High-order Local Pooling and Encoding Gaussians Over A Dictionary of Gaussians

Peihua Li, *Member, IEEE,* Hui Zeng, Qilong Wang, *Student Member, IEEE,* Simon C. K. Shiu, Lei Zhang, *Senior Member, IEEE*

***Abstract*—Local pooling (LP) in configuration (feature) space proposed by Boureau et al. explicitly restricts similar features to be aggregated, which can preserve as much discriminative information as possible. At the time it appeared, this method combined with sparse coding achieved competitive classification results with only a small dictionary. However, its performance lags far behind state-of-the-art results as only zero-order information is exploited. Inspired by the success of high-order statistical information in existing advanced feature coding or pooling methods, we make an attempt to address the limitation of LP. To this end, we present a novel method called high-order local pooling (HO-LP) to leverage the information higher than the zero-order one. Our idea is intuitively simple: we compute the first- and second-order statistics per configuration bin and model them as a Gaussian. Accordingly, we employ a collection of Gaussians as visual words to represent the universal probability distribution of features from all classes. Our problem is naturally formulated as encoding Gaussians over a dictionary of Gaussians as visual words. This problem, however, is challenging since the space of Gaussians is not a Euclidean space but forms a Riemannian manifold. We address this challenge by mapping Gaussians into the Euclidean space, which enables us to perform coding with common Euclidean operations rather than complex and often expensive Riemannian operations. Our HO-LP preserves the advantages of the original LP: pooling only similar features and using a small dictionary. Meanwhile, it achieves very promising performance on standard benchmarks, with either conventional, hand-engineered features or deep learning based features.**

***Index Terms*—Image classification, high-order local pooling (HO-LP), manifold of Gaussians.**

I. Introduction

In the past decade, great progress has been achieved in the area of visual recognition [1–3]. One of the most effective models is the Bag of visual Words (BoW) [1], which has been successfully applied in scene categorization [4], object classification [5], among others. The BoW framework is composed of several consecutive processes: (1) extraction of features in the input image, e.g., low-level, hand-engineered SIFT [6] or high-level, deep learning based features [7]; (2) encoding of these features over a dictionary of visual words learned from the training examples; (3) pooling (aggregation) of coding vectors to obtain image-level representations, which are then fed to a classifier such as support vector machine.

The pooling step is a necessary component of the BoW framework, and has a great effect on improving recognition accuracy [8]. The local spatial pooling plays a fundamental role at either small patch-level or the overall image-level. For image features such as SIFT, the gradients in a small patch are typically aggregated in smaller spatial cells and orientation cells, which are then concatenated to represent the properties of local neighborhoods. To obtain image-level representations, local pooling of coding vectors is widely used in the spatial bins of a pyramid (sub-regions at different scales) [1].

The local pooling (LP) in configuration space was first presented by Boureau et al. [9]. Its core idea is to only aggregate the codes of similar features so that more discriminative information can be preserved. Specifically, the space of training features are first divided into multiple cells (called configuration bins therein) using k-means clustering. The codes of features assigned to the same configuration bin are aggregated, leading to a pooling vector per configuration bin. All such pooling vectors are concatenated to form the final image representation. By considering the spatial pyramid strategy as well, the LP method ensures that features to be aggregated are close in the spatial-feature space. At the time, the LP method combined with sparse coding (SC) [10] achieved competitive results with only a small dictionary.

However, the main drawback of LP is lack of high-order information which limits its performance. Generally, exploiting high-order statistics can contribute to more accurate feature modeling and image representation. A recent work explicitly showed the benefits of high-order information in feature pooling method [11]. And numerous recent research exploiting high-order statistics has experimentally shown great gains in classification accuracy [12–14]. After extensive evaluations, Huang et al. [15] reported that the Fisher vectors (FV) [16], which exploits first- and second-order statistics, outperforms those only using zero-order information (e.g. SC, LLC [17], soft-assignment methods [18]), first-order statistics [19], or combination of zero- and first-order ones [20]. In view of the significance of LP yet its performance lagging behind state-of-the-arts, it is interesting to study whether high-order statistics can be leveraged for performance improvement under the LP framework.

In this paper, we propose a novel method called high-order local pooling (HO-LP) to tackle this problem. It exploits high-order information of the features and meanwhile maintain the merits of the original local pooling method. Different from LP

P. Li, and Q. Wang are with the School of Information and Communication Engineering, Dalian University of Technology, China.E-mail: peihuali@dlut.edu.cn, qlwang@mail.dlut.edu.cn

H. Zeng, Simon C. K. Shiu and L. Zhang are with the Department of Computing, The Hong Kong Polytechnic University. E-mail: {cshzeng, csckshiu, cslzhang}@comp.polyu.edu.hk

P. Li was supported by National Natural Science Foundation of China (No. 61471082); L. Zhang was supported by National Natural Science Foundation of China (No. 61672446).

which aggregates the coding vectors of similar features, we directly perform high-order statistical pooling of features falling into the same configuration bin. Specifically, we estimate the first- and second-order central moments in configuration bin, which are viewed as parameters of a Gaussian distribution (Gaussian for short). We first build a dictionary to represent the universal distribution of features from all classes which exploits both the center and covariance of each cluster to form a Gaussian as a visual word. Then, our problem is formulated as *how to encode Gaussians, collected in the joint spatial and configuration bins, over the dictionary of Gaussians as visual words* [1].

Encoding of Gaussians is a challenging problem, which, as far as we know, has not been studied previously. The main difficulty is that the space of Gaussians is not a Euclidean space but forms a Riemannian manifold [22]. This manifold is different from the common manifolds, e.g., manifold formed by symmetric positive definite (SPD) matrices, and thus the corresponding methods [23] cannot be applied. Feature coding based on kernel methods via (dis)similarity measures of Gaussians seems plausible, which however is not scalable to large scale datasets. An alternative is to perform feature coding by mapping them into Euclidean space. Unfortunately, the existing mapping schemes are computationally demanding, inappropriate to high-dimensional features.

To address this challenge, we present an effective method to map Gaussians into the Euclidean space. This method is suitable for statistical modeling of high-dimensional features, for which the diagonal covariance assumption is commonly adopted [16, 21]. In our method, we first identify Gaussians as affine matrices, all of which are known to form a matrix Lie group [24]. We proceed to map this matrix Lie group to its Lie algebra, the tangent space at the identity matrix. We show that the matrix logarithm is a diffeomorphism from this Lie group to its Lie algebra, thus establishing their equivalent relationship. In this way, we transform encoding of Gaussians to that of common vectors in the Euclidean space defined by this Lie algebra.

II. Related Work

There are roughly two categories of methods leveraging higher order information in image modeling. *In the first category*, dictionaries built from training samples of all categories are essentail for feature coding [12, 16, 25]. FV [16] learns a universal Gaussian mixture model (GMM) as a dictionary, and computes the derivatives of the likelihood function relative to the parameters of GMM, normalized by Fisher Information matrix. Motivated by FV, Kobayashi [12] proposed the Dirichlet Fisher kernel for histogram feature transforming and Dirichlet-derived GMM Fisher kernel; Klein et al. [25] derived Fisher vector expressions with respect to Laplacian Mixture Models and hybrid Gaussian-Laplacian Mixture Model. Li et al. [13] proposed a dictionary of affine subspaces and performed the first- and second-order coding of each feature over top-k nearest subspaces. Peng et al. [11] presented H-VLAD to include second- or third-order statistics. Recently, Wang et al. [26] developed a patch-level, end-to-end architecture called PatchNet, upon which they proposed a novel encoding method called vector of semantically aggregated descriptors (VSAD). The VSAD method performed FV-like encoding, where features extracted from one PatchNet pretrained on scene dataset are combined with the probabilities from another PatchNet pre-trained on ImageNet, achieving state-of-the-art result on SUN-397 dataset [27].

The methods *in the second category* do not need dictionary for feature coding. They estimated a global covariance matrix [28] or Gaussian distribution [14, 22, 29] to represent an image. Since either covariance matrices or Gaussians do not lie in the Euclidean space, their geometric structure needs to be favorably considered. Carreira et al. [28] introduced second-order pooling on free-form image regions, where the covariance matrices are mapped to vector space by the Log-Euclidean framework. Nakayama et al. [22] measured the similarity of different Gaussians based on information geometry. In [29], the covariance matrices are projected to the linear space which are then concatenated with the mean vectors before classification. Wang et al. [14] first embedded Gaussians into the space of SPD matrices and further mapped the SPD matrices to vector space according to the Log-Euclidean framework.

Our proposed HO-LP is different from the methods mentioned above. Specifically, we first perform *statistical local pooling*, i.e., we determine assignments of features into spatial-configuration bins, in each of which we compute mean and covariance to form a Gaussian. Next, we *encode* Gaussians with respect to a dictionary of Gaussians, and then concatenate coding vectors from all bins. Our method is also different from two-layer or multi-layer coding methods (e.g. [30, 31]), whose structure are similar to the architecture of CNN: the outputs of the current layer at finer spatial scale are fed to the next, spatially coarser layer, with each layer subject to pooling and coding operations. In contrast, our method only involves one layer, i.e., statistical pooling per spatial-configuration bin followed by coding with respect to a dictionary of Gaussians.

As far as we know, encoding Gaussians over a dictionary of Gaussians has not been studied previously. To deal with this challenge, one natural solution is to develop kernel methods based on probability (dis)similarity measures (e.g. Kullback-Leibler (KL) [32]). However, the kernel methods are computationally expensive for both classifier training and test, not scalable to large scale problems. The probability (dis)similarity measures among Gaussians are coupled, and currently for the kernels based on them there exist no approximate, finite-dimensional feature mappings fit for linear classifiers. Harandi et al. [33] proposed kernel-based Riemannian coding framework to handle the problem of coding on general manifolds, which maps points on one manifold into a reproducing kernel Hilbert space (RKHS) with some Riemannian kernels and performs SC or LLC in that RKHS. The proposed kernel LCC (kLCC) coding method (i.e., LLC in RKHS) can be computed analytically.

An alternative is to map Gaussians into the Euclidean space where the coding can be performed. Li et al. [34] embedded

[1]The dictionary consisting of a set of Gaussians has long been used in the BoW framework, e.g., spherical Gaussians in [18] and Gaussians with diagonal covariances in [16, 21].

Gaussians in the space of SPD matrices, which are then mapped to the linear space via the Log-Euclidean framework. Nakayama et al. [22] defined the space of Gaussians as a flat manifold characterized by an affine coordinate system, and the coordinates of Gaussians are further mapped to a common tangent space. However, in [22, 34], computation of mapping vectors is time consuming. Furthermore, for high-dimensional features (over hundreds of dimensions) diagonal covariances are generally used, the sizes of their mapping vectors are very large and redundant, making subsequent coding operations very expensive.

III. Proposed Method

We begin with an overview of the proposed method in Section III-A. Then we introduce the formulation of the high-order local pooling (HO-LP) in Section III-B. Finally we describe how to encode Gaussians in Section III-C.

A. Overview of our classification method

The illustration of image classification using the proposed high-order local pooling is shown in Fig. 1. For input images, we first extract either hand-engineered features or responses of pre-trained CNN models. Our construction of configuration bins and dictionary is similar to [9]. The difference is that we independently estimate two GMMs [2] rather than perform simple k-means clustering, using the features extracted in training images. One GMM is to partition the feature space into multiple bins (called configuration bins) and the other one serves as the dictionary to be used in feature coding. We also divide images into smaller sub-regions as in the spatial pyramid scheme [1]. Hence, we get a pre-defined, universal spatial-configuration bins, which jointly partitions the space of spatial position and the space of image features.

During coding process, we first assign features into the pre-defined spatial-configuration bins, in each of which we perform statistical pooling by estimating a Gaussian distribution. We map both per-bin Gaussian and the pre-trained dictionary consisting of Gaussians to a vector space through the proposed mapping $\phi = \log \circ \psi$ (see Section III-C). After that, the classical coding methods in the Euclidean space such as LLC or SC can be employed to encode the mapped Gaussians, leading to a coding vector per spatial-configuration bin. Finally, these coding vectors are weighted and concatenated as the final image representation for training and test with a linear SVM.

B. High Order Local Pooling

We represent an image feature by a two-tuple $[\mathbf{z}^T, \mathbf{f}^T]^T$, where $\mathbf{z} = [x, y] \in \mathbb{R}^2$ denotes the spatial coordinate and $\mathbf{f} \in \mathbb{R}^n$ is an n-dimensional feature, which is typically extracted from the patch centered at $\mathbf{z}$. The spatial pooling is performed in the spatial bins, which consists of a fixed, predetermined collection of possibly overlapping image regions. In practice, the spatial bins are the cells of the spatial pyramid [1], $\overline{\mathcal{Z}} = \{\mathcal{Z}_m, m = 1, \cdots, M\}$, which enables us to capture the local information at various scales.

The configuration pooling is performed in a set of configuration bins, i.e., predetermined multi-dimensional regions in the configuration space. The configuration bins, denoted by $\overline{\mathcal{F}} = \{\mathcal{F}_k, k = 1, \cdots, K\}$, can be constructed by Voronoi tessellation of the configuration space via the k-means clustering [9]. A better way is to employ the Gaussian mixture model (GMM). In this case, we compute the responsibility (or posterior probability) of one feature and assign it to the cluster with maximal responsibility. The configuration bins constructed by k-means have linear boundaries, while those by GMM have quadratic ones.

Our local pooling is performed in the product space

$$\overline{\mathcal{Z}} \times \overline{\mathcal{F}} = \{(\mathcal{Z}_m, \mathcal{F}_k)\}_{m=1,\ldots,M, k=1,\ldots,K}. \tag{1}$$

Clearly the pooling operations take place locally both in the space of spatial position and in the configuration space. Hence, only similar features in the spatial-configuration space are to be aggregated while dissimilar ones are not. Specifically, given an input image, we determine the assignments of features $\mathbf{f}$, then we estimate the "personalized" statistics of features for each bin $(\mathcal{Z}_m, \mathcal{F}_k)$, by computing their first- and second-order moments viewed as parameters of a Gaussian:

$$g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_{(m,k)}, \widehat{\boldsymbol{\Sigma}}_{(m,k)}) = \tag{2}$$
$$|2\pi\widehat{\boldsymbol{\Sigma}}_{(m,k)}|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\mathbf{f} - \widehat{\boldsymbol{\mu}}_{(m,k)})^T \widehat{\boldsymbol{\Sigma}}^{-1}_{(m,k)}(\mathbf{f} - \widehat{\boldsymbol{\mu}}_{(m,k)})),$$

where $|\cdot|$ denotes the matrix determinant, $\widehat{\boldsymbol{\mu}}_{(m,k)}$ and $\widehat{\boldsymbol{\Sigma}}_{(m,k)}$ are respectively the mean vector and covariance matrix, estimated by maximum likelihood estimation from the features in cell $(\mathcal{Z}_m, \mathcal{F}_k)$. This kind of pooling is in the statistical sense, clearly different from the commonly used average pooling or max pooling. It is natural since we are concerned with high-order statistics.

As in [9], we build the dictionary from the training sample corpus by the clustering methods, independent of construction of the configuration bins. In practice, we build dictionary by estimating a universal GMM based on the expectation maximization (EM) algorithm [16, 21]. Hence, our dictionary consists of a collection of Gaussians

$$\mathcal{G} = \{g(\mathbf{f}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\}_{j=1,\cdots,N}, \tag{3}$$

where $g(\mathbf{f}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ is the j-th component of the GMM. Different from [9] where only cluster centers (first-order) are employed as visual words, we consider both the centers (first-order) and the shapes (second-order) of clusters, which allow us to describe the distribution of features more accurately.

As a consequence of HO-LP, we face the following encoding problem: *How to encode a Gaussian* $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_{(m,k)}, \widehat{\boldsymbol{\Sigma}}_{(m,k)})$ *over a dictionary of Gaussians* $\{g(\mathbf{f}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \ldots, g(\mathbf{f}|\boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)\}$*?* This problem is challenging as the the space of Gaussians is not a vector space but forms a Riemannian manifold. Our idea is, by mapping Gaussians to a vector space via some function $\phi(\cdot)$,

[2]Note that for statistical modeling of high-dimensional features, GMMs with full covariances incur numerical instability and heavy, computational cost. Hence, GMMs with diagonal covariances are widely used in literature [16, 21].

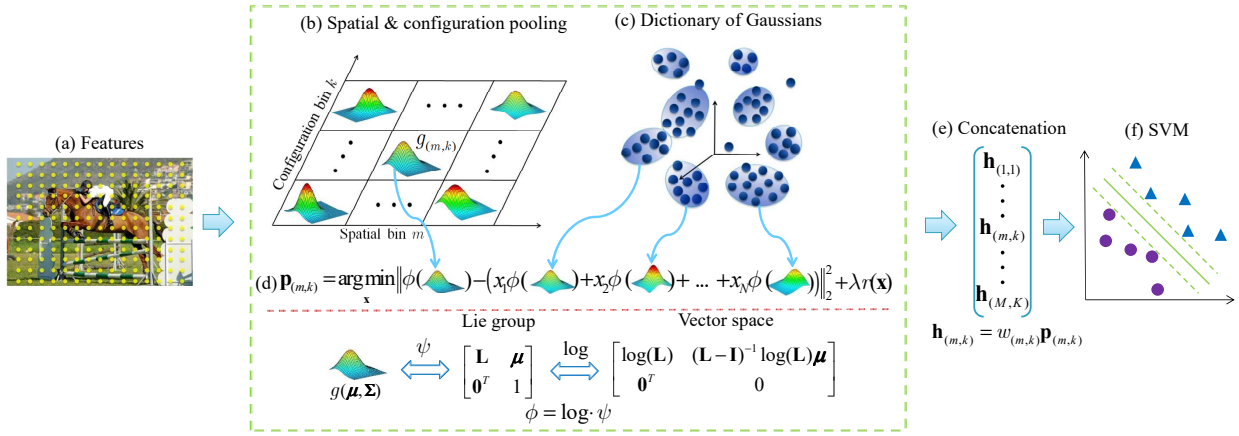


Fig. 1. Illustration of image classification using the proposed higher-order local pooling (HO-LP). For an input image, we extract features and assign them into pre-defined spatial-configuration bins, in each of which statistical pooling is performed to construct a Gaussian. Both the Gaussians estimated in spatial-configuration bins and the dictionary composed of Gaussians are mapped to vector space using the proposed mapping $\phi = \log \circ \psi$. Hence, encoding of Gaussians on the Gaussian manifold is transformed to the classical, vector coding in the Euclidean space, and so the common methods such as SC or LLC can be used. The per-bin coding vectors are weighted and concatenated as the final image representation for classification with a linear SVM.

we transform encoding of Gaussians to the classical encoding of vectors in the Euclidean space

$$\mathbf{p}_{(m,k)} = \arg\min_{\mathbf{x}} \|\phi(g_{(m,k)}) - \sum_{j=1}^{N} x_j \phi(g_j)\|_2^2 + \lambda r(\mathbf{x}), \quad (4)$$

where $g_{(m,k)}$ and g_j are abbreviations of $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_{(m,k)}, \widehat{\boldsymbol{\Sigma}}_{(m,k)})$ and $g(\mathbf{f}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, respectively, $\lambda > 0$ and $r(\mathbf{x})$ is a regularizing function. The coding vector $\mathbf{p}_{(m,k)}$ of Gaussian $g_{(m,k)}$ is weighted to reflect relative importance among all coding vectors,

$$\mathbf{h}_{(m,k)} = w_{(m,k)}(\alpha) \cdot \mathbf{p}_{(m,k)}. \quad (5)$$

Here $w_{(m,k)}(\alpha)$ takes the following form:

$$w_{(m,k)}(\alpha) = \left(\widehat{N}_{(m,k)} / \sum_{k'=1}^{K} \widehat{N}_{(m,k')} \right)^{\alpha}, \quad (6)$$

where $0 \leq \alpha \leq 1$ and $\widehat{N}_{(m,k)}$ is the occurrence of features in $(\mathcal{Z}_m, \mathcal{F}_k)$. Indeed, $w_{(m,k)}$ represents the zero-order statistics in our model. We will see in Section IV-B that appropriate choice of the value of α brings performance improvement. The final coding vector (image-level representation) is a concatenation of the coding vectors of all bins, i.e., $\mathbf{h}_{(m,k)}$, $m = 1, \cdots, M$, $k = 1, \ldots, K$, which is of size $M \times K \times N$.

An alternative method to handle our coding problem is using the kernel-based Riemannian coding framework proposed by Harandi et al. [33]. Specifically, we can first map the Gaussian distributions into RKHS with some Riemannian kernels and then perform LLC in that RKHS as kLCC has analytical solution. Comparisons between the proposed mapping scheme and kLCC are conducted in the experimental section.

C. Encoding of Gaussians

In this section, we begin with an introduction of identifying Gaussians as affine matrices, and then describe how to map the affine matrices into the Euclidean space, where coding is actually performed.

1) Shape of Gaussians and Geodesic Distance: The theory of shape of Gaussians (SoG) was introduced in [24]. Assume that $\mathbf{f}_0$ is an n-dimensional random vector following the standard (multivariate) Gaussian distribution, and that $\boldsymbol{\Sigma}$ is an SPD matrix. It is well known that $\boldsymbol{\Sigma}$ has a unique Cholesky factorization $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is a lower triangular matrix with positive diagonals. Consider the affine transformation, $\mathbf{f} = \mathbf{L}\mathbf{f}_0 + \boldsymbol{\mu}$, or equivalently in homogeneous coordinate form,

$$\begin{bmatrix} \mathbf{f} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{L} & \boldsymbol{\mu} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{f}_0 \\ 1 \end{bmatrix}, \quad (7)$$

where $\mathbf{0}$ denotes a $1 \times n$ zero vector. According to the property of Gaussians, we know that the probability density function of random vector $\mathbf{f}$ is a Gaussian with the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$. Eq. (7) is called *positive definite lower triangular affine transformation* (PDLTAT), and the affine matrix involved is called PDLTAT matrix. In this way, an arbitrary Gaussian $g(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be uniquely mapped to the corresponding PDLTAT matrix through

$$\psi : g(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \rightarrow \begin{bmatrix} \mathbf{L} & \boldsymbol{\mu} \\ \mathbf{0} & 1 \end{bmatrix} \quad (8)$$

Consider the set of all PDLTAT matrices,

$$G(n+1) = \left\{ \mathbf{P}_{\mathbf{L},\boldsymbol{\mu}} \triangleq \begin{bmatrix} \mathbf{L} & \boldsymbol{\mu} \\ \mathbf{0} & 1 \end{bmatrix} \middle| \mathbf{L} \in L^+(n), \boldsymbol{\mu} \in \mathbb{R}^n \right\} \quad (9)$$

where $L^+(n)$ denotes the set of $n \times n$ lower triangular matrices with positive diagonals. $G(n+1)$ is closed under regular matrix multiplication and matrix inversion, both of which are evidently smooth. Hence, $G(n+1)$ is a Lie group which forms a Riemannian manifold. The distance between $g(\mathbf{f}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $g(\mathbf{f}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ are measured by the geodesic length ρ connecting their corresponding PDLTAT matrices $\mathbf{P}_{\mathbf{L}_1,\boldsymbol{\mu}_1}$ and $\mathbf{P}_{\mathbf{L}_2,\boldsymbol{\mu}_2}$, which is of the form

$$\rho(\mathbf{P}_{\mathbf{L}_1,\boldsymbol{\mu}_1}, \mathbf{P}_{\mathbf{L}_2,\boldsymbol{\mu}_2}) = \left\| \log(\mathbf{P}_{\mathbf{L}_1,\boldsymbol{\mu}_1}^{-1} \mathbf{P}_{\mathbf{L}_2,\boldsymbol{\mu}_2}) \right\|_F, \quad (10)$$

where $\log(\cdot)$ denotes the matrix logarithm and $\|\cdot\|_F$ the matrix Frobenius norm. Since $G(n+1)$ is a Riemannian manifold,

the common Euclidean operations cannot be applied to it. The geodesic distance (10) is not decoupled, and therefore, if employed in coding methods, one will have to turn to the kernel methods which are known to be unscalable to large-scale problems. Above all, it is unclear whether the kernels based on (10) are positive definite.

2) Mapping $G(n+1)$ to Its Lie Algebra: A Lie group is a differentiable manifold which is locally Euclidean. The Riemannian metric, which defines the inner product on the tangent space, varies from point to point on the manifold. The Lie algebra of a Lie group is a vector space, more specifically, the tangent space at the identity element. The Lie algebra of $G(n+1)$, denoted by $\mathfrak{g}(n+1)$, can be written as [35]

$$\mathfrak{g}(n+1)=\left\{\begin{bmatrix}\mathbf{X} & \mathbf{t}\\ \mathbf{0} & 0\end{bmatrix}\middle|\mathbf{X}\in L(n),\mathbf{t}\in\mathbb{R}^n\right\}, \tag{11}$$

where $L(n)$ is the set of $n\times n$ lower triangular matrices.

The matrix exponential establishes a mapping between a matrix Lie group and its Lie algebra. However, this mapping may be neither one to one nor onto, and indeed, in general it is only locally one to one and onto: there exists a neighborhood (containing the zero element) of the Lie algebra which can be homomorphically mapped to some neighborhood (containing the identity) of the corresponding Lie group [36, Chap. 2.7]. Hence, an element in the Lie group usually can not be mapped uniquely to its Lie algebra. Fortunately, for our case, we find that the exponential (or its inverse, the logarithm) is a smooth bijection between $G(n+1)$ and its Lie algebra $\mathfrak{g}(n+1)$, as described in the following theorem:

Theorem 1: The matrix logarithm

$$\log: G(n+1)\rightarrow\mathfrak{g}(n+1),\ \mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}\mapsto\log(\mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}) \tag{12}$$

is a diffeomorphism. In particular, for a Gaussian $g(\mathbf{f}|\boldsymbol{\mu},\mathrm{diag}(\sigma_i^2))$ with mean vector $\boldsymbol{\mu}=[\mu_i]$ and diagonal covariance matrix $\mathrm{diag}(\sigma_i^2)$, the logarithm of its PDLTAT matrix has closed-form:

$$\log\begin{bmatrix}\mathrm{diag}(\sigma_i) & [\mu_i]\\ \mathbf{0} & 1\end{bmatrix}=\begin{bmatrix}\mathrm{diag}(\log\sigma_i) & \left[\frac{\mu_i\log\sigma_i}{\sigma_i-1}\right]\\ \mathbf{0} & 1\end{bmatrix}. \tag{13}$$

Here $[\mu_i]$ is an abbreviation of the vector $[\mu_1,\ldots,\mu_n]^T$. Proof of Theorem 1 is given in Appendix A.

We clarify that the conclusion that $\log: G(n+1)\rightarrow\mathfrak{g}(n+1)$ is a diffeomorphism, as stated in Theorem 1, has not appeared either in [24] or other previous literature. We noticed that Cheng et al. [37] mentioned the existence of mappings between the Lie group formed by PDLTAT matrices with full covariance matrices and their Lie algebra. And Li et al. [38] identifies the Gaussian distribution as an upper triangular matrix and discloses the Lie group structure of Gaussian distributions in the Log-Euclidean framework. Although sharing some similarity with them, Theorem 1 mathematically proves that PDLTAT matrices with diagonal covariance matrices can be uniquely mapped into their Lie algebra (linear space) with matrix logarithm and derives an explicit mapping expression which is clearly distinct from both of them. We have three remarks regarding Theorem 1 as follows:

(1) It establishes the equivalence between Lie group $G(n+1)$ and its Lie algebra $\mathfrak{g}(n+1)$, so that the Riemannian operations on $G(n+1)$ can be transformed, through the logarithm, to the Euclidean operations in $\mathfrak{g}(n+1)$.

(2) Under the logarithm, the geodesic distance between any two PDLTAT matrices is preserved (to the first order approximation) in its Lie algebra [35, Section 4.1]:

$$\|\log(\mathbf{P}_{\mathbf{L}_1,\boldsymbol{\mu}_1}^{-1}\mathbf{P}_{\mathbf{L}_2,\boldsymbol{\mu}_2})\|_F\approx\|\log(\mathbf{P}_{\mathbf{L}_1,\boldsymbol{\mu}_1})-\log(\mathbf{P}_{\mathbf{L}_2,\boldsymbol{\mu}_2})\|_F \tag{14}$$

(3) The widely used features are of high dimension, for which Gaussians with diagonal covariances are usually employed for statistical modeling. Such Gaussians can be mapped to the Euclidean space at negligible cost via Eq. (13).

As a Lie group is a manifold involving complicated, expensive Riemannian operations, it is a common practice to map the elements to its Lie algebra, which is a vector space and where simple and efficient Euclidean operations can be used. For example, on the Lie group of SPD matrices, the well-known Log-Euclidean metric [39], maps through the logarithm the SPD matrices to the corresponding Lie algebra where the Euclidean distances are measured. Similar to Eq. (14), in terms of Baker-Campbell-Hausdorff formula [35, Section 4.1], this metric is the first-order approximation of the geodesic distance (a.k.a. affine Riemannian metric). For such first-order approximations, it is often difficult to analyze theoretically how good the approximation is, as in most cases closed-form error functions can not be obtained. Instead, researchers [35, 39] including us are more concerned with the effectiveness and efficiency in light of experimental validation.

3) Encoding of Gaussians in vector space: For an input image, we estimate a collection $\{g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_{(m,k)},\widehat{\boldsymbol{\Sigma}}_{(m,k)})\}_{m=1,\cdots,M,k=1,\cdots,K}$ of Gaussians with diagonal covariances, and our task is to encode them over a dictionary of Gaussians with diagonal covariances, i.e. $\{g(\mathbf{f}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)\}_{j=1,\cdots,N}$. For notational simplicity, we transform two-dimensional indexes (m,k) of bins $(\mathcal{Z}_m,\mathcal{F}_k)$ to one-dimensional ones, e.g., by letting $t=(m-1)K+k$, $t=1,\ldots,K\times M$. Let $\mathbf{P}_{\widehat{\mathbf{L}}_t,\widehat{\boldsymbol{\mu}}_t}$ be the corresponding PDLTAT matrix of $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_t,\widehat{\boldsymbol{\Sigma}}_t)$, where $\widehat{\boldsymbol{\Sigma}}_t=\mathrm{diag}(\widehat{\sigma}_{ti}^2)$ is a diagonal covariance matrix. Since the logarithm of a PDLTAT (Log-PDLTAT for short) matrix is in the Euclidean space, for easy manipulation we vectorize it as follows:

$$\mathbf{q}_t=\left[\log\widehat{\sigma}_{t1},\ldots,\log\widehat{\sigma}_{tn},\frac{\widehat{\mu}_{t1}\log\widehat{\sigma}_{t1}}{\widehat{\sigma}_{t1}-1},\ldots,\frac{\widehat{\mu}_{tn}\log\widehat{\sigma}_{tn}}{\widehat{\sigma}_{tn}-1}.\right]^T \tag{15}$$

Correspondingly, we denote by $\mathbf{d}_j$ the Log-PDLTAT matrix of $g(\mathbf{f}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$ in the dictionary $\mathcal{G}$. Computation of $\mathbf{q}_t$ only takes $O(2n)$ time.

In the vector space formed by the Lie algebra of $G(n+1)$, we employ the locality-constrained linear coding (LLC) [17]. We select LLC rather than sparse coding (SC) [10] because LLC is much faster, while guaranteeing that similar features have similar codes, a desirable property that SC fails to have. Note that the locality constraint directly leads to the sparsity. Let $\mathcal{N}_p(\mathbf{q}_t)$ be the set of top-p nearest neighbors of $\mathbf{q}_t$ in $\{\mathbf{d}_j\}_{j=1,\cdots,N}$. Let $\overline{\mathcal{K}}=\{1,2,\cdots,N\}$. We denote by $\mathcal{K}_t$ the set of indexes for which $\mathbf{d}_j\in\mathcal{N}_p(\mathbf{q}_t)$, i.e., $\mathcal{K}_t=\{j|j\in$

$\overline{\mathcal{K}}, \mathbf{d}_j \in \mathcal{N}_p(\mathbf{q}_t)\}$. The objective function of the LLC can be written as

$$\mathbf{p}_t = \arg \min_{\forall x_{tj}, j \in \mathcal{K}_t} \left\|\mathbf{q}_t - \sum_j x_{tj}\mathbf{d}_j\right\|_2^2 \text{ s.t. } \sum_j x_{tj} = 1. \quad (16)$$

For coding vector $\mathbf{x}_t = [x_{t1}, \cdots, x_{tN}]$, if $j \in \mathcal{K}_t$, x_{tj} is computed according to (16), and otherwise we set $x_{tj} = 0$. The problem (16) can be solved by the Least Square method whose complexity is $O(2np^2)$ [40].

Mapping scheme in [34]: Li et al. proposed to embed $n-$dimensional Gaussians in the space of $(n+1) \times (n+1)$ SPD matrices, which are further mapped to the vector space based on the Log-Euclidean framework [41]. For Gaussian $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_t, \widehat{\boldsymbol{\Sigma}}_t)$ with diagonal covariance matrix, the mapping vector takes the following form:

$$\tilde{\mathbf{q}}_t = \text{vec}\left(\log \begin{bmatrix} \text{diag}(\widehat{\sigma}_{ti}^2) + \widehat{\boldsymbol{\mu}}_t\widehat{\boldsymbol{\mu}}_t^T & \widehat{\boldsymbol{\mu}}_t \\ \widehat{\boldsymbol{\mu}}_t^T & 1 \end{bmatrix}\right), \quad (17)$$

where $\text{vec}(\mathbf{A})$ denotes the operation which vectorizes the upper triangular entries of $\mathbf{A}$. Note that the dimension of $\tilde{\mathbf{q}}_t$ is $(n+1)(n+2)/2$. The computation of $\tilde{\mathbf{q}}_t$ costs $O(10(n+1)^3)$ operations via eigen-decomposition [34] while the solution to LLC costs $O((n+1)(n+2)p^2/2)$.

Mapping scheme in [22]: Nakayama et al. defined a flat manifold by taking an appropriate affine coordinate system η, in which the tangent spaces are flatly connected. This coordinate system is interpreted as the space of sufficient statistics. Let $g(\mathbf{f}|\overline{\boldsymbol{\mu}}, \overline{\boldsymbol{\Sigma}})$ be the "average" Gaussian estimated from the features of the entire training corpus, and $\mathbf{F}(\overline{\boldsymbol{\mu}}, \overline{\boldsymbol{\Sigma}})$ the corresponding Riemannian metric. Any Gaussian $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_t, \widehat{\boldsymbol{\Sigma}}_t)$ can be mapped to the tangent space of the average Gaussian:

$$\hat{\mathbf{q}}_t = \mathbf{F}^{\frac{1}{2}}(\overline{\boldsymbol{\mu}}, \overline{\boldsymbol{\Sigma}})\left[\widehat{\boldsymbol{\mu}}_t^T, (\text{vec}(\text{diag}(\widehat{\sigma}_{ti}^2) + \widehat{\boldsymbol{\mu}}_t\widehat{\boldsymbol{\mu}}_t^T))^T\right]^T, \quad (18)$$

where $\mathbf{F}^{\frac{1}{2}}$ denotes the square root of $\mathbf{F}$. The dimension of $\hat{\mathbf{q}}_t$ is $n(n+3)/2$. One needs $O(n^2(n+3)^2/4)$ time and $O(n(n+3)p^2/2)$ time to compute $\hat{\mathbf{q}}_t$ and solve the LLC problem, respectively.

IV. Experiments

We start with an introduction of the benchmarks and the experimental setup. Then we evaluate the proposed HO-LP from several respects and also make a comparison with LP. Finally, we compare HO-LP with other methods under different feature settings.

A. Benchmarks and experimental setup

We employ six widely used image benchmarks in the experiments. Some sample images from these datasets are shown in Fig. 2.

Scene-15 [1] This dataset contains 4,485 images of 15 scene categories. According to the standard experimental setup, we randomly select 100 training images per category and the remaining ones for testing. We randomly repeat the experiments 5 times and report the average accuracy.

FMD [42] This dataset is composed of 1,000 material images of 10 material categories. Following the common protocol, for each category, we randomly select 50 images for training and the remaining 50 images for testing without using the provided binary mask. We report the average accuracy over five trials.

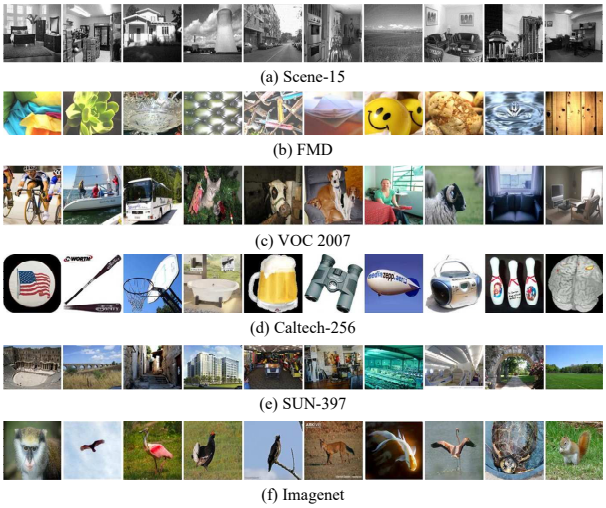


Fig. 2. Some sample images from (a) Scene-15, (b) FMD, (c) VOC 2007, (d) Caltech-256, (e) SUN-397 and (f) ImageNet.

PASCAL VOC 2007 [2] This dataset contains 20 categories and a total of 9,963 images. We follow the standard protocol: the training, validating and testing are performed on the "train", "val" and "test" sets, respectively. We employ the mean Average Precision (mAP) over 20 categories for performance measurement.

Caltech-256 [43] This dataset includes 256 object categories and a background class, with 30,607 images in total. Following the usual practice [16], we conduct experiments with varying number of training images per category: 15, 30, 45, 60. The remainder of images is used for testing. We average the classification accuracy over five random train/test splits.

SUN-397 [27] This dataset contains 108,754 images of 397 different scene categories. Following the protocol in [27], we use the pre-defined ten splits for evaluation and test the performance using 5, 10, 20 or 50 samples per class for training and 50 samples for testing. The average accuracy of ten rounds is reported.

ImageNet [44] This is large-scale dataset for object recognition. It provides a training set consists of 1.2 million labeled images in 1000 categories, with 732 to 1300 images for each category. It also provides a validation set containing 50 samples from each category and a test set with 100 samples for each category. The top1 and top5 error rates are usually employed to measure the performance on this dataset.

Recent works [7, 45–47] have shown that responses of CNN pre-trained on large ImageNet as features (which we call CNN features) achieved state-of-the-art results on a variety of vision tasks. In this paper, we make experiments using both SIFT, one of the most widely used hand-engineered features, and CNN-based features. Our purpose is to test whether the proposed HO-LP can generalize well to traditional, low-level features (e.g. SIFT) and novel, high-level features (e.g. CNN features).

Several efforts have been made in order to estimate mean-

ingful high-order statistics (i.e. covariances) in each confi uration and spatial bin. This first and most important one that we use diagonal covariances rather than full covarianc which alleviates the demand of samples. Secondly, we u multi-scale strategy to extract a large number of features p image. Specifically, about 70,000 SIFT features are extract from each image on multi-scale square patches of size r $2^{4+i/2}$ with $r/4$ pixel strides, $i = 0, 1, \cdots, 4$, which indicat several hundred of SIFT features are on average assigned each bin. The feature allocation of three sample images a shown in the middle of Fig. 3. The SIFT features are furth reduced to $n = 64$ by PCA to make the diagonal assumpti more suitable. The local spatial pooling is performed inside three-level pyramid (1×1, 3×1 and 2×2 sub-regions) on benchmarks.

Following [7, 45], the *CNN features* are extracted as t responses of the last convolutional layer (immediately after the ReLU operation). Specifically, we resize isotropically each image so that its maximum side is no more than 500 pixels. Then we rescale isotropically the resized image at 5 scales $2^s, s = -0.8, -0.4, 0, 0.4, 0.8$. In this way, we can extract around 4,000 CNN features for each image. For this case, we do not use the spatial pyramid strategy which encourages more features to be assigned into each bin. More importantly, we experimentally find that the feature allocation tends to be sparse in the case of high dimensional CNN features. As illustrated in the right of Fig. 3, only several bins receive most of the features, and similar observation is also found in [46]. Hence, we can estimate diagonal covariances in the bins with sufficient features.

In addition, the weighting strategy (Eq. 5) helps us further control the impact of number of features on their coding vectors. For the bins having few features, we directly set their weights to zero. Such bins never contribute to the final image representation. Finally, a small positive number (1e-4) is added to the diagonal covariances for numerical stability throughout the experiments.

Similar to [9], we separately estimate one GMM for building configuration bins and a second GMM as the dictionary. We determine the number of neighbors in LLC by cross validation. The final coding vector is ℓ_2-normalized before fed to a linear SVM. The SIFT extraction, k-means algorithm or GMM estimation, and one-versus-all SVM classifier are all implemented using the subroutines in the VLFeat software package [48]. Extraction of CNN features with pre-trained VGG-M [49] and VGG-VD [50] are implemented with MatConvNet [51], without using techniques of data augmentations or fine-tuning. The programs are written with Matlab, running on a PC with i7-4790k processor @4.0GHz and 64GB RAM.

B. Evaluation of HO-LP on VOC 2007

We conduct a series of experiments on VOC 2007 to evaluate the parameters of the proposed HO-LP under the hand-engineered feature (SIFT) setting. In particular, we test the effects of combination of configuration bin number K and dictionary size N, and weight parameter α in Eq. (4) on HO-LP. We also compare our mapping scheme with the other ones, all of which can map Gaussians into the vector space.

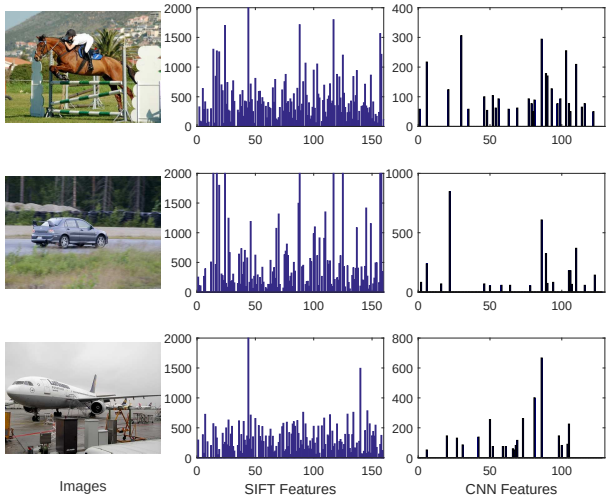


Fig. 3. Sample images and corresponding features allocation in configuration bins. The x-axis indicates the index of configuration bins and the y-axis shows the number of features assigned into each bin. Bins number are set to 160 and 128 for SIFT and CNN features, respectively. No spatial pyramid is considered in both cases for simplicity. For SIFT features, we set a upper limit (2000) to y-axis for more clear illustration of all bins.

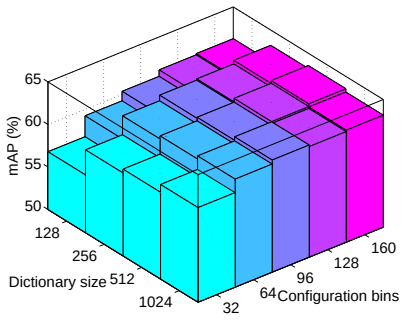


Fig. 4. Effects of combination of configuration bin number K and dictionary size N on VOC 2007.

Configuration bin number and dictionary size Fig. 4 shows the accuracy as a function of K and N, where we choose $\alpha = 0$, i.e., setting identical weight to coding sub-vectors of all configuration bins. We find for fixed K ($K \leq 96$), the mAP increases continuously with N; however, for larger K, the value of mAP increases until $N = 512$ and then begins to drop. On the other hand, when N is fixed, the value of mAP increases consistently with growing K. Interestingly,

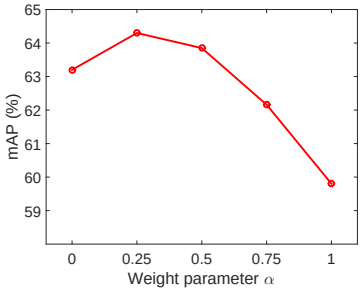


Fig. 5. Effects of weight parameter α on VOC 2007.

TABLE I
COMPARISON OF DIFFERENT MAPPING SCHEMES AND KLCC [33] ON VOC 2007.

Mapping scheme	Time (s)	mAP (%)
HO-LP (Eq. (17) [34])	7.1	62.8
HO-LP (Eq. (18) [22])	7.0	62.4
HO-LP (kLCC-KL [32])	4.2	61.1
HO-LP (kLCC-LogE [34])	8.8	57.2
HO-LP (Eq. (19))	0.9	63.2
HO-LP (Eq. (15))	0.9	64.3

the performance appears not to saturate, indicating potentially higher performance with a larger K, but at considerably increased cost. The largest mAP value, 63.6%, is achieved with $K = 160$ and $N = 512$. To trade-off between accuracy and speed, we set $K = 160$ and $N = 256$ throughout the following experiments if no additional mention, with which HO-LP yields 63.2% in recognition accuracy with the final dimension of 327,680 ($8 \times 160 \times 256$). We also make experiments where both the configuration bins and dictionary are obtained by using k-means, and the results show that accuracies drop by more than 1%.

Weight parameter We proceed to test the effect of α on HO-LP ($K = 160$ and $N = 256$). From Fig. 5, it can be seen that the mAP reaches a peak value, 64.3%, at $\alpha = 0.25$ and then decreases gradually. As explained in [16, Section 2.3], this power technique restrains the side effect of features which take place duplicately in classification. The optimal value of α may change slightly with K. In all the following experiments, we fix this parameter to 0.25.

Mapping schemes & kernel LCC Under exactly the same experimental setting (dictionary with 256 Gaussians, 160 configuration bins, 8 spatial bins, etc.), we now compare different mapping schemes for HO-LP which first map Gaussians into vector spaces and then perform LLC in the vector space, as well as HO-LP using kernel LCC (kLCC) [33] for coding. Specifically, we compare our mapping scheme (15) with that of Li et al. [34] and that of Nakayama et al. [22], as shown in Eq. (17) and Eq. (18), respectively. We also compare with a naive mapping scheme which simply vectorizes the matrix $\mathbf{P}_{\widehat{\mathbf{L}}_t, \widehat{\boldsymbol{\mu}}_t}$ corresponding to Gaussian $g(\mathbf{f}|\widehat{\boldsymbol{\mu}}_t, \widehat{\boldsymbol{\Sigma}}_t)$ as follows:

$$\mathbf{p}'_t = [\widehat{\sigma}_{t1}, \ldots, \widehat{\sigma}_{tn}, \widehat{\mu}_{t1}, \ldots, \widehat{\mu}_{tn}]^T. \quad (19)$$

We evaluated kLCC based on two different kernels including the KL-kernel [32] and Log-Euclidean kernel [34]. The source code of kLCC was kindly provided by the authors of [33][3]. Note that we did not employ dictionary learning for kLCC, since all the other compared methods did not use it for fair comparison. We have tried our best to optimize the parameters of these two methods.

Table I presents both the accuracy (mAP) and average running time taken by image-level modeling, which includes feature extraction, pooling and coding. We can first notice that both the mapping scheme of Li et al. [34] and that of Nakayama et al. [22] are time consuming, taking about eight times the computing time as ours. And they yield slightly inferior performance, compared to the other two mapping schemes. The reason may be that, in their schemes, the components of the mean vector are unfavorably distributed in the covariance matrix. Regarding the kLCC, the cost is also much higher than the proposed mapping scheme. This is because that in order to find the nearest neighbors, we need to calculate the kernel distance between each dictionary atom and local Gaussian. Moreover, the accuracy of kLCC is not very competitive in our situation which may be mainly caused by the diagonal covariance and high-dimension feature we used. In such case, distance computation and numerical stability may have larger effect on kernel methods, which limits their effectiveness. Our mapping scheme takes almost the same time as the naive one, only about 0.9s, but has over 1% performance gain, indicating that our method improves the performance at negligible cost. Considering that both the mapping schemes in [34] and in [22] and kLCC are computationally demanding, not suitable for the framework of HO-LP. We do not make experiments using these methods in the remaining experiments.

[3]We appreciate Harandi for sending us the source code of kLCC.

C. Comparison with LP and FV-based LP

In the second set of experiments we first compare the proposed HO-LP with the original LP [9]. This comparison is made on Scene-15 [1] and Caltech-256 [43] (30 training samples). As LP uses sparse coding (SC), we also implement the proposed HO-LP with SC in the embedding vector space, i.e., $r(\mathbf{x})$ is selected as the ℓ_1-norm in Eq. (4). Table II(a) presents the comparison results under three combinations of dictionary size N and configuration bin number K.

It can be clearly seen that the performance gains of HO-LP(SC) over LP are significant, in any case and on any dataset. Regarding the best results of these two methods, the gaps between HO-LP(SC) and LP are 4.7% on Scene-15 and 9.7% on Caltech-256, respectively. We ascribe this big improvement to the first- and second-order statistics successfully leveraged in the proposed HO-LP. Also, we find that in all cases HO-LP(LLC) performs slightly better than HO-LP(SC) but is much faster. Note that the final coding vector sizes for the two methods are comparable.

We also compare the proposed method with Fisher Vector in the local pooling paradigm (called FV-LP for simplicity) under the same experimental settings. Specifically, for FV-LP, we allocate features to spatial and configuration bins, in each of which the features are encoded using FV and then aggregated to a single vector, and finally the aggregated vectors of all bins are concatenated to obtain the image-level representation. The comparisons are conducted on Scene-15 and Caltech-256 datasets using SIFT descriptors. For both HO-LP and FV-LP, the features are extracted at five scales whose dimensions are reduced to 64, the number M of configuration bins and number K of spatial bins are 128 and 8, respectively. The number of Gaussian components of the universal GMM as dictionary is set to 256 for both methods. As the aggregated FV per bin is of high dimension (32,768-D), straightforward concatenation leads to expensive computations as well as excessive storage and so we adopt PCA for compactness. The comparison results are presented in Table II(b).

TABLE II
COMPARISON WITH LP (A) AND FV-LP (B) ON SCENE-15 AND CALTECH-256 (30 TRAINING SAMPLES).

(a)

	$N \times K \times M$	LP [9]	HO-LP (SC)	HO-LP (LLC)
Scene-15	$256 \times 128 \times 8$	81.1 (0.5)	88.0 (0.3)	88.9 (0.2)
	$1024 \times 64 \times 8$	82.4 (0.7)	87.2 (0.3)	88.8 (0.3)
	Best	83.3 (1.0)	88.0 (0.3)	89.1 (0.5)
Caltech-256	$256 \times 128 \times 8$	40.3 (0.6)	51.4 (0.3)	52.0 (0.2)
	$1024 \times 64 \times 8$	41.7 (0.8)	50.8 (0.3)	51.8 (0.2)
	Best	41.7 (0.8)	51.4 (0.3)	52.1 (0.1)

(b)

$N \times K \times M$	Methods		Scene-15	Caltech-256	Image-level repres. size	Time (s)
$256 \times 128 \times 8$	HO-LP (LLC)		88.9 (0.2)	52.0 (0.2)	262,144-D	0.90
	FV-LP	(PCA256)	88.1 (0.4)	49.0 (0.2)	262,144-D	1.72
		(PCA512)	88.4 (0.5)	49.7 (0.1)	524,288-D	1.98
		(PCA768)	88.5 (0.3)	50.2 (0.1)	786,432-D	2.22

TABLE III
COMPARISON WITH DIFFERENT BOW METHODS USING CONVENTIONAL, HAND-ENGINEERED FEATURES.

(a) VOC 2007

Methods	mAP (%)	Dim.
LLC [17]	57.6	32,768
SV [20]	58.2	655,360
MLCW+MKL [52]	57.5	32,768
H-VLAD [11]	61.2	491,520
Kobayashi [12]	63.8	524,288
LASC [13]	63.6	524,288
FV (SIFT) [16]	61.8	262,144
FV (SIFT+LCS) [16]	63.9	262,144
HO-LP (SIFT)	64.3	327,680
HO-LP (SIFT+LCS)	67.4	327,680

(b) Caltech-256

# of train	15	30	45	60
LLC [17]	34.4	41.2	45.3	47.7
MLCW+MKL [52]	35.2	40.1	44.9	47.9
GOLD [29]	–	43.9	–	49.4
MSSR [53]	38.8 (0.3)	45.7 (0.5)	49.8 (0.2)	52.8 (0.5)
Kobayashi [12]	41.8 (0.2)	49.8 (0.1)	54.4 (0.3)	57.4 (0.4)
LASC [13]	43.7 (0.4)	52.1 (0.1)	57.2 (0.3)	60.1 (0.3))
FV (SIFT) [16]	38.5 (0.2)	47.4 (0.1)	52.1 (0.4)	54.8 (0.4)
FV (SIFT+LCS) [16]	41.0 (0.3)	49.4 (0.2)	54.3 (0.3)	57.3 (0.2)
M-HMP [54]	42.7	50.7	54.8	58.0
HO-LP (SIFT)	46.0 (0.2)	52.5 (0.1)	57.3 (0.2)	60.1 (0.5)
HO-LP (SIFT+LCS)	49.9 (0.2)	57.0 (0.1)	61.7 (0.2)	64.7 (0.5)

(c) FMD

Method	Acc. (%)
Sharan et al. [55]	57.1
Kobayashi [12]	57.3 (0.9)
FV (SIFT) [56]	58.2 (1.7)
FV (SIFT+LCS) [56]	63.3 (1.9)
HO-LP (SIFT)	61.5 (1.9)
HO-LP (SIFT+LCS)	65.4 (1.6)

(d) SUN-397

# of train	5	10	20	50
Xiao et al. [27]	14.5	20.9	28.1	38.0
Kobayashi [12]	–	–	–	46.1 (0.1)
LASC [13]	19.4 (0.4)	27.3 (0.3)	35.6 (0.1)	45.3 (0.4))
FV (SIFT) [16]	19.2 (0.4)	26.6 (0.4)	34.2 (0.3)	43.3 (0.2)
FV (SIFT+LCS) [16]	21.1 (0.3)	29.1 (0.3)	37.4 (0.3)	47.2 (0.2)
HO-LP (SIFT)	21.9 (0.4)	29.9 (0.2)	37.6 (0.2)	47.1 (0.1)
HO-LP (SIFT+LCS)	25.7 (0.3)	34.6 (0.1)	42.9 (0.2)	51.4 (0.2)

Like HO-LP, FV-LP significantly outperforms LP due to the leverage of high-order statistics. With the same size of the image-level representations, HO-LP is slightly better ($+0.8\%$) than FV-LP on the small Scene-15, while outperforming FV-LP by a large margin ($+3.0\%$) on Caltech-256 which is much larger than Scene-15; as for efficiency, HO-LP only takes about half of time of FV-LP for processing one image. The performance of FV-LP slightly increases with growing of PCA dimension on both datasets, however, the computation cost as well as the storage cost significantly increase such that higher dimensions for FV-LP is prohibitive, particularly for large datasets (e.g. Caltech-256 or larger ones). These comparisons under exactly the same settings demonstrate HO-LP is superior to FV-LP in terms of both the recognition accuracy and the cost. The high-dimensional nature of FV makes it unfit for the local pooling paradigm.

D. Comparison with hand-engineered feature based methods

In this part we conduct experiments to compare HO-LP with various methods using conventional, hand-engineered features on four datasets. Particularly, we compare with FV using separate SIFT and a combination of SIFT and LCS features [16] by score level fusion (0.7*SIFT+0.3*LCS for all experiments). The LCS features are extracted in the same way as SIFT. The comparison results are presented in Table III. To make the comparison more clearly, we also report the dimension of the final image representation for each method on VOC 2007. As listed in the right column of Table III(a) that the dimension of HO-LP is comparable to FV, but less than

other coding methods, including H-VLAD [11], Kobayashi [12], LASC [13] and SV [20].

On VOC 2007, with only SIFT, the proposed HO-LP yields an accuracy of 64.3%, outperforming other high-order based methods, H-VLAD [11], LASC [13] and FV; it even has better performance than FV incorporating SIFT and LCS, and the Dirichlet-derived GMM Fisher kernel proposed by Kobayashi [12]. By combining SIFT and LCS, HO-LP produces 67.4% in accuracy, higher than H-VLAD incorporating supervised dictionary learning (65.1%). It should be mentioned that FV in [49] achieves 68.0% by using data augmentation which significantly improves performance but at several times extra computational cost. Finally, it can be seen that all higher-order based methods outperform lower-order based ones, including LLC [17], SV [20], and supervised pooling [52].

For material recognition on FMD, HO-LP (SIFT) yields an accuracy of 61.5%, which is much higher than FV (SIFT), the methods of Kobayashi [12] and Sharan et al. [55]. By combining SIFT and LCS, HO-LP outperforms FV by $\sim 2.1\%$.

On Caltech-256, HO-LP shows a clear advantage over its competitors. In particular, HO-LP using only SIFT yields much better performance than FV (SIFT+LCS) and Kobayashi [12] ($\sim 3.1\%$ on average), as well as LASC. On the other hand, HO-LP incorporating SIFT and LCS significantly improves the performance over HO-LP (SIFT) ($\sim 4.4\%$).

On SUN-397, HO-LP (SIFT) yields higher accuracy than Kobayashi [12], LASC [13] and performs much better than FV (SIFT). Note that HO-LP using only SIFT is comparable to FV incorporating SIFT and LCS. By combining SIFT and LCS, the performance of HO-LP has a further improvement of $\sim 4.3\%$ on average.

E. Comparison with CNN features based methods

This section compares the proposed HO-LP with state-of-the-art, CNN features based methods. As in Section IV-B, we also evaluated the parameters of HO-LP. We observed that their effects on HO-LP's behavior under the CNN features are similar to those under SIFT, which are therefore not reported here. We choose a dictionary of 512 atoms and 128 configuration bins which are slightly different from the case of SIFT. As in [7, 50, 59], we do not use spatial pyramid scheme (spatial bins) as it brings no improvement. The weight parameter α is set to 0.25. We report results on VOC 2007, FMD, Caltech-256 (60 training images per category) and SUN-397 (50 training images per category) in Table IV-D.

The proposed method is tested on two pre-trained CNN models, 8-layer VGG-M [49] and 19-layer VGG-VD [49]. In the top panel of Table IV-D, we present the results based on CNN models of no more than 8 layers and the middle panel includes the methods exploiting VGG-VD. We also compare with state-of-the-art results achieved by hybrid methods in the bottom panel, where multiple CNN models trained on different type of databases, or various coding methods, or features from different layers are combined to improve performance.

We first notice that HO-LP improves significantly by using CNN features, producing much higher accuracy on all datasets than by using hand-engineered, SIFT features, i.e., 11.8%$\sim$21.8% with VGG-VD and 7.5%$\sim$14.7% with VGG-M, respectively. We also see that no matter using a single VGG-M or VGG-VD model, HO-LP achieves very competitive performance on all benchmarks compared to the other methods based on the same or a similar model. Note that FV with data augmentations or fine-tuning produces higher accuracy [49], which are not listed here as the compared methods do not exploit such tricks.

As seen at the bottom panel of Table IV-D, hybrid methods can generally obtain better results than using one single model. By integrating FV (VGG-VD) and the responses of the fully-connected layer, [7] obtains 82.4% on FMD. Combining two CNN models (16-layer and 19-layer), [50] reports 89.7% on VOC 2007 and 86.2% on Caltech-256. Several recent hybrid methods [26, 61, 62], focusing on scene recognition problem, achieved significant improvements (73% by VSAD is state-of-the-art) by exploiting the CNN models trained on Places databases [57]. Furthermore, they all benefit from multiple, complementary CNN models pre-trained on Places dataset and ImageNet dataset, respectively. Furthermore, Xie et al. combines FV and LLC to encode features from both convolutional and fully connected layers. In contrast, we focus on a novel high-order encoding method, which is suitable for general classification tasks including object, scene and material classifications, while only using convolutional features outputted from a single CNN model pre-trained on ImageNet.

Note that it is not easy to make completely fair comparisons for all competing methods due to different parameter settings in CNN models, responses of different layers as features, etc. Nevertheless, our experiments show that the proposed HO-LP with CNN features is very promising, producing performance comparable to or better than state-of-the-art methods in general image classification tasks.

F. Results on large-scale ImageNet dataset

In the last part of experiments, with the experimental setting as described in Section IV-E, we evaluate the proposed HO-LP on ImageNet (ILSVRC 2012) dataset based on CNN models of 8-layer VGG-M and 19-layer VGG-VD. For efficiency, we reduce image-level representations to 4096 dimensions by PCA , and train softmax classifiers using stochastic gradient descent algorithm. The top1 and top5 errors on validation set are reported in Table V. Our results are similar to the VGG-M and VGG-VD models.

As far as we know, we are among the first who evaluate dictionary-based coding method using CNN features on large-scale ImageNet. We clarify that VGG-M and VGG-VD are trained on ImageNet with large scale training samples in end-to-end architectures, where feature learning, image represenation and classifier training are jointly optimized, and the same benchmark is used to evaluate. But in our method these stages are separated, independent of each other. It is worth mentioning that there are much more practical applications where such large scale training samples are unavailable and a pre-trained or fine-tuned CNN model has to be used. In these cases (such as FMD, Caltech-256, SUN-397, etc), local pooling can achieve great improvements over pre-trained or

TABLE IV
COMPARISON OF ACCURACY WITH STATE-OF-THE-ART, CNN-BASED METHODS.

CNN Model	Methods	FMD	VOC 2007	Caltech-256	SUN-397
CNNs (layers≤8)	Zhou et al. [57]	–	–	67.2 (0.3)	54.3 (0.1)
	Zeiler et al. [58]	–	–	74.2 (0.3)	–
	Liu et al. [45]	–	77.8	–	–
	Chatfield et al. [49]	–	77.0	77.0 (0.5)	–
	Gong et al. [47]	–	–	–	51.98
	Mandar et al. [59]	–	–	–	54.4 (0.3)
	Wu et al. [60]	–	–	–	58.1
	FV (VGG-M) [7]	73.5 (2.0)	76.4	–	–
	HO-LP (VGG-M)	76.5 (1.3)	80.1	78.5 (0.3)	58.9 (0.2)
VGG-VD	Simonyan et al. [50]	–	89.3	85.1 (0.3)	–
	Ms-DSP [46]	–	89.3	85.5 (0.1)	59.8 (0.5)
	FV (VGG-VD) [7]	79.8 (1.8)	84.9	–	–
	HO-LP (VGG-VD)	81.4 (1.4)	87.2	86.5 (0.2)	63.2 (0.1)
Hybrid	FV + FC [7]	82.4 (1.5)	–	–	–
	Simonyan et al. [50]	–	89.7	86.2 (0.3)	–
	VSAD [26]	–	–	–	73.0
	Xie et al. [61]	–	–	–	70.7 (0.2)
	Herranz et al. [62]	–	–	–	70.2

TABLE V
ERROR RATE (%) ON IMAGENET 2012.

	Top-1 error	Top-5 error
VGG-M [49]	36.9	15.5
HO-LP (VGG-M)	36.7	15.2
VGG-VD [50]	27.3	9.0
HO-LP (VGG-VD)	27.6	8.7

fine-tuned CNN models. This indicates local pooling is still important for visual object classification, although deep CNN models have achieved promising performance in large scale ImageNet classification. In addition, it is very interesting to implement the proposed HO-LP in an end-to-end manner and compare with state-of-the-art CNN architectures, which will be our future research.

V. CONCLUSIONS

We proposed a high-order local pooling method, called HO-LP, for image classification. It is different from and is therefore complementary to the existing high-order based methods. Our main contributions are summarized as follows.

- We extended the local pooling (LP) method [9] to handle the first- and second-order statistics. The proposed high-order method preserves the advantages of LP, i.e., only pooling similar features and using small dictionary, while significantly improves its performance.
- We studied how to encode Gaussians over a dictionary of Gaussians as visual words. As far as we know, we are among the first who touch the problem of encoding over the Gaussian manifold. We hope this work motivates the interests on processing data consisting of Gaussians.
- We made extensive experiments to evaluate the proposed HO-LP and compared with state-of-the-arts. Our experiments showed that HO-LP is very competitive and generalizes well to both the traditional, hand-engineered features and novel CNN features.

As in [9], we maintained a single dictionary of Gaussians for all configuration bins. An alternative is to learn a dictionary for each configuration bin, which makes pooling more local but may lead to very large dictionary. In future work we will study the problem of learning simultaneously the parameters of CNN, HO-LP and SVM in an end-to-end fashion, which may further improve the classification performance.

APPENDIX A
PROOF OF THEOREM 1

Recall that for matrix Lie group, the matrix exponential and logarithm are respectively defined by [35] $\exp(\mathbf{Y}) = \sum_{k=0}^{\infty} \frac{1}{k!}\mathbf{Y}^k$ and $\log(\mathbf{Q}) = \sum_{k=1}^{\infty} \frac{1}{k}(-1)^{k-1}(\mathbf{Q}-\mathbf{I})^k$, where $\mathbf{I}$ denotes the identity matrix. To prove Theorem 1, we first introduce the following proposition [63, Section 3]:

Proposition 1: Let $S(k)$ be the space of $k \times k$ real matrices whose eigenvalues have imaginary parts on the interval $(-\pi, \pi)$. let $\exp(S(k))$ be the image of $S(k)$ under matrix exponential.

(1) Any $k \times k$ real, invertible matrix $\mathbf{Q}$ with non-negative eigenvalues has unique matrix logarithm $\log(\mathbf{Q}) \in S(k)$,
(2) $\exp(S(k))$ is the space of real invertible matrices with non-negative eigenvalues and $\exp : S(k) \rightarrow \exp(S(k))$ is a diffeomorphism.

Let $g(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be a Gaussian with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and $\mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}$ be its PDLTAT matrix, where $\boldsymbol{\Sigma}$ has Cholesky factorization $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$ and $\mathbf{L}$ is a lower triangular matrix with positive diagonal entries $l_{jj}, j = 1, \cdots, n$. Let us consider the characteristic function of $\mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}$: $|\lambda\mathbf{I}' - \mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}| = \begin{vmatrix} \lambda\mathbf{I}-\mathbf{L} & \boldsymbol{\mu} \\ \mathbf{0} & \lambda-1 \end{vmatrix} = (\lambda-1)\prod_{j=1}^{n}(\lambda - l_{jj})$, $\mathbf{I}'$ and $\mathbf{I}$ are $(n+1)\times(n+1)$ and $n \times n$ identity matrix, respectively. Note that here we use the recursive property of the determinant [64, Section 1.4]. Hence, the eigenvalues of $\mathbf{P}_{\mathbf{L},\boldsymbol{\mu}}$ are 1, $l_{jj} > 0, j = 1, \cdots, n$. According to Proposition 1, $\log(\mathbf{P}_{\mathbf{L},\boldsymbol{\mu}})$ exists uniquely lying in Lie algebra $\mathfrak{g}(n+1)$. On the other hand, for any $\mathbf{P}_{\mathbf{X},\boldsymbol{t}} \in \mathfrak{g}(n+1)$, from the definition of matrix exponential, it is not difficult to know $\exp(\mathbf{P}_{\mathbf{X},\boldsymbol{t}}) = \exp\left(\left[\begin{smallmatrix} \mathbf{X} & \mathbf{t} \\ \mathbf{0} & 0 \end{smallmatrix}\right]\right) \in G(n+1)$. From Proposition 1, we conclude that $\exp$ is a diffeomorphism from $\mathfrak{g}(n+1)$ to $\mathbf{G}(n+1)$, and so its inverse $\log$ is a diffeomorphism as well. Finally, we have (after some manipulations)

$$\log\left(\begin{bmatrix}\mathrm{diag}(\sigma_i) & \boldsymbol{\mu}\\ \mathbf{0} & 1\end{bmatrix}\right)=\sum_{k=1}^{\infty}\frac{(-1)^{k-1}}{k}\left(\begin{bmatrix}\mathrm{diag}(\sigma_i) & \boldsymbol{\mu}\\ \mathbf{0} & 1\end{bmatrix}-\mathbf{I}'\right)^k$$

$$=\begin{bmatrix}\log\sigma_1 & & & \frac{\mu_1\log\sigma_1}{\sigma_1-1}\\ & \ddots & & \vdots\\ & & \log\sigma_n & \frac{\mu_n\log\sigma_n}{\sigma_n-1}\\ & & & 0\end{bmatrix}$$

REFERENCES

[1] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. CVPR*, 2006, pp. 2169–2178. 1, 3, 6, 8

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010. 6

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105. 1

[4] S. H. Khan, M. Hayat, M. Bennamoun, R. Togneri, and F. A. Sohel, "A discriminative representation of convolutional features for indoor scene recognition," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3372–3383, July 2016. 1

[5] L. Xie, Q. Tian, M. Wang, and B. Zhang, "Spatial pooling of heterogeneous features for image classification," *IEEE Trans. Image Process.*, vol. 23, no. 5, pp. 1994–2008, May 2014. 1

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, pp. 91–110, 2004. 1

[7] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proc. CVPR*, 2015, pp. 3828–3836. 1, 6, 7, 10, 11

[8] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proc. ICML*, 2010, pp. 111–118. 1

[9] Y.-L. Boureau, N. Le Roux, F. Bach, J. Ponce, and Y. LeCun, "Ask the locals: multi-way local pooling for image recognition," in *Proc. ICCV*, 2011, pp. 2651–2658. 1, 3, 7, 8, 9, 11

[10] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. CVPR*, 2009, pp. 1794–1801. 1, 5

[11] X. Peng, L. Wang, Y. Qiao, and Q. Peng, "Boosting VLAD with supervised dictionary learning and high-order statistics," in *Proc. ECCV*, 2014, pp. 660–674. 1, 2, 9, 10

[12] T. Kobayashi, "Dirichlet-based histogram feature transform for image classification," in *Proc. CVPR*, 2014, pp. 3278–3285. 1, 2, 9, 10

[13] P. Li, X. Lu, and Q. Wang, "From dictionary of visual words to subspaces: Locality-constrained affine subspace coding," in *Proc. CVPR*, 2015, pp. 2348–2357. 2, 9, 10

[14] Q. Wang, P. Li, L. Zhang, and W. Zuo, "Towards effective codebookless model for image classification," *Pattern Recognition*, 2016. 1, 2

[15] Y. Huang, Z. Wu, L. Wang, and T. Tan, "Feature coding in image classification: A comprehensive study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 493–506, 2014. 1

[16] J. Sánchez, F. Perronnin, T. Mensink, and J. J. Verbeek, "Image classification with the Fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, 2013. 1, 2, 3, 6, 8, 9

[17] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. CVPR*, 2010, pp. 3360–3367. 1, 5, 9, 10

[18] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *Proc. ICCV*, 2011, pp. 2486–2493. 1, 2

[19] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. CVPR*, 2010, pp. 3304–3311. 1

[20] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using super-vector coding of local image descriptors," in *Proc. ECCV*, 2010, pp. 141–154. 1, 9, 10

[21] X. Zhou, N. Cui, Z. Li, F. Liang, and T. S. Huang, "Hierarchical Gaussianization for image classification," in *Proc. ICCV*, 2009, pp. 1971–1977. 2, 3

[22] H. Nakayama, T. Harada, and Y. Kuniyoshi, "Global Gaussian approach for scene categorization using information geometry," in *Proc. CVPR*, 2010, pp. 2336–2343. 2, 3, 6, 8

[23] P. Li, Q. Wang, W. Zuo, and L. Zhang, "Log-Euclidean kernels for sparse representation and dictionary learning," in *Proc. ICCV*, 2013, pp. 1601–1608. 2

[24] L. Gong, T. Wang, and F. Liu, "Shape of Gaussians as feature descriptors," in *Proc. CVPR*, 2009, pp. 2366–2371. 2, 4, 5

[25] B. Klein, G. Lev, G. Sadeh, and L. Wolf, "Associating neural word embeddings with deep image representations using Fisher vectors," in *Proc. CVPR*, 2015, pp. 4437–4446. 2

[26] Z. Wang, L. Wang, Y. Wang, B. Zhang, and Y. Qiao. (2016) Weakly supervised patchnets: Describing and aggregating local patches for scene recognition. [Online]. Available: http://arxiv.org/abs/1609.00153 2, 10, 11

[27] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proc. CVPR*, 2010, pp. 3485–3492. 2, 6, 9

[28] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Free-form region description with second-order pooling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, pp. 1177–1189, 2015. 2

[29] G. Serra, C. Grana, M. Manfredi, and R. Cucchiara, "GOLD: Gaussians of local descriptors for image representation," *Comput. Vis. and Image Understand.*, vol. 134, pp. 22 – 32, 2015. 2, 9

[30] A. Agarwal and B. Triggs, "Hyperfeatures - multilevel local coding for visual recognition," in *Proc. ECCV*, 2006, pp. 30–43. 2

[31] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Fisher networks for large-scale image classification," in *Proc. NIPS*, 2013, pp. 163–171. 2

[32] N. Vasconcelos, P. Ho, and P. Moreno, "The Kullback-Leibler kernel as a framework for discriminant and localized representations for visual recognition," in *Proc. ECCV*, 2004, pp. 430–441. 2, 8

[33] M. Harandi and M. Salzmann, "Riemannian coding and dictionary learning: Kernels to the rescue," in *Proc. CVPR*, 2015, pp. 3926–3935. 2, 4, 8

[34] P. Li, Q. Wang, and L. Zhang, "A novel earth mover's distance methodology for image matching with Gaussian mixture models," in *Proc. ICCV*, 2013, pp. 1689–1696. 2, 3, 6, 8

[35] O. Tuzel, F. M. Porikli, and P. Meer, "Learning on Lie groups for invariant detection and tracking," in *Proc. CVPR*, 2008, pp. 1–8. 5, 11

[36] B. C. Hall, *Lie groups, Lie algebras, and representations: an elementary introduction*. Springer, 2015, vol. 222. 5

[37] F. Cheng, Z. Wang, and D. Li, "Extended shape of gaussian: Feature descriptor based on element set of matrix lie group," *Optik-International Journal for Light and Electron Optics*, vol. 124, no. 19, pp. 3806–3811, 2013. 5

[38] P. Li, Q. Wang, H. Zeng, and L. Zhang, "Local Log-Euclidean multivariate Gaussian descriptor and its application to image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2016. 5

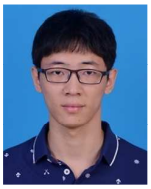
[39] V. Arsigny, O. Commowick, N. Ayache, and X. Pennec, "A fast and log-euclidean polyaffine framework for locally linear registration," *JMIV*, vol. 33, no. 2, pp. 222–238, 2009. 5

[40] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3. 6

[41] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Log-Euclidean metrics for fast and simple calculus on diffusion tensors," *Magnetic Resonance in Medicine*, vol. 56, pp. 411–421, 2006. 6
[42] L. Sharan, R. Rosenholtz, and E. Adelson, "Material perception: What can you see in a brief glance?" *J. Vis.*, vol. 9, no. 8, p. 784, 2009. 6
[43] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset." California Institute of Technology, 2007. 6, 8
[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015. 6
[45] L. Liu, C. Shen, and A. van den Hengel, "The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification," in *Proc. CVPR*, 2015, pp. 4749–4757. 6, 7, 11
[46] B. Gao, X. Wei, J. Wu, and W. Lin. (2015) Deep spatial pyramid: The devil is once again in the details. [Online]. Available: http://arxiv.org/abs/1504.05277 7, 11
[47] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *Proc. ECCV*, 2014, pp. 392–407. 6, 11
[48] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proc. ACM Multimedia*, 2010, pp. 1469–1472. 7
[49] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. BMVC*, 2014. 7, 10, 11
[50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ICLR*, 2015. 7, 10, 11
[51] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proc. ACM Multimedia*, 2015, pp. 689–692. 7
[52] S. R. Fanello, N. Noceti, C. Ciliberto, G. Metta, and F. Odone, "Ask the image: supervised pooling to preserve feature locality," in *Proc. CVPR*, 2014, pp. 851–858. 9, 10
[53] B.-D. Liu, Y.-X. Wang, B. Shen, Y.-J. Zhang, and M. Hebert, "Self-explanatory sparse representation for image classification," in *Proc. ECCV*, 2014, pp. 600–616. 9
[54] L. Bo, X. Ren, and D. Fox, "Multipath sparse coding using hierarchical matching pursuit," in *Proc. CVPR*, 2013, pp. 660–667. 9
[55] L. Sharan, C. Liu, R. Rosenholtz, and E. H. Adelson, "Recognizing materials using perceptually inspired features," *Int. J. Comput. Vis.*, vol. 103, no. 3, pp. 348–371, 2013. 9, 10
[56] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. CVPR*, 2014, pp. 3606–3613. 9
[57] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. NIPS*, 2014, pp. 487–495. 10, 11
[58] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833. 11
[59] M. Dixit, S. Chen, D. Gao, N. Rasiwasia, and N. Vasconcelos, "Scene classification with semantic fisher vectors," in *Proc. CVPR*, 2015, pp. 2974–2983. 10, 11
[60] R. Wu, B. Wang, W. Wang, and Y. Yu, "Harvesting discriminative meta objects with deep CNN features for scene classification," in *Proc. ICCV*, 2015, pp. 1287–1295. 11
[61] S. Y. C.-L. L. Guo-Sen Xie, Xu-Yao Zhang, "Hybrid cnn and dictionary-based models for scene recognition and domain adaptation," *IEEE Trans. Circuits Syst. Video Technol*, 2015. 10, 11
[62] L. Herranz, S. Jiang, and X. Li, "Scene recognition with cnns: objects, scales and dataset bias," in *Proc. CVPR*, 2016, pp. 571–579. 10, 11
[63] J. Gallier. (2013) Logarithms and Square Roots of Real Matrices. [Online]. Available: http://arxiv.org/abs/0805.0245 11
[64] M. Artin, *Algebra*. Prentice-Hall, Inc., 1991. 11

Peihua Li is a professor in School of Information and Communication Engineering, Dalian University of Technology. He received the PhD degree from Harbin Institute of Technology in 2002, and was awarded the honorary nomination of National Excellent Doctoral dissertation in 2005. He was supported by Program for New Century Excellent Talents in University of Ministry of Education of China in 2011. Currently he is mainly interested in image classification and search using theoretical and computational methods of information geometry. He has published over fifty papers in referred conferences and journals.

Hui Zeng received the M.Sc. degree from School of Information and Communication Engineering, Dalian University of Technology, China, in 2016. He is currently pursuing his Ph.D in the Dept. of Computing, The Hong Kong Polytechnic University, under the supervision of Prof. Lei Zhang. His research interests include computer vision, image and video processing and deep learning.



Qilong Wang is a PhD candidate in School of Information and Communication Engineering, Dalian University of Technology. His research interests include image and video classification and recognition. He has published several papers in top conferences including ICCV, CVPR and ECCV.

Simon C. K. Shiu obtained his degrees from City University of London, Newcastle Upon Tyne University and the Hong Kong Polytechnic University in 1985, 1986 and 1997 respectively. He had published over 100 papers and book chapters. He is now the Chairman of the Postgraduate Scheme in Computing in the department.

Lei Zhang (M'04, SM'14) received his B.Sc. degree in 1995 from Shenyang Institute of Aeronautical Engineering, Shenyang, P.R. China, and M.Sc. and Ph.D degrees in Control Theory and Engineering from Northwestern Polytechnical University, Xian, P.R. China, respectively in 1998 and 2001, respectively. His research interests include Computer Vision, Pattern Recognition, Image and Video Processing, and Biometrics, etc. Prof. Zhang has published more than 200 papers in those areas. As of 2017, his publications have been cited more than 25,000 times in the literature. Prof. Zhang is an Associate Editor of IEEE Trans. on Image Processing, SIAM Journal of Imaging Sciences and Image and Vision Computing, etc. He is a "Web of Science Highly Cited Researcher" selected by Thomson Reuters. More information can be found in his homepage http://www4.comp.polyu.edu.hk/ cslzhang/.