Abstract

This paper takes a step forward in image and video coding by extending the well-known Vector of Locally Aggregated Descriptors (VLAD) onto an extensive space of curved Riemannian manifolds. We provide a comprehensive mathematical framework that formulates the aggregation problem of such manifold data into an elegant solution. In particular, we consider structured descriptors from visual data, namely Region Covariance Descriptors and linear subspaces that reside on the manifold of Symmetric Positive Definite matrices and the Grassmannian manifolds, respectively. Through rigorous experimental validation, we demonstrate the superior performance of this novel Riemannian VLAD descriptor on several visual classification tasks including video-based face recognition, dynamic scene recognition, and head pose classification.

1. Introduction

This paper extends the Vector of Locally Aggregated Descriptors (VLAD) [27] to general class of curved Riemannian manifolds, hence adding a novel dimension to the applicability of VLAD in tackling fundamental recognition problems in computer vision.

Our motivation stems from the fact that, in $\mathbb{R}^n$, coding local image or video descriptors using VLAD has been shown to be exceptionally successful in addressing a variety of challenging problems, such as image retrieval [27, 17], texture classification [9], and scene recognition [17]. The advantage of VLAD can be even more appreciated by noting that its high discriminatory power is achieved using rudimentary vector subtraction and addition operations, a negligible computational cost compared to more involved approaches like deep convolutional networks.

To put the discussion into perspective, describing images or videos by local descriptors is preferable to holistic representations when, for instance, the recognition problem pertains large intra-class variations, articulated shapes, self-occlusions, and changing backgrounds, to name a few. However, almost all previous studies [27, 41, 50, 17, 9] that extracted local descriptors in the form of vectors disregarded the underlying intrinsic manifold structures, probably because proven and general techniques to aggregate non-vectorized structures are scarce.

On a related note, structured representations such as Region Covariance Descriptors (RCovD) and linear subspaces have been shown to provide robust and efficient representations for a wide range of tasks [22, 11, 51, 12, 46, 48, 49]. Therefore, a curious mind might inquire

- can we extract structured descriptors from visual data and then aggregate them in a fashion similar to VLAD to obtain more discriminating representations?
- is there any mathematical framework that helps us formulate the aggregation problem into an elegant yet general and accurate solution?

This paper provides answers to the aforementioned questions by introducing the Riemannian version of the conventional VLAD, called R-VLAD, a new coding approach that enables fusing local descriptors on curved spaces.

More specifically, we present a universal framework for constructing a rich representation out of local image or video descriptors where each local descriptor is indeed a point on a Riemannian manifold. The conventional VLAD algorithm can hence be considered as a special case of our Riemannian formulation when the manifold is chosen to be the Euclidean space.

We then turn our attention to two widely utilized Riemannian manifolds in vision, namely the manifold of Symmetric Positive Definite matrices (SPD) and the manifold of linear subspaces, known as the Grassmannian manifold. To this end, we develop the sister family of R-VLAD by exploiting other forms of metrics defined on SPD matrices and Grassmannian manifolds. In particular, we make use of the Stein [43] and Jeffrey [52] divergences on SPD matrices and...
the projection distance [20] on Grassmannian manifolds to yield computationally more efficient versions of R-VLAD.

Our experiments demonstrate the superiority of the proposed R-VLAD descriptor against several baseline and state-of-the-art methods such as the Weighted ARay of COVariances (WARCO) of Tosato et al. [46] for head pose classification, the Adaptive Deep Network Template (ADNT) of Hayat et al. [23] for video-based face classification, and the Bag of Spatiotemporal Energy (BoSE) of Feichtenhofer et al. [13] for dynamic scene recognition.

To the best of our knowledge, using the standard protocol, the proposed R-VLAD achieves top results on standard benchmarks: 85% for HOCCofe [46], 79.1% for HDM05 [32], 79.9% for YouTube Celebrities [28], 97.6% for Dytex++ [14], and 99.8% for UPENN [10].

2. Related Work

Since the late nineties, Bag of Words (BoW) [40, 18, 30, 42] and its extensions [29, 19] have been the de facto driving force for image and video representation. Two notable examples are Video Google [42] for object matching in videos and Spatio-temporal Pyramid Matching (SPM) [29] for scene classification.

Broadly speaking, the local models such as BoW, Fisher Vectors (FV) [37], and VLAD [27] can benefit from powerful local feature descriptors (e.g., SIFT [31]), which to some extent provide robustness to transformations such as scaling, translation, and occlusion. Furthermore, the resulting vector can be compared using the Euclidean distance norms and utilized [36] in conventional classifiers (e.g., Support Vector Machines (SVM)).

The VLAD descriptor, our main focus in this work, is a coding scheme similar in spirit to the earlier FV [37]. It is able to provide compact codes by capturing certain aspects of the distribution of features. While inheriting the useful properties of BoW, VLAD departs from it as it encodes the differences from the cluster centers rather than counting the number of assignments to them.

Several recent studies [26, 7, 2] suggest that promising performances on various benchmarks can be attained by effective use of the original VLAD descriptor. For example, Gong et al. proposed to exploit VLAD to pool the activations of Deep Convolutional Neural Networks. Impressively, the results of multi-scale VLAD reported in [17] outperformed various state-of-the-art methods. One example is the scene classification on the MIT dataset, where the multi-scale VLAD with only 4096 features comfortably outperformed the mixture of FV and bag-of-parts, which used 221,550 features for its decision [17]. Another example is the work of Cimpoi et al. who showed that the VLAD descriptor is preferable to others for the task of texture recognition [9]. Some other recent advances include better normalization schemes for the VLAD descriptor [2], supervised codebook learning for VLAD [34], VLAD with higher order statistics [34], and VLAD for action recognition [25].

3. Riemannian VLAD

In this section, we derive a general formulation for Riemannian VLAD. In doing so, we start by studying the conventional VLAD formulation, and in particular through its predecessor, the Fisher Vector (FV) [37].

3.1. Conventional VLAD in Euclidean Spaces

Given a set of local descriptors $X = \{x_t\}_{t=1}^m$, $x_t \in \mathbb{R}^d$ extracted from an image or video, let us assume that $X$ admits a probability density function in the form of a Gaussian Mixture Model (GMM)

$$p(x_t|\lambda) = \sum_{i=1}^K \omega_i N(x_t|\mu_i, \Sigma_i),$$

with $\lambda = \{\omega_i, \mu_i, \Sigma_i\}$ being the mixture weight, mean and covariance of the Gaussian components, respectively.

In the FV method, $X$ is encoded through its score function\(^1\). Related to VLAD is the gradient part with respect to $\mu$, which has the following form

$$\nabla_{\mu_i} \log p(X|\lambda) = \sum_{t=1}^m \gamma_t(x_t) \Sigma_i^{-1} (\mu_i - x_t), \quad (1)$$

where $\gamma_t(x_t)$ is the soft-assignment of $x_t$ to the i-th Gaussian component, i.e.,

$$\gamma_t(x_t) = \frac{\omega_i N(x_t|\mu_i, \Sigma_i)}{\sum_{j=1}^K \omega_j N(x_t|\mu_j, \Sigma_j)}.$$

In VLAD, the input space $\mathbb{R}^d$ is partitioned into $K$ Voronoi cells by means of a codebook $C$ with centers $\{c_i\}_{i=1}^K$, $c_i \in \mathbb{R}^d$. For the aforementioned query set $X$, the VLAD code $V \in \mathbb{R}^{Kd}$ is obtained by concatenating $K$ Local Difference Vectors (LDV) $v_i$ storing the differences $c_i - x_t$ in each cell, i.e.,

$$v_i = \sum_{x_t \in c_i} c_i - x_t, \quad (2)$$

where $x \in c_i$ means that the local descriptor $x$ belongs to the Voronoi defined by $c_i$, i.e., the closest codeword to $x$ is $c_i$. Direct comparison between Eq. (1) and Eq. (2) reveals the following about VLAD

1. In contrast to FV which uses soft assignment of local descriptors to the Gaussians, VLAD exploits a hard assignment scheme.

2. Unlike FV, the covariance matrices of the mixture components with VLAD are assumed to be diagonal and fixed, i.e., $\Sigma_i = \sigma I_d$, $\forall i \in \{1, 2, \cdots , K\}$.\(^1\)

---

\(^1\)In statistics, the score function is the gradient of the log-likelihood of the data on the model.
The take-home message here is that the LDV is indeed the gradient of the Euclidean distance\footnote{To be more precise, the gradient of a Gaussian function where the associated normalization terms are discarded.} and the VLAD framework is a clever simplification of the FV algorithm. With this introduction, we are now ready to introduce Riemannian VLAD (R-VLAD).

\section*{3.2. Extension to Riemannian Manifolds}

Now let us assume that $X = \{x_t\}_{t=1}^{m}$, $x_t \in \mathcal{M}$ and $C = \{c_k\}_{k=1}^{K}$, $c_k \in \mathcal{M}$, are a set of local descriptors (extracted from a query image or video) and codewords on a Riemannian manifold $\mathcal{M}$, respectively. The R-VLAD descriptor on $\mathcal{M}$ is obtained once we have the following tools at our disposal

- a metric $\delta(x, y) : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$ required to determine how the local descriptors should be assigned to the codewords.
- operators to perform the role of vector addition or subtraction on $\mathcal{M}$.

Since a Riemannian manifold is a metric space, one could seamlessly use the geodesic distance $\delta_g : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$ to address the first requirement. As for the second requirement, we note that on a Riemannian manifold, one can see a vector $\overrightarrow{ab}$ (attached at point $a$) as a vector of the tangent space at $a$, i.e. $T_a \mathcal{M}$. Therefore, subtraction on a Riemannian manifold can be attained through the logarithm map, $\log_a : \mathcal{M} \rightarrow T_a \mathcal{M}$.\footnote{Due to the lack of space, we skip rigorous definition of tangent vectors, logarithm map and other concepts of Riemannian geometry. The interested reader is referred to textbooks on Riemannian geometry for a formal treatment.} This concept has been used widely in the literature. For example, vector subtraction through the logarithm map was used to address the problem of interpolation and filtering \cite{35}, sparse coding \cite{24}, and dimensionality reduction \cite{15}, to name a few.

The aforementioned discussion hints towards devising the R-VLAD as follows

- exploit the geodesic distance to determine the closest local descriptors to each codeword.
- build a Riemannian LDV per codeword using the tangent space attached to each codeword on the manifold.

Since the pole of the tangent space, i.e., $c_t$ is fixed, the outputs of the logarithm map are compatible with each other and no further special care (e.g., parallel transport) is required\footnote{To be precise, this argument is valid as long as $x_t$ is not in the cut locus of $c_t$. This is of course not a very restricting assumption as in many manifolds (e.g., the SPD manifold) the cut locus is indeed empty.}. Therefore, Eq. (2) on a curved Riemannian manifold boils down to

$$v_t = \sum_{x_i \in c_t} \log_{c_t}(x_t). \quad (3)$$

While being perfectly valid, for reasons that become clear later, we are interested in having a more general formulation for R-VLAD. More specifically, rather than having a method that only works with the geodesic distance, we would like to extend our formulation such that any metric on $\mathcal{M}$ can be used.

Obviously, for a new metric $\delta : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$, we need to only take care of the second requirement. Since the LDV can be understood as the gradient of the distance function in the Euclidean case (see § 3.1), it is tempting to define the LDV on $\mathcal{M}$ as $\nabla_{c_t} \delta^2(c_t, x_t)$\footnote{On an abstract Riemannian manifold $\mathcal{M}$, the gradient of a smooth real function $f$ at a point $x \in \mathcal{M}$, denoted by $\nabla_x f$, is the element of $T_x \mathcal{M}$ satisfying $\langle \nabla_x f(x), \zeta \rangle_x = Df_x[\zeta]$ for all $\zeta \in T_x \mathcal{M}$, where $Df_x[\zeta]$ denotes the directional derivative of $f$ at $x$ in the direction of $\zeta$.}. The following theorem reinforces this idea even more.

\textbf{Theorem 1.} For a Riemannian manifold $\mathcal{M}$, the gradient of the geodesic distance function, $\delta_g : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$ is given by

$$\nabla_x \delta^2(x, y) = -2 \log_a(y). \quad (4)$$

\textbf{Proof.} The interested reader is referred to \cite{45} for the proof of this theorem. $\square$

Unfortunately, choosing $\nabla_{c_t} \delta^2(c_t, x_t)$ for LDV will not work in practice. The main reason being that for $\delta_g$, the norm of $\nabla_x \delta^2(x, y)$ is related directly to the metric, i.e.,

$$\| \nabla_x \delta^2(x, y) \|^2 = 4 \| \log_a(y) \|^2 = 4 \delta^2_0(x, y).$$

This is of course inherited to the Euclidean space when the metric is chosen to be the geodesic distance, i.e., the Euclidean distance. However, this will not generalize to other metrics as shown by the following example.

\textbf{Example 1.} Fig. 1 shows the behavior of $\nabla_X \delta^2(X, Y)$ by varying $\delta^2(X, Y)$ for the projection metric on the Grassmann manifold $G^2_3$ (see § 5 for the equations). Interestingly, the norm of the gradient will start decreasing while point $Y$ gets farther away from $X$. This means, during encoding, a point which should contribute significantly to the output, can act as an insignificant point, hence deteriorating the discriminatory power of the descriptor.

The aforementioned example provides us with the following guideline for constructing an LDV on $\mathcal{M}$.

- the length of the LDV should represent the metric considered on $\mathcal{M}$.

As such, we propose the following form of LDV for our general R-VLAD descriptor (see Algorithm 1 for a step-by-step on the R-VLAD technique).

$$v_t = \sum_{x_i \in c_t} \delta(c_t, x_t) \frac{\nabla_{c_t} \delta^2(c_t, x_t)}{\| \nabla_{c_t} \delta^2(c_t, x_t) \|}. \quad (5)$$

4953
Algorithm 1 The proposed R-VLAD algorithm

Input:
- local descriptors $\mathcal{X} = \{x_i\}_{i=1}^m$, $x_i \in \mathcal{M}$, extracted from a query image or video,
- codebook $\mathcal{C} = \{c_k\}_{k=1}^K$, $c_k \in \mathcal{M}$

Output:
- $V(\mathcal{X})$ the Riemannian VLAD representation of $\mathcal{X}$

1: for $i = 1 \rightarrow k$ do
2: Find $x_i \in c_j$, all nearest query points from $\mathcal{X}$ to $c_j$
3: Compute $v_i$, $i$-th Local Difference Vector (LDV), using Eq. (5)
4: end for
5: Concatenate the resulting LDVs to form the final descriptor, i.e.,
$V(\mathcal{X}) = [v_1^T, v_2^T, \ldots, v_k^T]^T$

Remark 1. In line with the recommendations in [38], post-processing of VLAD codes could increase the discriminative power of the codes. In practice, we normalize the R-VLAD codes in two steps. First, an element-wise power normalization is performed using the transfer function $y : \mathbb{R} \rightarrow \mathbb{R}$, $y(x) = \text{sign}(x) \sqrt{|x|}$, where $x$ is the element of VLAD vector and $| \cdot |$ denotes absolute value. This is to avoid having a concentrated distribution around zero. The power normalization is followed by an $\ell_2$ normalization to make the energy of descriptors uniform.

In the following sections, we develop the R-VLAD for two widely used manifolds in vision, i.e., the SPD and the Grassmannian manifolds (see Table 1 for a quick peak at the studied metrics and the associated gradients as required by Eq. (5)). Before concluding this section and for the sake of completeness, we discuss how a Riemannian codebook can be learned from training samples.

3.3. k-Means on Riemannian Manifolds

Given an abstract manifold $\mathcal{M}$ with an associated metric $\delta : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$, we train a codebook similar to standard k-means using an EM-based approach. The algorithm starts by selecting $k$ points from the training data randomly as the cluster centers. In the E-step, each of the points are assigned to the nearest cluster center using $\delta$. Then in the M-step, the cluster centers are re-computed using the Fréchet mean.

Definition 1. The Fréchet mean for a set of points $\{x_i\}_{i=1}^m$, $x_i \in \mathcal{M}$ is local minimizer of the cost function
$$c^* = \arg \min_c \sum_{i=1}^m \delta^2(c, x_i).$$

In general, an analytic solution for Eq. (6) cannot be sought and iterative schemes that exploit the logarithm and exponential maps must be employed [35]. This, for a high-dimensional manifold with a big $m$, could easily become overwhelming. One reason in generalizing the R-VLAD to work with an arbitrary metric comes from the fact that, unlike the general case, for some metrics Eq. (6) has an analytic solution. We will provide more details on how Eq. (6) can be solved for the special cases of interest in this paper later.

4. R-VLAD on SPD Manifold

The space of $d \times d$ SPD matrices is mostly studied when endowed with a Riemannian metric and thus forming a Riemannian manifold [35]. The Affine Invariant Riemannian Metric (AIRM) is probably the most popular Riemannian structure for analyzing SPD matrices [35].

Definition 2. The geodesic distance $\delta_g : S_{++}^d \times S_{++}^d \rightarrow [0, \infty)$ induced by the AIRM is defined as
$$\delta_g(X, Y) = \|\log(X^{-1/2}YX^{-1/2})\|_F,$$ (7)
where $\log(\cdot)$ is the matrix principal logarithm.

Beside the AIRM, two types of Bregman divergences, namely the Jeffrey [52] and Stein [43] are widely used to measure similarities on SPD matrices.

Definition 3. The $J$ divergence (also known as Jeffrey or symmetric KL divergence) $\delta_j : S_{++}^d \times S_{++}^d \rightarrow [0, \infty)$ is a symmetric type of Bregman divergence and is defined as
$$\delta_j^2(X, Y) \triangleq \frac{1}{2} \text{Tr}(X^{-1}Y) + \frac{1}{2} \text{Tr}(Y^{-1}X) - d.$$ (8)

Definition 4. The Stein metric $\delta_S : S_{++}^d \times S_{++}^d \rightarrow [0, \infty)$ is also a symmetric type of Bregman divergence and is defined as
$$\delta_S^2(X, Y) \triangleq \ln \det \left( \frac{X + Y}{2} \right) - \frac{1}{2} \ln \det (XY).$$ (9)

As for the $\delta_g$, the Fréchet mean is obtained using an iterative approach [35]. The same holds for the Stein metric as a result of the following theorem.

Theorem 2. The Fréchet mean of a set of SPD matrices $\{X_i\}_{i=1}^m \in S_{++}^d$ with $\delta_S$ is obtained iteratively via
$$\mu(t+1) = \left[ \frac{1}{m} \sum_{i=1}^m \left( X_i + \mu(t) \right)^{-1} \right]^{-1}.$$ (10)
Proof. See [8] for the proof.

However, with the Jeffrey divergence, we have the luxury of obtaining the Fréchet mean analytically.

Theorem 3. The Fréchet mean of a set of SPD matrices \( \{X_i\}_{i=1}^m \in S^d_{++} \) with \( \delta \) is

\[
\mu = \frac{1}{m} \sum_{i=1}^m X_i - \frac{1}{m} \sum_{i=1}^m X_i, \tag{11}
\]

where \( P = \sum_i X_i^{-1} \) and \( Q = \sum_i X_i \).

Proof. The solution is obtained by zeroing out the derivative of \( \sum_i \delta^2_i(X_i, \mu) \) with respect to \( \mu \). The proof is relegated to the supplementary material.

The gradient of a function \( f : S^d_{++} \rightarrow \mathbb{R} \) at \( X \) has the following form on \( S^d_{++} \) [44]

\[
\nabla_X f = X \text{sym}(Df)X, \tag{12}
\]

where \( \text{sym}(X) = 0.5(X + X^T) \) and \( Df \) is the derivative of the function \( f : \mathbb{R}^{d \times d} \rightarrow \mathbb{R} \) with respect to \( X \).

The derivatives of \( D\delta^2_\Sigma \) and \( D\delta^2_\delta \) are reported in [8]. From [8] we can deduce the gradients required in the R-VLAD algorithm as depicted in Table I.

### Computational Cost

The computational loads of computing \( \delta^2_\Sigma \), \( \delta^2_\delta \) and \( \delta^2_\Sigma \) are \( 4d^3 \), \( 8/3d^3 \) and \( d^3 \), respectively [8]. Computing the gradient of \( \delta^2_\Sigma \) requires an eigenvalue decomposition (for computing principal matrix logarithm) which adds up to a total of \( 9d^3 \) flops for \( \delta^2_\Sigma \) (considering the matrix multiplications).

For \( \delta^2_\delta \) and \( \delta^2_\Sigma \), computing gradient just requires a matrix inversion which is \( O(d^3) \). As such, the computational load of R-VLAD using \( \delta^2_\Sigma \) and \( \delta^2_\Sigma \) is \( O(17/3d^3) \) and \( O(4d^3) \), respectively.

### 5. R-VLAD on Grassmannian

The space of \( p \)-dimensional linear subspaces of \( \mathbb{R}^d \) for \( 0 < p < d \) is not a Euclidean space, but a Riemannian manifold known as the Grassmannian \( G^d_p \) [1]. A point on the Grassmannian \( G^d_p \) may be represented by an arbitrary \( d \times p \) matrix with orthogonal columns, i.e., \( X \in G^d_p \Rightarrow X^TX = I_p \). For the Grassmannian, the geodesic distance between two points \( X \) and \( Y \) is given by

\[
\delta_g(X, Y) = \|\Theta\|_2, \tag{13}
\]

where \( \Theta \) is the vector of principal angles between \( X \) and \( Y \) [1].

In addition to the geodesic distance, a popular metric on \( G^d_p \) is the projection metric \( \delta_P : G^d_p \times G^d_p \rightarrow \mathbb{R}^+ \) defined as [22, 20]

\[
\delta_P^2(X, Y) = \|XX^T - YY^T\|_F^2, \tag{14}
\]

where \( \| \cdot \|_F \) denotes the Frobenius norm. Unlike \( \delta_g \) which does not have an analytic form for the Fréchet mean (see [47] for more details), the projection metric has the following interesting property.

Theorem 4 (Closed-Form Mean). The Fréchet mean for a set of points \( \{X_i\}_{i=1}^m \) is \( X_i \in G^d_p \) based on \( \delta_P \) admits a closed-form solution. That is the \( p \) largest eigenvectors of \( \sum_{i=1}^m X_iX_i^T \).

Proof. The solution is obtained by maximizing \( \text{Tr}(\mu^T \{ \sum_{i=1}^m X_iX_i^T \} \mu) \) and considering the orthogonality constraint \( \mu^T \mu = I_p \). The proof is relegated to the supplementary material due to the lack of space.

The gradient of a function on Grassmannian, i.e., \( f : G^d_p \rightarrow \mathbb{R} \) has the form

\[
\nabla_X f = (I_d - XX^T)Df, \tag{15}
\]

where \( Df \) is a \( d \times p \) matrix of partial derivatives of \( f \) with respect to the elements of \( X \), i.e.,

\[
Df_{i,j} = \frac{\partial f}{\partial X_{i,j}}.
\]

The logarithm map (and also the exponential map) on Grassmannian does not have an analytic form. However, numerical methods for computing both mappings do exist. In particular, we will use the formulation introduced in [5] to compute R-VLAD using the geodesic distance.

Note that in Table 3 of [8] a scalar factor of 0.5 is wrongly dropped from the Jeffrey divergence (KLDM according to [8]). Also please note that the gradient reported in [8] is the Euclidean gradient not the Riemannian as required here.

### Table I: Metrics and associated gradients on the SPD and Grassmannian manifold.

<table>
<thead>
<tr>
<th>Manifold</th>
<th>Metric</th>
<th>( \delta^2(X, Y) )</th>
<th>( \nabla_X \delta^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S^d_{++} )</td>
<td>geodesic</td>
<td>( | \log(X^{-1/2}YX^{-1/2}) |_F^2 )</td>
<td>( 2X^{-1/2} \log(X^{-1/2}YX^{-1/2})X^{1/2} )</td>
</tr>
<tr>
<td>( S^d_{++} )</td>
<td>Stein</td>
<td>( \ln \det \left( \frac{X+Y}{2} \right) - \frac{1}{2} \ln \det(XY) )</td>
<td>( X(X+Y)^{-1}X - \frac{1}{2}X )</td>
</tr>
<tr>
<td>( S^d_{++} )</td>
<td>Jeffrey</td>
<td>( \frac{1}{2} \text{Tr}(X^{-1}Y) + \frac{1}{2} \text{Tr}(Y^{-1}X) - d )</td>
<td>( \frac{1}{2}X(Y^{-1} - X^{-1}YX^{-1})X )</td>
</tr>
<tr>
<td>( G^d_p )</td>
<td>geodesic</td>
<td>( | \Theta |_2 )</td>
<td>No analytic form</td>
</tr>
<tr>
<td>( G^d_p )</td>
<td>projection</td>
<td>( 2p - 2 | X^TY |_2^2 )</td>
<td>( -4(I_d - XX^T)YY^TX )</td>
</tr>
</tbody>
</table>

---

\[ \text{Table 1: Metrics and associated gradients on the SPD and Grassmannian manifold.} \]
As for the projection metric, using Eq. (15) and noting that
\[ \delta_p^2(X, Y) = 2p - 2 \| X^T Y \|_F^2 \]
leads to the following analytic form for the gradient as required in Eq. (5)
\[ \nabla_X \delta_p^2(X, Y) = -4 \left( I_d - XX^T \right) YY^T X. \]  
(16)

Computational Cost
The computational load of coding in R-VLAD is dictated by the complexity of the used metric \( \delta^2 \) and its gradient. On top of this, one should pay attention to the complexity of Riemannian \( k \)-means. As long as the complexity of coding is considered, we note that \( \delta_g^2 \) on Grassmannian is obtained through SVD decomposition. As such, computing \( \delta_g^2 \) requires \( dp^2 + p^3 \) flops on \( G_{d,p}^{G} \). In contrast, the complexity of computing \( \delta_p^2 \) on \( G_{d,p}^{G} \) is \( dp^2 \).

Computing the gradient of \( \delta_g^2 \) (or logarithm map) using a very efficient implementation requires a matrix inversion of size \( p \times p \), two matrix multiplications of size \( d \times p \), and a thin SVD of size \( d \times p \). Computing thin SVD using a stable algorithm like the Golub-Reinsch [16] requires \( 14dp^2 + 8p^3 \) flops. This adds up to a total of \( O(10p^3 + 17dp^2) \) flops for one local descriptor. As for \( \delta_p^2 \), computing the gradient according to the Table 1 demands for \( 4dp^2 \) operations. This results in a total of \( 5dp^2 \) flops for the projection metric.

To give the reader a better sense on the computational complexity of R-VLAD using \( \delta_g^2 \) and \( \delta_p^2 \), we measured the coding time for 1000 videos each with its own set of local descriptors on \( G_{177}^{177} \) (this is an example of the Grassmannian we will use in our experiments later). On a quad-core machine using Matlab, coding time for \( \delta_p^2 \) and \( \delta_g^2 \) were observed to be around 155 and 440 seconds, respectively.

6. Experiments
This section presents comparative evaluation results of our proposal against the baseline and state-of-the-art for a number of visual recognition problems defined on the SPD and Grassmannian manifold. In all our experiments, a set of overlapping blocks or cubes are extracted from images or videos. Each block or cube is then represented by an RCovD or a linear subspace, hence it corresponds to a point on the SPD or the Grassmannian manifold, respectively. Different algorithms tested in our experiments are

BoW\(_G\): Riemannian BoW model using geodesic distance. An image or video is described by the histogram of its building blocks or cubes using geodesic distance. The codebook is learned by Riemannian \( k \)-means algorithm as described in § 3.3.

BoW\(_{LE}\): BoW model trained by flattening the manifold through a fixed tangent space. We follow the terminology introduced in [3] and label this as Log-Euclidean BoW or for short BoW\(_{LE}\).

VLAD\(_{LE}\): Similar in concept to BoW\(_{LE}\) but instead of BoW, we assess the performance of BoW\(_{LE}\) by flattening the manifold through a fixed tangent space.

R-VLAD\(_G\): R-VLAD using geodesic distance.

R-VLAD\(_{I/S/P}\): R-VLAD using the Jeffrey, Stein, or projection metrics.

Besides the Log-Euclidean and BoW\(_G\) methods that serve as baseline methods, we will exclusively consider previous state-of-the-art algorithms for each studied problem to demonstrate the power of R-VLAD.

For classification, the VLAD descriptors, either Log-Euclidean or R-VLAD, are fed to a Nearest Neighbor (NN) or a linear Support Vector Machine (SVM) [6]. We separately report the performances for these two classifiers. In all the experiments, the size of codebook for VLAD is set to 16. This value is obtained empirically as a good compromise between computational cost and classification accuracy. For the BoW\(_G\), we will report the best accuracy by searching over various codebook sizes.

6.1. SPD Manifold
For tests on the SPD manifold, an image or video is described by a set of RCovDs. More specifically, given a block \( I(x, y) \) of size \( W \times H \), let \( \mathcal{O} = \{ \omega_i \}_{i=1}^r \), \( \omega_i \in \mathbb{R}^d \) be a set of \( r \) observations extracted from \( I(x, y) \), e.g., \( \omega_i \) concatenates intensity values, gradients, filter responses, etc. for image pixel \( i \). Then, block \( I \) can be represented by the \( d \times d \) RCovD using
\[ C_I = \frac{1}{r-1} \sum_{i=1}^r (\omega_i - \overline{\omega})(\omega_i - \overline{\omega})^T, \]  
where \( \overline{\omega} = \frac{1}{r} \sum_{i=1}^r \omega_i \).

Head Pose Classification:
As our first experiment, we study the problem of head pose (orientation) classification. To this end, we utilized the Heads Of Coffee break (HOCoffee) dataset [46] which presents outdoor images captured by a head detector for the purpose of automatically detecting social interactions. This dataset is composed of 18,117 low-resolution images of size \( 50 \times 50 \) pixels. The dataset comes with a predefined test protocol in which 9522 images are considered as training data and the remaining 8595 images are used for evaluation.

Similar to [46], we used a Difference Of Offset Gaussian (DOOG) filter-bank along color and image gradients to describe each image. More specifically, the feature vector assigned to each pixel in the image is
\[ \omega_{x,y} = \left[ I_1(x, y), I_4(x, y), I_6(x, y), \sqrt{I_x^2 + I_y^2}, \arctan \left( \frac{I_x}{I_y} \right), G_1(x, y), G_2(x, y), \cdots, G_8(x, y) \right], \]

\( ^8 \) A similar discussion holds for cubes extracted from videos.
Table 2: Recognition accuracies for the HOCoffee [46] and HDM05 [32] datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>HOCoffee</th>
<th>HDM05</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Best</td>
<td>80.8 [46]</td>
<td>73.3 ± 11.4 [21]</td>
</tr>
<tr>
<td>BOW(_{G})-NN</td>
<td>81.8</td>
<td>65.3 ± 14.1</td>
</tr>
<tr>
<td>BOW(_{G})-SVM</td>
<td>80.0</td>
<td>71.6 ± 7.7</td>
</tr>
<tr>
<td>BOW(_{LL})-NN</td>
<td>81.6</td>
<td>50.4 ± 11.2</td>
</tr>
<tr>
<td>BOW(_{LE})-SVM</td>
<td>78.9</td>
<td>65.9 ± 8.5</td>
</tr>
<tr>
<td>VLAD(_{LE})-NN</td>
<td>82.4</td>
<td>65.4 ± 13.5</td>
</tr>
<tr>
<td>VLAD(_{LE})-SVM</td>
<td>82.4</td>
<td>71.2 ± 10.6</td>
</tr>
<tr>
<td>R-VLAD(_{G})-NN</td>
<td><strong>85.0</strong></td>
<td>78.1 ± 5.8</td>
</tr>
<tr>
<td>R-VLAD(_{G})-SVM</td>
<td>84.3</td>
<td><strong>79.1 ± 7.5</strong></td>
</tr>
<tr>
<td>R-VLAD(_{S})-NN</td>
<td>84.9</td>
<td>72.5 ± 10.1</td>
</tr>
<tr>
<td>R-VLAD(_{S})-SVM</td>
<td>84.2</td>
<td>74.1 ± 5.2</td>
</tr>
<tr>
<td>R-VLAD(_{J})-NN</td>
<td>84.5</td>
<td>74.3 ± 8.2</td>
</tr>
<tr>
<td>R-VLAD(_{J})-SVM</td>
<td>83.3</td>
<td>76.5 ± 11.6</td>
</tr>
</tbody>
</table>

where \( I_c(x,y) \), \( c \in \{ L, a, b \} \), denotes the CIELab color information at position \((x,y)\), \( I_x \) and \( I_y \) are luminance derivatives, and \( G(x,y) \) denotes the response of the \( i \)-th DOOG centered at \( I_L(x,y) \). Therefore, each local covariance descriptor is on \( S_+^{13} \).

In the second column of Table 2, we report the recognition accuracies of all the studied methods for this dataset. Several conclusion can be drawn here. First of all, the local approach, even the simple BOW\(_{G}\)-NN, outperforms the previous state-of-the-art method. The R-VLAD technique with all studied metrics achieve the best performances (with a NN classifier), with R-VLAD\(_{G}\) being the overall winner in terms of accuracy. However, the performance of R-VLAD with the Stein and Jeffrey is on par or slightly worse than that of the geodesic solution while being at least 27 times faster in coding and 65 times faster (especially for the case of Jeffrey) in the training phase. We also observe that the proposed R-VLAD method is significantly superior as compared to the Log-Euclidean methods, which suggests that the underlying Riemannian structure is better exploited in R-VLAD.

**Action Recognition from Motion Capture Data:**
As our second experiment, we tackled the task of human action recognition from the skeletal information. To this end, we used HDM05 dataset [32], which contains 14 different human actions performed by 5 subjects.

We followed the protocol of [21] in which only the real-world location of 4 joints related to arms and legs are utilized for generating RCovDs. Therefore, RCovDs of size \( 12 \times 12 \), extracted with a temporal overlap of 75% were used as local descriptors for an action. We used a leave-one-subject-out protocol, where the data of 4 subjects are used for training and the remaining one is used for testing.

The average recognition accuracies along the standard deviations are reported in the last column of Table 2. From this table we conclude that the R-VLAD equipped with a linear SVM classifier outperforms the state-of-the-art [21] regardless of its metric. The maximum accuracy is obtained by R-VLAD\(_{G}\) which is nearly 6 percentage points better than [21]. Furthermore, the R-VLAD\(_{G}\) is significantly better than R-VLAD\(_{S}\) and R-VLAD\(_{J}\) which makes it the preferred technique if coding time is not an issue.

Moreover, we evaluated the performance of the VLAD in Euclidean space (VLAD\(_{E}\)) using very small to large codebook sizes to obtain signatures with the dimensionality similar or greater than that of R-VLAD’s signatures. We observed that R-VLAD is significantly superior to VLAD\(_{E}\). For instance, the best accuracy of VLAD\(_{E}\) on the HOCoffee and HDM05 datasets are 79.9% and 69.4%, respectively.

**6.2. Grassmannian Manifold**

The experiments on the Grassmannian manifolds are designed to tackle the problem of recognition from videos or image-sets. In this context, a video or an image-set is divided into 3D blocks, followed by describing each block by a linear subspace through SVD decomposition.

**Face Recognition:**
As our first experiment on Grassmannian manifolds, we tackled the task of video-based face recognition. To this end, we considered the YouTube Celebrity dataset [28] which contains 1910 videos of 47 people (see Fig. 2 for examples). The large diversity of poses, illumination, and facial expressions in addition to high compression ratio of face images provide a significant challenges in this dataset.

For our evaluation, we followed the setup used in the method of Adaptive Deep Network Template (ADNT) by Hayat et al. [23]. More specifically, from each video, the face regions are extracted using the tracker of Ross et al. [39]. Then, without any further refinement, each face region was divided into \( 4 \times 4 \) distinct non-overlapping blocks and the histogram of Local Binary Patterns (LBP) [33] was extracted for each patch and concatenated to form the final frame descriptors.

We note that various evaluation protocols were used by researchers on this dataset. Here, we use the five-fold cross validation protocol introduced in [23], which divides the whole dataset equally (with minimum overlap) into five folds with 9 videos per subject in each fold. Three of the videos were randomly selected for training, while the remaining six were used for testing.

We generated linear subspaces of order 6 by grouping features of every 6 consecutive frames. Therefore, each local descriptor belongs to \( S_+^{60} \). The second column of Table 3 summarizes the average recognition rates and the standard deviations of all the studied methods.

We note that the R-VLAD with both geodesic and projection metric comfortably outperforms the state-of-the-art
ADNT algorithm. The maximum accuracy of 79.9% is achieved by R-VLAD using projection metric with a linear SVM classifier.

**Dynamic Texture Classification:**
We performed an experiment to classify videos of dynamic textures using the DynTex++ dataset [14]. Dynamic textures are videos of moving scenes (smoke, waves, high way, forest fire, etc.) that exhibit certain stationarity properties in time domain. The DynTex++ dataset contains 3600 $(50 \times 50 \times 50)$ videos of moving scenes in 36 classes.

To obtain local Grassmannian descriptors, each video is decomposed into 3D blocks of size $15 \times 15 \times 15$ with spatial (temporal) overlap of 5 pixels (frames). The 3D block was then described by grouping its internal frames and describing each with the 3D extension of the Local Binary Pattern (LBP) [33], namely LBP Three Orthogonal Planes (LBPTOP) [53]. For each 3D block and from the LBPTOP features, we extracted a subspace of dimension 6 using SVD. This resulted in having local descriptors on $\mathbb{G}_6^{177}$. In total, we extracted 512 subspaces from each video.

For this experiment, we followed the evaluation protocol used in [4]. That is, half of the videos of each class were randomly chosen as training data and the remaining ones were used as test data. The process of random selection was repeated 10 times and the average accuracy along standard deviations are reported in the third column of Table 3.

Table 3 shows that the proposed R-VLAD outperforms the state-of-the-art by more than 5 percentage points. R-VLAD with projection metric equipped with a linear SVM classifier achieves the highest recognition rate of 97.6%. Compared to Log-Euclidean solutions, again R-VLAD is preferable though the gap is not as big as that of the previous experiment.

**Dynamic Scene Recognition:**
As the last experiment, we considered the task of scene recognition from the videos using UPenn dataset [10] (see Fig. 2 for example classes). The UPenn dataset consists of 420 videos of natural scenes spanning 14 categories. The videos are obtained from various sources including personal footage captures and online repositories such as YouTube. Moreover, significant differences in image resolution and appearance, frame rate, scale, viewpoint, and illumination conditions exist in this dataset.

To extract local Grassmannian points, we followed the setup used in the previous experiment and represented each video using 6 dimensional subspaces extracted from $15 \times 15 \times 15$ subblocks (using LBPTOP features). Leave-one-video-out scenario is the standard evaluation protocol on this dataset. To the best of our knowledge, the recent Bag of Spatiotemporal Energy (BoSE) method of Feichtenhofer et al. [13] has achieved the highest accuracy on this dataset.

The recognition accuracies for all the studied methods are shown in the last column of Table 3. The results are self-explanatory. The R-VLAD equipped with a linear SVM classifier achieves the highest accuracy, outperforming the state-of-the-art by more than 3 percentage points. Notably, the Log-Euclidean VLAD is performing slightly better than the state-of-the-art method of Feichtenhofer et al. [13].

**7. Main Findings and Future Directions**

Inspired by the recent success of compact descriptors in Euclidean spaces and superior discriminative power of the descriptors on Riemannian manifolds, in this paper we proposed R-VLAD, the Riemannian version of the conventional *Vector of Locally Aggregated Descriptors* (VLAD). In addition to the comprehensive formulation, we devised manifold-specific versions of the R-VLAD on the SPD and Grassmannian. An extensive set of successful experiments on several challenging vision tasks including action recognition from MoCaP data, face recognition from videos, and dynamic scene categorization supported our method. Since our formulation allows us to utilize any metric to construct VLAD, we plan to investigate how more robust metrics ($\ell_1$, $\ell_2$) can be employed in conventional VLAD. We are also interested to extend our framework to other types of Riemannian structures such as Kendall shape manifolds.

**Figure 2:** Examples of YouTube celebrity and UPENN datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>YouTube (Mean ± SD)</th>
<th>DynTex++ (Mean ± SD)</th>
<th>UPENN (Mean ± SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Best</td>
<td>71.4 ± 5.1 [23]</td>
<td>92.4 ± 4.1 [4]</td>
<td>96.2 ± 3.6 [13]</td>
</tr>
<tr>
<td>BOW$_G$-NN</td>
<td>60.3 ± 5.4</td>
<td>92.0 ± 0.7</td>
<td>81.2</td>
</tr>
<tr>
<td>BOW$_G$-SVM</td>
<td>64.5 ± 5.1</td>
<td>92.4 ± 0.5</td>
<td>92.9</td>
</tr>
<tr>
<td>BOW$_L$E-NN</td>
<td>50.6 ± 3.9</td>
<td>80.6 ± 0.9</td>
<td>75.2</td>
</tr>
<tr>
<td>BOW$_L$E-SVM</td>
<td>55.3 ± 2.9</td>
<td>81.1 ± 0.5</td>
<td>92.6</td>
</tr>
<tr>
<td>VLAD$_L$E-NN</td>
<td>62.1 ± 1.6</td>
<td>93.1 ± 0.6</td>
<td>86.4</td>
</tr>
<tr>
<td>VLAD$_L$E-SVM</td>
<td>65.2 ± 2.8</td>
<td>93.3 ± 0.4</td>
<td>96.9</td>
</tr>
<tr>
<td>R-VLAD$_G$-NN</td>
<td>75.5 ± 3.4</td>
<td>96.4 ± 0.4</td>
<td>90.0</td>
</tr>
<tr>
<td>R-VLAD$_G$-SVM</td>
<td>75.6 ± 2.5</td>
<td>96.7 ± 0.3</td>
<td>99.5</td>
</tr>
<tr>
<td>R-VLAD$_P$-NN</td>
<td>78.5 ± 3.6</td>
<td>96.9 ± 0.4</td>
<td>91.2</td>
</tr>
<tr>
<td>R-VLAD$_P$-SVM</td>
<td>79.9 ± 3.6</td>
<td>97.6 ± 0.4</td>
<td>99.8</td>
</tr>
</tbody>
</table>
Acknowledgements

Authors would like to thank Professor Richard Hartley for fruitful discussions.

This research was supported under Australian Research Councils Discovery Projects funding scheme (project DP150104645).

References


