_{\boldsymbol{p}}(\boldsymbol{d}_j)\right\|_{\boldsymbol{p}}^2+\lambda\|\boldsymbol{\alpha}\|_1. \tag{25}$$

Following the terminology of [2], we will refer to this approach as *log-Euclidean* coding and will use it as a baseline in our experiments.

This log-Euclidean coding approach suffers from the fact that, since it uses a single tangent space, only the distances to the center of projection are equal to the true geodesic distances, as illustrated by our unit sphere example in Section 3.3. The intrinsic sparse coding formulation introduced in [23] and [6], and discussed in its more general form in Section 2, alleviates this issue by considering the tangent space at the query $\boldsymbol{x}$. However, dictionary learning, achieved in [23] by gradient descent along the geodesics, may become excessively complicated for some manifolds. Indeed, following an alternating minimization approach, the update of $\boldsymbol{d}_j$ at iteration t has the form

$$\boldsymbol{d}_j^{(t+1)}=\exp_{\boldsymbol{d}_j^{(t)}}(-\eta\Delta)\ , \tag{26}$$

where η is the step size and the tangent vector $\Delta:\mathbb{R}\rightarrow T_{\boldsymbol{d}_j}\mathcal{M}$ is computed as the gradient of the objective function[6]. Since the objective function consists of the reconstruction error in (3), it depends on the logarithm map, which is highly nonlinear, or does not even have an analytic expression (*e.g.*, for Grassmann manifolds). As a consequence, and as acknowledged in [23], dictionary learning is far from obvious in the general case and must rely on numerical gradient approximations.

Although specifically designed for SPD manifolds and for the special case of sparse coding, the methods of [21] and [36] are probably the closest to our work, in the sense that they also exploit kernels: the Stein kernel [46] in [21], which is specific for SPD matrices, and more general log-Euclidean kernels in [36], whose weaknesses for general manifolds were discussed in Section 3.3. In contrast to our approach, however, both methods make use of a gradient descent strategy to learn the dictionary, which may yield dictionary elements outside the SPD manifold. This is circumvented by a further projection to the manifold, which **(i)** does not guarantee convergence of the algorithms; and **(ii)** is non-trivial to generalize to arbitrary manifolds.

Aside from Riemannian sparse coding methods, our

to $\boldsymbol{W}$ and concatenating a value 1 to $\boldsymbol{\alpha}$.

[6] On an abstract Riemannian manifold $\mathcal{M}$, the gradient of a smooth real function f at a point $x\in\mathcal{M}$, denoted by $\mathrm{grad}f(x)$, is the element of $T_x(\mathcal{M})$ satisfying $\langle\mathrm{grad}f(x),\zeta\rangle_x=Df_x[\zeta]$ for all $\zeta\in T_x(\mathcal{M})$, where $Df_x[\zeta]$ denotes the directional derivative of f at x in the direction of ζ.

work is of course also related to the nonlinear sparse coding and dictionary learning techniques introduced for Euclidean space. For instance, the notion of kernel sparse coding was studied for object and face recognition in [12] and [35], and more recently for general purpose in [30]. Dictionary learning in RKHS was also recently tackled in [42]. Here, we go beyond these works by considering the more general scenario of Riemannian manifolds, which Euclidean spaces are instances of, as well as a more general coding formulation. Furthermore, we also derive efficient algorithms for coding and supervised dictionary learning, as well as an approach to learn the kernel parameters.

6. Experiments

We demonstrate the effectiveness of our kernel-based techniques on four different types of Riemannian manifolds (including Kendall's shape manifold in supplementary material). We refer to the different algorithms evaluated in our experiments as:

logEuc-SC: log-Euclidean sparse coding as described in (25) and employed in [57, 16].

logEuc-LCC: log-Euclidean locality-constrained coding. Mapping to a single tangent space followed by approximate LLC [52].

int-SC: intrinsic sparse coding [23, 6].

int-LCC: our intrinsic extension of LLC to Riemannian manifolds (Section 2.1).

kSC: our kernel sparse coding with unsupervised dictionary learning (Sections 3.1 and 4.1).

kLCC: our kernel locality-constrained coding with unsupervised dictionary learning (Sections 3.2 and 4.1).

kSSC: our kernel supervised sparse coding (Sections 3.1 and 4.2).

kSLCC: our kernel supervised locality-constrained coding (Sections 3.2 and 4.2).

For all the unsupervised methods, we trained a separate ridge regression classifier on the codes (with the same form used in our supervised algorithm in Section 4.2) to perform classification. For the supervised ones, we simply used the learned classifier to obtain our results.

For the kernel-based methods, all the experiments were performed using manifold-specific Gaussian-like kernels. To obtain an initial dictionary, we used kmeans in a fixed tangent space of the manifold followed by a projection using the exponential map. The bandwidth of the Gaussian kernel was then learned from training data with this initial dictionary by using the algorithm described in Section 3.4. To this end, we used a gradient-based trust-region method provided by the fmincon matlab function.

For the intrinsic methods, since the manifolds studied in this work do not necessarily result in an analytic form of the gradient required in Eq. 26, we used a modified version of intrinsic kmeans to learn the dictionary. More specifically, starting from the initial dictionary described above, we iteratively computed the intrinsic codes and performed a weighted Karcher mean atom by atom to update the dictionary. The weights of the weighted Karcher mean were chosen as the absolute value of the intrinsic codes. In practice, we observed that this procedure yields a better dictionary than the one obtained with a simple intrinsic kmeans. On a related note, the geodesic distance was used to determine the local dictionary atoms for int-LCC.

In practice, we found that the accuracy of all the methods saturates to a maximum value as the number of dictionary atoms increases (see the curves provided in supplementary material). Therefore, in our experiments, we set this number to a large enough value (the same for all the algorithms) so that all the methods have reached saturation, and thus perform at, or close to, their best.

6.1. The SPD Manifold

We evaluated our different techniques on two challenging classification datasets where the images are represented with region covariance descriptors (RCovDs) [50], which lie on SPD manifolds. In our experiments, we used the Stein kernel of [46]. In addition to the baselines mentioned above, we compare our methods against the state-of-the-art infinite-dimensional RCovDs of [19], denoted by $S_{\mathcal{H}}$-SVM, and Discriminative Covariance Learning (CDL) [53].

Virus Classification:
As a first experiment, we used the virus dataset of [33] which contains 15 different virus classes. We used the 10 splits provided with the dataset in a leave-one-out manner, *i.e.*, 10 experiments with 9 splits for training and 1 split as query. Following [19], Gabor filters were used to build the RCovDs.

The results of the different methods are reported in the middle column of Table 1. Note that the log-Euclidean solutions achieve the lowest accuracies among all the studied coding schemes, which is not surprising given the distortion induced by flattening the manifold at a single tangent space. Intrinsic coding approaches yield higher accuracies, with our int-LCC algorithm outperforming the int-SC of [6, 23]. This accuracy is further increased by all of our kernel methods, with the maximum accuracy of 82.0% obtained by our kSLCC algorithm, which, to the best of our knowledge, represents the state-of-the-art for this dataset.

Material Categorization:
We then performed material recognition using the KTH-TIPS2b dataset [5] which contains 4752 samples of 11 materials captured under different illuminations, poses and scales. We utilized the setup and features of [19] where each sample was encoded with a 23×23 RCovD.

Method	Virus	KTH-TIPS2-b
CDL [53]	$69.5\% \pm 3.1$	$76.3\% \pm 5.1$
$S_{\mathcal{H}}$-SVM [19]	$81.2\% \pm 2.9$	$80.1\% \pm 4.6$
logEuc-SC	$68.3\% \pm 3.9$	$67.8\% \pm 2.7$
logEuc-LCC	$72.3\% \pm 3.5$	$75.9\% \pm 3.1$
int-SC	$73.3\% \pm 3.6$	$78.7\% \pm 4.0$
int-LCC	$74.0\% \pm 3.2$	$80.5\% \pm 4.9$
kSC	$78.5\% \pm 2.7$	$78.8\% \pm 4.8$
kLCC	$79.4\% \pm 2.9$	$79.8\% \pm 4.6$
kSSC	$81.7\% \pm 2.8$	$79.9\% \pm 4.6$
kSLCC	$\mathbf{82.0\% \pm 2.8}$	$\mathbf{81.2\% \pm 5.2}$

Table 1: **Coding on SPD manifolds.**

Method	Hand Gesture	Mice Behavior
GDA [17]	82.4%	$81.7\% \pm 2.0$
SSSC [39]	83.1%	N/A
logEuc-SC	62.8%	$66.9\% \pm 3.1$
logEuc-LCC	64.9%	$63.1\% \pm 3.0$
int-SC	71.5%	$66.0\% \pm 2.2$
int-LCC	81.9%	$86.5\% \pm 1.6$
kSC	86.1%	$88.5\% \pm 1.1$
kLCC	85.4%	$89.8\% \pm 1.0$
kSSC	89.7%	$90.5\% \pm 0.6$
kSLCC	**90.7%**	$\mathbf{90.8\% \pm 0.6}$

Table 2: **Coding on Grassmannians.**

Method	YALE-B	C101
SRC [54]	80.5%	70.7%
LC-KSVD [28]	95.0%	73.6%
kSC	96.9%	75.1%
kLCC	97.2%	75.4%
kSSC	98.2%	75.7%
kSLCC	**98.4%**	**76.2%**

Table 3: **Coding in Euclidean space.**

The last column of Table 1 provides the recognition accuracies averaged over the four splits of this dataset. As before, the log-Euclidean approaches yield the lowest accuracies among all the studied coding schemes. Here, however, the intrinsic coding methods perform roughly on par with their unsupervised kernel counterparts. The highest accuracy is still achieved by our supervised kSLCC method.

In [8], a different test protocol was used on KTH-TIPS2-b. Following this protocol, kSSC and kSLCC achieved 71.2% and 71.7% accuracy, respectively. Note that, while not state-of-the-art, this outperforms the accuracy of the deep convolutional network DeCAF [10], which was reported as 70.7% in [8], despite the fact that we rely on much simpler features.

6.2. The Grassmann Manifold

We then performed experiments on the Grassmannian, which is the manifold of linear subspaces. Here, we used the RBF projection kernel of [22]. In addition to our coding baselines, we compare our results with the state-of-the-art Semi-Supervised Spectral Clustering (SSSC) [39] and Grassmannian Discriminant Analysis (GDA) [17].

Hand Gesture Recognition:
On the Grassmannian, we first used the Cambridge hand-gesture dataset [31], which consists of 900 image sequences of 9 gesture classes. We employed the descriptors (linear subspaces based on HoG features) and the test protocol of [39], where the first 80 sequences of each class are used as test data, with the remaining 20 as training data.

The recognition accuracies are shown in the middle column of Table 2. Note that our int-LCC method performs significantly better than int-SC. Note also that our kernel coding schemes outperform the state-of-the-art methods. The maximum accuracy of 90.7% is achieved by kSLCC, which is more than 7% better than the state-of-the-art SSSC.

Mouse Behavior Analysis:
We performed classification on the Grassmannian using the mice behavior dataset [27], which contains 2000 videos depicting 8 behaviors of mice with different coating colors, sizes and genders. In each video, we performed background subtraction to extract the region containing the mouse in each frame. These regions were then resized to 48×48, and the video represented with an order 6 subspace. We randomly chose 25 videos from each behavior for training and used the remaining videos for testing.

The recognition accuracies averaged over 10 random partitions are shown in the right column of Table 2. These results confirm the trends of the previous experiments, with kernel coding solutions outperforming the other methods.

6.3. The Euclidean Space

Finally, as a proof of concept, we evaluated our kernel coding schemes in Euclidean space, which is a flat Riemannian manifold. As baselines, we employed SRC [54] and the state-of-the-art LC-KSVD [28], which is a supervised extension of sparse coding and dictionary learning. Note that, in Euclidean space, log-Euclidean coding and intrinsic coding just boil down to standard linear coding techniques, and are thus superseded by the baselines. For the comparison to be fair, we used the data and partitions provided by the authors of [28] for the extended YALE-B dataset [13] and Caltech101 [11]. For the experiment on YALE-B, following [28], we learned a dictionary of size 570 for each algorithm. For the experiment on Caltech101, 30 images per category were used for training.

Table 3 summarizes the results of these two experiment. Note that both kSC and kLLC, which are unsupervised, outperform the state-of-the-art LC-KSVD method. With supervision, our formulation boosts the performance even further with maximum accuracies achieved by kSLCC.

7. Conclusions and Future Work

In this paper, we have introduced a general framework for coding on Riemannian manifolds. In particular, we have shown how the use of kernels could make Riemannian coding and dictionary learning easier than intrinsic formulations. Our experiments on several manifolds have demonstrated the benefits of our kernel formulation over existing Riemannian coding strategies, as well as over other classification algorithms for Riemannian manifolds. In particular, our supervised kernel locality-constrained coding scheme performed consistently well in all our experiments. In the future, we intend to exploit the notion of multiple kernel learning in our framework, thus allowing us to combine multiple RKHSs to obtain a richer space for coding.

References

[1] A. Agarwal and B. Triggs. Hyperfeatures–multilevel local coding for visual recognition. In *Proc. European Conference on Computer Vision (ECCV)*, pages 30–43. Springer, 2006.

[2] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic resonance in medicine*, 56(2):411–421, 2006.

[3] M. Berger. *A panoramic view of Riemannian geometry*. Springer, 2003.

[4] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2559–2566, 2010.

[5] B. Caputo, E. Hayman, and P. Mallikarjuna. Class-specific material categorisation. In *Proc. Int. Conference on Computer Vision (ICCV)*, volume 2, pages 1597–1604, 2005.

[6] H. Cetingul, M. Wright, P. Thompson, and R. Vidal. Segmentation of high angular resolution diffusion mri using sparse riemannian manifold clustering. *IEEE Transactions on Medical Imaging*, 33(2):301–317, Feb 2014.

[7] A. Cherian and S. Sra. Riemannian sparse coding for positive definite matrices. In *Proc. European Conference on Computer Vision (ECCV)*, pages 299–314. Springer, 2014.

[8] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3606–3613, June 2014.

[9] A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 921–928, 2011.

[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proc. Int. Conference on Machine Learning (ICML)*, 2013.

[11] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding (CVIU)*, 106(1):59–70, 2007.

[12] S. Gao, I. W.-H. Tsang, and L.-T. Chia. Kernel sparse representation for image classification and face recognition. In *Proc. European Conference on Computer Vision (ECCV)*, pages 1–14. Springer, 2010.

[13] A. S. Georghiades, P. N. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[14] A. Goh and R. Vidal. Clustering and dimensionality reduction on riemannian manifolds. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7, 2008.

[15] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. Int. Conference on Computer Vision (ICCV)*, volume 2, pages 1458–1465. IEEE, 2005.

[16] K. Guo, P. Ishwar, and J. Konrad. Action recognition using sparse representation on covariance manifolds of optical flow. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 188–195, 2010.

[17] J. Hamm and D. D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 376–383, 2008.

[18] O. C. Hamsici and A. M. Martinez. Rotation invariant kernels and their application to shape analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1985–1999, 2009.

[19] M. Harandi, M. Salzmann, and F. Porikli. Bregman divergences for infinite dimensional covariance matrices. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1, June 2014.

[20] M. Harandi, C. Sanderson, C. Shen, and B. Lovell. Dictionary learning and sparse coding on grassmann manifolds: An extrinsic solution. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 3120–3127. IEEE, 2013.

[21] M. T. Harandi, R. Hartley, B. C. Lovell, and C. Sanderson. Sparse coding on symmetric positive definite manifolds using bregman divergences. *IEEE Transactions on Neural Networks and Learning Systems*, pages –, 2015.

[22] M. T. Harandi, M. Salzmann, S. Jayasumana, R. Hartley, and H. Li. Expanding the family of grassmannian kernels: An embedding perspective. In *Proc. European Conference on Computer Vision (ECCV)*, pages 408–423. Springer International Publishing, 2014.

[23] J. Ho, Y. Xie, and B. Vemuri. On a nonlinear generalization of sparse coding and dictionary learning. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 1480–1488, 2013.

[24] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi. Kernel methods on the riemannian manifold of symmetric positive definite matrices. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 73–80, 2013.

[25] S. Jayasumana, M. Salzmann, H. Li, and M. Harandi. A framework for shape analysis via hilbert space embedding. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 1249–1256. IEEE, 2013.

[26] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, 2012.

[27] H. Jhuang, E. Garrote, X. Yu, V. Khilnani, T. Poggio, A. D. Steele, and T. Serre. Automated home-cage behavioural phenotyping of mice. *Nature communications*, 1:68, 2010.

[28] Z. Jiang, Z. Lin, and L. Davis. Label consistent k-svd: Learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2651–2664, 2013.

[29] D. G. Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London Mathematical Society*, 16(2):81–121, 1984.

[30] M. Kim. Efficient kernel sparse coding via first-order smooth optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 25(8):1447–1459, Aug 2014.

[31] T.-K. Kim and R. Cipolla. Canonical correlation analysis of video volume tensors for action categorization and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8):1415–1428, 2009.

[32] J. Koenderink and A. Van Doorn. The structure of locally orderless images. *Int. Journal of Computer Vision (IJCV)*, 31(2-3):159–168, 1999.

[33] G. Kylberg and I.-M. Sintorn. Evaluation of noise robustness for local binary pattern descriptors in texture classification. *EURASIP Journal on Image and Video Processing*, 2013(1):1–20, 2013.

[34] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178, 2006.

[35] H. Li, Y. Gao, and J. Sun. Fast kernel sparse representation. In *International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 72–77. IEEE, 2011.

[36] P. Li, Q. Wang, W. Zuo, and L. Zhang. Log-euclidean kernels for sparse representation and dictionary learning. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 1601–1608, 2013.

[37] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.

[38] L. Liu, L. Wang, and X. Liu. In defense of soft-assignment coding. In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 2486–2493. IEEE, 2011.

[39] A. Mahmood, A. Mian, and R. Owens. Semi-supervised spectral clustering for image set classification. In *CVPR*, 2014.

[40] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research (JMLR)*, 11:19–60, 2010.

[41] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A. Zisserman, et al. Supervised dictionary learning. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1033–1040, 2009.

[42] H. Nguyen, V. Patel, N. Nasrabadi, and R. Chellappa. Design of non-linear kernel dictionaries for object recognition. *IEEE Transactions on Image Processing*, 22(12):5123–5135, Dec 2013.

[43] B. A. Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[44] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the fisher vector: Theory and practice. *Int. Journal of Computer Vision (IJCV)*, 105(3):222–245, 2013.

[45] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Computational learning theory*, pages 416–426. Springer, 2001.

[46] S. Sra. A new metric on the manifold of kernel matrices with application to matrix geometric means. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 144–152, 2012.

[47] S. Sra and A. Cherian. Generalized dictionary learning for symmetric positive definite matrices with application to nearest neighbor retrieval. In *Machine Learning and Knowledge Discovery in Databases*, pages 318–332. Springer, 2011.

[48] A. Srivastava, I. Jermyn, and S. Joshi. Riemannian analysis of probability density functions with applications in vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

[49] A. Srivastava, S. H. Joshi, W. Mio, and X. Liu. Statistical shape analysis: Clustering, learning, and testing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):590–602, 2005.

[50] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *Proc. European Conference on Computer Vision (ECCV)*, pages 589–600. Springer, 2006.

[51] J. C. van Gemert, C. J. Veenman, A. W. Smeulders, and J.-M. Geusebroek. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1271–1283, 2010.

[52] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3360–3367, 2010.

[53] R. Wang, H. Guo, L. S. Davis, and Q. Dai. Covariance discriminative learning: A natural and efficient approach to image set classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2496–2503, 2012.

[54] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

[55] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1794–1801, 2009.

[56] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, volume 9, page 1, 2009.

[57] C. Yuan, W. Hu, X. Li, S. Maybank, and G. Luo. Human action recognition under log-euclidean Riemannian metric. In H. Zha, R.-i. Taniguchi, and S. Maybank, editors, *Proc. Asian Conference on Computer Vision (ACCV)*, volume 5994 of *Lecture Notes in Computer Science*, pages 343–353. Springer Berlin Heidelberg, 2010.

[58] D. Zhang, M. Yang, and X. Feng. Sparse representation or collaborative representation: Which helps face recognition? In *Proc. Int. Conference on Computer Vision (ICCV)*, pages 471–478. IEEE, 2011.