Sparse Coding Based Fisher Vector Using a Bayesian Approach

Kart-Leong Lim and Han Wang

***Abstract*—A recently proposed sparse coding based Fisher vector extends traditional GMM based Fisher Vector with a sparse term. Our experiments revealed that the addition of this sparse term alone significantly outperforms GMM based Fisher Vector by almost 20% improvement on small sized datasets (15-Scene, Caltech-10) and up to 5% improvement on medium sized datasets (MIT-67). In the original work, sparse coding based Fisher vector requires an off-the-shelf Sparse Coding solver. From a statistical perspective, an off-the-shelf solver may appear as a black-box. A more elegant way is to use a probabilistic model to learn Sparse Coding. We propose a probabilistic model known as sparse coding based GMM. It differs from GMM by an additional sparse coefficient hidden variable. The prior model of the sparse term is assumed Gaussian distributed for tractability. Inference of the model is performed by iteratively computing a set of closed-form solution obtained via variational method. Experimental results on several well-cited datasets show that our probabilistic based solver obtained on-par learning performance to an off-the-shelf solver as far as sparse coding based Fisher vector is concerned.**

***Index Terms*—Fisher vector, Gaussian mixture model, sparse coding, variational inference.**

I. Introduction

IN THE era of deep learning [1], Fisher Vector (FV) [2], [3] remains one of the contemporary method in image categorization and retrieval for a couple of good reasons. One of the few reasons is the low computation cost involved and it does not require large amount of training dataset by comparison. Some authors [4]–[8] have also looked into complementing FV with deep learning. FV represents an image by sum-pooling the gradients of GMM parameters. The main advantage of such representation is comparative or better performance to a Bag-of-Word vector while keeping the GMM computation cost low. Because FV computes a high dimensional representation, VLAD [9] is proposed to reduce FV's computation cost by keeping the model parameters and dimension to a minimal. The Bayesian approach to GMM defines additional prior distributions [10]. Cinbis *et al.* [5] took this opportunity to further extend FV's model parameters with additional hyperparameters. More recently, Liu *et al.* [11] replace the GMM model in FV with a Sparse Coding model [12]. This finding led to the interest in this paper.

Three issues remain with FV. First, when GMM is used to model a high dimensional feature vector (e.g., 4096-dim), it will lead to poor fitting. Even as the number of mixture increases, the fit does not improve much [11]. Second, very few works have looked into using a different model to improve the performance of FV. A unique case is Liu *et al.*'s Sparse Coding based Fisher Vector (SC-FV) [11], which leads to the third issue. In Liu *et al.*'s approach, an off-the-shelve solver [13] is required. Most Sparse Coding algorithms only consider the problem as a convex optimization [13], [14]. From a nonprobabilistic perspective, it is not easy to see Sparse Coding's relationship with FV [11]. Our work is strongly influenced by Liu *et al.*'s work [11]. Although, SC-FV was originally proposed to solve the first and second problem, i.e., address poor fitting and a new model for FV, our work mainly addresses the third problem, i.e., using a probabilistic solver for Sparse Coding.

Unlike most Sparse Coding algorithms which consider the problem as a convex optimization, we treat the problem as a probabilistic model based on GMM. We call the model a Sparse Coding based GMM (sGMM). Our major contribution is that we propose an algorithm based on variational method to approximate the hidden variables of sGMM. Then, we use the learnt sGMM to compute a SC-FV. Empirically results show that this SC-FV significantly improves the performance over GMM based Fisher Vector. In comparison with convex optimization solver, using a variational method achieves similar learning performance as far as SC-FV is concerned.

II. Background

A. GMM Based Fisher Vector (GMM-FV)

For a set of N observed feature vectors denoted as $x = \{x_n\}_{n=1}^{N} \in \mathbb{R}^D$, GMM models x with parameters $\theta = \{B, \Sigma, z, \pi\}$. We define the covariance as $\Sigma_k = \sigma_k^2 I$, $\sigma_k^2 =$ constant. The cluster assignment is denoted $z = \{z_n\}_{n=1}^{N}$ where z_n is a $1-of-K$ binary vector, $\sum_{k=1}^{K} z_{nk} = 1$, and $z_{nk} \in \{0, 1\}$. The basis vector (or cluster mean) is denoted $B = \{B_k\}_{k=1}^{K} \in \mathbb{R}^D$ and the mixture component is denoted $\pi = \{\pi_k\}_{k=1}^{K}$ where $0 \leq \pi_k \leq 1$ and is subjected to $\sum_{k=1}^{K} \pi_k = 1$. The feature vector dimensions and number of mixtures are denoted D and K, respectively.

In a Bayesian approach, each GMM hidden variable is modeled with a prior distribution. The joint distribution of GMM can be factorized as [10]

$$p(x, B, z, \pi) = p(x \mid B, z)p(B)p(z \mid \pi)p(\pi). \tag{1}$$

The random variables are modeled as follows [10]:

$$\begin{aligned} x \mid B, z &\sim \mathcal{N}(B)^z \\ z \mid \pi &\sim \text{Mult}(\pi) \\ B &\sim \mathcal{N}\left(B_0, \gamma^{-1}\right) \\ \pi &\sim \text{Dir}(\alpha_0) \end{aligned} \tag{2}$$

Manuscript received October 26, 2016; revised December 1, 2016; accepted December 5, 2016. Date of publication December 7, 2016; date of current version January 5, 2017. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Magno T. M. Silva.

The authors are with the Nanyang Technological University, Singapore 639798, Singapore (e-mail: lkartl@yahoo.com.sg; hw@ntu.edu.sg).

Digital Object Identifier 10.1109/LSP.2016.2636900

where $\mathcal{N}$, Mult, and Dir are the Gaussian, Multinomial, and Dirichlet distributions, respectively. Due to page limit, for more details on the Bayesian approach in the above expressions, we refer the reader to [10].

In its simplest form, GMM-FV is defined as the gradient of $\ln p(x,\theta)$ over B only [3], [9]. Although FV [2] was originally defined using a Maximum Likelihood approach, we can alternatively use a Maximum a Posteriori (MAP) approach here. Both approaches are identical when we assume noninformative prior for B as follows:

$$\begin{aligned} v_k &= \sum_{n=1}^{N} \nabla_{B_k} \ln p(x_n, B_k, z_{nk}, \pi_k) \\ &= \sum_{n=1}^{N} \nabla_{B_k} \left(\ln p(x_n \mid B_k, z_{nk}) + \ln p(B_k) + \text{const.}\right) \\ &= \sum_{n=1}^{N} \left(x_n - B_k\right) z_{nk}. \end{aligned} \tag{3}$$

In the above, when maximizing $\ln p(x,\theta)$, the factorial terms of $\ln p(x,\theta)$ not related to B are absorbed into a constant.

To further simplify FV, for a shared covariance assumption, i.e., $\sigma_k = 1$ and $\sum_{n=1}^{N} z_{nk} = \frac{1}{K}$, each cluster centered at B_k has equal sample size. Thus, we can treat z_{nk} as a constant and we obtain GMM-FV as follows:

$$v_k = \sum_{n=1}^{N} \left(x_n - B_k\right) \tag{4}$$

The training of B_k require offline learning using an approximation such as the EM algorithm or variational Bayes [10].

B. Sparse Coding Based Fisher Vector (SC-FV)

The probabilistic model of Sparse Coding is [12], [11]

$$\begin{aligned} p(x) &= \int p(x \mid u, B) p(u) du \\ x &\sim \mathcal{N}\left(Bu\right),\ p(u) \propto \exp\left(-\Omega\left|u\right|\right). \end{aligned} \tag{5}$$

The sparse coefficient u is drawn from a zero mean Laplace distribution while x is drawn from a Gaussian distribution with mean Bu.

Instead of computing the integral, Liu *et al.* approximate $p(x)$ using a MAP estimate [11]:

$$\begin{aligned} p(x) &\approx p(x \mid u^*, B) p(\hat{u}) \\ u^* &= \arg\max_u \ \ln p(x \mid u, B) + \ln p(u). \end{aligned} \tag{6}$$

Similarly to GMM-FV earlier, a SC-FV [11] is defined as the gradient of $p(x)$ over B:

$$v = \sum_{n=1}^{N} \nabla_B \ln p(x_n) \ = \sum_{n=1}^{N} \left(x_n - B u_n^*\right) u_n^* \tag{7}$$

In SC-FV, u^* and B are solved by using an off-the-shelf sparse coding solver [13].

III. Methodology

A. Sparse Coding Based GMM (sGMM)

We now present a new GMM model known as the sGMM as seen in Fig. 1. We define the hidden variables as

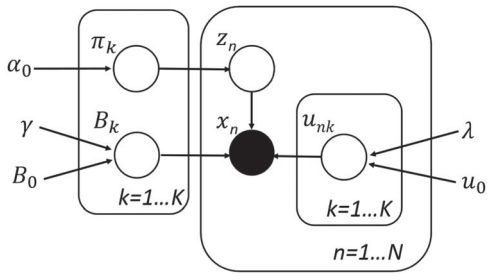

Fig. 1. Probabilistic model of sGMM

$\phi = \{z, u, B, \pi\}$. We denote the sparse coefficient as $u = \{u_n\}_{n=1}^{N} \in \mathbb{R}^K$. The joint distribution of the observed and hidden random variables $p(x,\phi)$ is

$$p(x, z, u, B, \pi) = p(x \mid B, u, z) p(u) p(B) p(z \mid \pi) p(\pi). \tag{8}$$

We use a Gaussian distributed prior to model u. Using Bayesian approach [10], the probability models of $p(x,\phi)$ are further defined as:

$$\begin{aligned} x \mid u, B, z &\sim \mathcal{N}\left(Bu\right)^z \\ B &\sim \mathcal{N}\left(B_0, \gamma^{-1}\right) \\ u &\sim \mathcal{N}\left(u_0, \lambda^{-1}\right) \\ z \mid \pi &\sim \text{Mult}\left(\pi\right) \\ \pi &\sim \text{Dir}\left(\alpha_0\right). \end{aligned} \tag{9}$$

The main difference between sGMM and GMM is the additional term, $u \sim \mathcal{N}\left(u_0, \lambda^{-1}\right)$. Whereas, the main difference between sGMM and Sparse Coding is due to the fact that $u_n \in \mathbb{R}^K$ is a $1-\text{of}-K$ vector and has a cardinality constraint, $\text{Card}\left(u_n\right) = 1$. It means that there is only one nonzero element in u_n while the rest of the elements are zero [15]. Also, in our sGMM model, we did not normalize the summation of the absolute value of each element to 1, $|u_n| \neq 1$.

B. Variational Inference of sGMM

We now turn to the model inference of $p(x,\phi)$. Our main goal is learning B from dataset since FV rely on B. We apply the variational method [10] to infer sGMM. For tractability, we use the following mean field factorization

$$q(z, u, B, \pi) = q(z) q(u, B, \pi). \tag{10}$$

The variational log posterior can be obtained by the form [10]:

$$\ln q(\phi_j) = E_{i \neq j}\left[\ln p(X, \phi_j)\right] + \text{const.} \tag{11}$$

Using the two expressions above, the variational log posterior distribution for the hidden variables can be obtained as follows:

$$\begin{aligned} \ln q(u) &= E_z\left[\ln p(x \mid u, B, z) + \ln p(u)\right] + \text{const.} \\ \ln q(B) &= E_z\left[\ln p(x \mid u, B, z) + \ln p\left(B\right)\right] + \text{const.} \\ \ln q(\pi) &= E_z\left[\ln p\left(z \mid \pi\right) + \ln p\left(\pi\right)\right] + \text{const.} \\ \ln q(z) &= E_{u,B,\pi}\left[\ln p(x \mid u, B, z) + \ln p(z \mid \pi)\right] + \text{const.} \end{aligned} \tag{12}$$

As variational posteriors cannot be learnt analytically, approximation techniques such as variational Expectation-Maximization is used [10].

C. Variational Posterior Learning

We further decompose the variational distribution as

$$q(u, B, \pi) = q(u)q(B)q(\pi). \tag{13}$$

By taking MAP estimation for the variational log posterior distributions, closed-form expression can be obtained

$$\hat{u_{nk}} = \arg\max_{u_{nk}} \ln q(u_{nk}) = \frac{B_k x_n E[z_{nk}] + \lambda u_0}{B_k^2 E[z_{nk}] + \lambda} \tag{14}$$

$$\hat{B}_k = \arg\max_{B_k} \ln q(B_k) = \frac{\sum_{n=1}^{N} u_{nk} x_n E[z_{nk}] + \gamma B_0}{\sum_{n=1}^{N} u_{nk}^2 E[z_{nk}] + \gamma} \tag{15}$$

$$\hat{\pi_k} = \arg\max_{\pi_k} \ln q(\pi_k) \text{ s.t. } \sum_k \pi_k = 1$$

$$= \frac{\sum_{n=1}^{N} E[z_{nk}] + (\alpha_0 - 1)}{\sum_{j=1}^{K}\sum_{n=1}^{N} E[z_{nj}]\,\zeta}. \tag{16}$$

ζ is a Lagrange multipler due to the constraint $\sum_k \pi_k = 1$. The hyperparameters $\lambda, u_0, \gamma, B_0, \alpha_0$ are empirically determined. More importantly, we have to solve $E[z_{nk}]$ before computing the above estimation.

The full expression for the inference of z is given by

$$E[z_{nk}] \propto \underset{u,B,\pi}{E} [\ln p(x \mid u, B, z_k = 1) + \ln p(z_k = 1 \mid \pi)]$$

$$= \exp\left(\frac{\ln(E[\pi_k]) - \frac{1}{2}(x_n - E[B_k]E[u_{nk}])^2)}{\sum_{j=1}^{K} \ln(E[\pi_j]) - \frac{1}{2}(x_n - E[B_j]E[u_{nj}])^2}\right). \tag{17}$$

Instead of defining moments for variational EM, we use MAP approximation for the expected values as follows:

$$E[\pi_k] \approx \hat{\pi_k}$$
$$E[B_k] \approx \hat{B}_k$$
$$E[u_{nk}] \approx \hat{u_{nk}}. \tag{18}$$

The proposed algorithm of learning sGMM is summarized in Algorithm 1.

D. sGMM Based Fisher Vector (sGMM-FV)

Taking the MAP of $p(x, \phi)$ over B, the sGMM-FV is computed as

$$v_k = \sum_{n=1}^{N} \nabla_{B_k} \ln p(x_n, z_{nk}, B_k, u_{nk}, \pi_k)$$

$$= \sum_{n=1}^{N} \nabla_{B_k} (\ln p(x_n \mid B_k, u_{nk}, z_{nk}) + \ln p(B_k) + \text{const.})$$

$$= \sum_{n=1}^{N} z_{nk}(x_n - B_k u_{nk}) u_{nk} - \gamma(B_k - B_0). \tag{19}$$

The difference between sGMM-FV and SC-FV is the presence of prior belief γ, hyperparameters B_0 and cluster assignment z_{nk}. If we set assumption $\sum_{n=1}^{N} z_{nk} = \frac{1}{K}$ and $\gamma = 0$ to the above expression, we obtain an identical form to SC-FV.

Algorithm 1: Training Sparse Coding based GMM (sGMM).

Input: x
Output: $\hat{B}, \hat{u}$
Initialization: $K, u_{nk}, E[z_{nk}], B_0, \gamma, \alpha_0, u_0, \lambda, \zeta$
For each iteration:
1) compute the cluster mean/basis

$$\hat{B}_k = \frac{\sum_{n=1}^{N} u_{nk} x_n E[z_{nk}] + \gamma B_0}{\sum_{n=1}^{N} u_{nk}^2 E[z_{nk}] + \gamma}$$

2) update the expected values

$$E[\pi_k] = \hat{\pi_k}$$
$$E[B_k] = \hat{B}_k$$
$$E[u_{nk}] = \hat{u_{nk}}$$

$$E[z_{nk}] = \exp\left(\frac{\ln(E[\pi_k]) - \frac{1}{2}(x_n - E[B_k]E[u_{nk}])^2}{\sum_{j=1}^{K} \ln(E[\pi_j]) - \frac{1}{2}(x_n - E[B_j]E[u_{nj}])^2}\right)$$

3) compute the mixture component and sparse coefficient

$$\hat{\pi_k} = \frac{\sum_{n=1}^{N} E[z_{nk}] + (\alpha_0 - 1)}{\sum_{j=1}^{K}\sum_{n=1}^{N} E[z_{nj}]\,\zeta}$$

$$\hat{u_{nk}} = \frac{\hat{B}_k x_n E[z_{nk}] + \lambda u_0}{\left(\hat{B}_k\right)^2 E[z_{nk}] + \lambda}$$

End

IV. Experiments

Dataset: We evaluate the performance of proposed method on Multiview Car [16], Caltech10, Scene Categories, and MIT Indoor. When coding an image from raw feature (e.g., pixel or HoG) to FV, we extract patches from the entire image. This is for both case of train and test images.

Image Feature: We only use a single-scale HoG as the feature. For each image patch, we uniformly extract image patch with 16×16 pixels every eight pixels. Each patch is used to compute a 2×2 partitioned HOG outputting a 1×32 descriptor with L2 normalization.

sGMM Initialization: We use the following initializations $u_{nk} = 1, B_0 = 0, \gamma = 0, K \in \{64, 128\}, \alpha_0 = 1, u_0 = 0, \lambda = 1, \zeta = 1, E[z_{nk}] \sim \text{Mult}(\pi), \pi_k = \frac{1}{K}$.

SC Initialization: For SC, we use $\Omega = 0.4$ for all experiments.We use batch size of 1000 samples and default setting for the rest of the parameters in [13].

sGMM Practical Implementation: Instead of computing the expected value $E[z_{nk}]$, we use the following to compute $E[z] \approx \hat{z} = \arg\max_z \ln q(z)$.

Iterations and Cluster Size: We use 40 iterations for all training, as we usually obtain very minor changes on the results of all methods after 10 iterations. We use cluster size of $K \in \{64, 128\}$ to run each method. Usually, there is negligible difference in FV improvement when increasing the cluster size beyond $K = 64$.

Fisher Vector: Kmeans, GMM, SC, and sGMM are used to train their respective variable B. Both SC and sGMM further

TABLE I
COMPARISION OF BASELINES AND PROPOSED METHODS ON MULTIVIEW CAR (EIGHT CLASSES)

	Kmeans-FV	GMM-FV	SC-FV	sGMM-FV
$K = 64$	0.57404	0.57356	**0.72426**	0.7163
$K = 128$	0.57486	0.57709	0.71346	**0.71657**

TABLE II
COMPARISION OF BASELINES AND PROPOSED METHODS ON CALTECH (TEN CLASSES)

	Kmeans-FV	GMM-FV	SC-FV	sGMM-FV
$K = 64$	0.44432	0.44463	0.55736	**0.6258**
$K = 128$	0.44396	0.44205	0.55525	**0.62676**

TABLE III
COMPARISION OF BASELINES AND PROPOSED METHODS ON SCENE CATEGORIES (15 CLASSES)

	Kmeans-FV	GMM-FV	SC-FV	sGMM-FV
$K = 64$	0.44565	0.44595	**0.66347**	0.65703
$K = 128$	0.44701	0.44626	0.66528	**0.66591**

TABLE IV
COMPARISION OF BASELINES AND PROPOSED METHODS ON MIT INDOOR (67 CLASSES)

	Kmeans-FV	GMM-FV	SC-FV	sGMM-FV
$K = 64$	0.14372	0.14251	0.19556	**0.19634**
$K = 128$	0.14270	0.14623	0.19666	**0.20807**

computes $\hat{u}$. The learnt variables B and/or $\hat{u}$ are used to compute FV for each baseline and the proposed method.

For Kmeans and GMM, we compute GMM-FV. We use the implementation in [17] to train B for GMM and Kmeans. For SC and sGMM, we compute SC-FV. We use the feature-sign algorithm in [13] to learn both variables for SC-FV. For sGMM-FV, we use Algorithm 1 to learn B and we initialize $u_{nk} = 1$ to compute $E[z_{nk}]$ and then compute $\hat{u}$. The hyperparameter values are used $u_0 = 0, \lambda = 1$ and $\sum_{n=1}^{N} z_{nk} = \frac{1}{K}$. Essentially, we treat sGMM-FV as SC-FV to avoid making any decision for the hyperparameter values. The only discrimination we use for sGMM-FV and SC-FV is the way their parameters are computed.

Each FV is further normalized by its $L2$ norm, $v_k = \frac{v_k}{\|v_k\|_2}$. Finally, FV coding is the concatenation, $V = [\cdots v_k \cdots]^T$

Classification and Evaluation: Linear SVM is used for classification on all methods. Classification performance is measured in mean Average Precision (mAP) [9]. We perform five reruns and average the mAP results for each method. The performance of the methods for different cluster sizes are shown in Tables I–IV.

Multiview Car: This is the smallest dataset, using 20 K features for unsupervised learning. Object bounding box is available. There are eight bin-pose view classes in total and 400 train images and 400 test images using the setup in [16]. We observe that both SC-FV and sGMM-FV are comparative and they significantly outperform both GMM and Kmeans by a huge 15% improvement in terms of mAP.

Caltech10: In the next dataset, we also use 20 K features for unsupervised learning. Object bounding box is available. We use the setup in [17] for 10 object classes and 400 train images and 2644 test images. In this dataset, our proposed method sGMM-FV significantly outperformed all baseline methods with 18% versus Kmeans-FV, GMM-FV and 7% versus SC-FV. The poorer performance of SC-FV than sGMM-FV is possible due to insufficient training data when using a convex solver for Sparse Coding.

Scene Categories: This is a well-known dataset for outdoor scene recognition. The entire image is used as there is no bounding box information given. There are 75 000 feature for unsupervised learning from 15 classes, and there are 1500 train and 2985 test images [18]. Both SC-FV and sGMM-FV outperformed both Kmeans-FV and GMM-FV by a wide margin of 22%.

MIT Indoor: This is another recently used and well-known dataset for both indoor and outdoor scene recognition. The entire image is used as there is no bounding box information given. There are 107 200 feature for unsupervised learning from 67 classes, and there are 5360 train and 1340 test images [19]. Both SC-FV and sGMM-FV outperformed both Kmeans-FV and GMM-FV by a gain of 5%.

Experiment Conclusion: From the experimental results we conclude that sGMM-FV is at least on-par with SC-FV despite having a cardinality constraint on variable u. This is within our expectation since sGMM-FV is solving identical problem to SC-FV abeit using a probabilistic approach. Both methods also significantly outperforms nonsparse coefficient enabled FV learners such as GMM and Kmeans consistently for all datasets.

Although not shown in the experimental results, the computation time for SC-FV and sGMM-FV are on par but both are much slower than Kmeans-FV and GMM-FV due to the additional sparse coefficient involved. Typically, for SC-FV and sGMM-FV, we require overnight training for both Scene Categories and MIT Indoor dataset.

V. CONCLUSION AND FUTURE WORKS

A recently proposed SC-FV is able to achieved much better image representation performance than a traditional GMM-FV. This is due to the presence of a sparse coefficient term which enforces good reconstruction of signal. But, because the authors did not propose a solution to SC-FV so an off-the-shelf Sparse Coding solver is suggested.

In this work, we Propose a new solution based on probabilistic modeling for such a purpose. We call our model the sGMM. It is similar to GMM and mainly differs by an additional hidden variable for representing the sparse term. We perform model inference by using variational method to obtain analytical solution to the model's hidden variables.

Experimental results on several image classification datasets show that SC-FV significantly outperformed a GMM-FV. Furthermore, from a SC-FV perspective, the learning performance of our probabilistic solver is at least on-par with off-the-shelf solver.

At the moment, we have only used the gradient of the basis variable B, for computing a SC-FV. In future, it would be interesting to find out how what performance would arise from extending the SC-FV to more than one parameter.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Proc. Syst.*, 2012, pp. 1097–1105.

[2] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

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

[4] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 392–407.

[5] R. G. Cinbis, J. Verbeek, and C. Schmid, "Approximate Fisher kernels of non-iid image models for image categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1084–1098, Jun. 2016.

[6] F. Perronnin and D. Larlus, "Fisher vectors meet neural networks: A hybrid classification architecture," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3743–3752.

[7] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Fisher networks for large-scale image classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 163–171.

[8] V. Sydorov, M. Sakurada, and C. H. Lampert, "Deep Fisher kernels-end to end learning of the Fisher kernel GMM parameters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1402–1409.

[9] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3304–3311.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[11] L. Liu, C. Shen, L. Wang, A. van den Hengel, and C. Wang, "Encoding high dimensional local features by sparse coding based Fisher vectors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1143–1151.

[12] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, 1997.

[13] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 801–808.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[15] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 1794–1801.

[16] M. Ozuysal, V. Lepetit, and P. Fua, "Pose estimation for category specific multiview object localization," in *Proc. Comput. Vis. Pattern Recognit.*, 2009, pp. 778–785.

[17] K.-L. Lim, H. Wang, and X. Mou, "Learning Gaussian mixture model with a maximization-maximization algorithm for image classification," in *Proc. 12th IEEE Int. Conf. Control Autom.*, 2016, pp. 887–891.

[18] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, vol. 2, pp. 2169–2178.

[19] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 413–420.