Discriminative Learned Dictionaries for Local Image Analysis

Julien Mairal[1,5] Francis Bach[1,5] Jean Ponce[2,5] Guillermo Sapiro[3] Andrew Zisserman[2,4,5]

[1]INRIA [2]Ecole Normale Supérieure [3]University of Minnesota [4]Oxford University

Abstract

Sparse signal models have been the focus of much recent research, leading to (or improving upon) state-of-the-art results in signal, image, and video restoration. This article extends this line of research into a novel framework for local image discrimination tasks, proposing an energy formulation with both sparse reconstruction and class discrimination components, jointly optimized during dictionary learning. This approach improves over the state of the art in texture segmentation experiments using the Brodatz database, and it paves the way for a novel scene analysis and recognition framework based on simultaneously learning discriminative and reconstructive dictionaries. Preliminary results in this direction using examples from the Pascal VOC06 and Graz02 datasets are presented as well.

1. Introduction

Sparse representations have recently drawn much interest in signal, image, and video processing. Under the assumption that natural images admit a sparse decomposition in some redundant basis (or so-called *dictionary*), several such models have been proposed, *e.g.*, curvelets, wedgelets, bandlets and various sorts of wavelets [21]. Recent publications have shown that learning non-parametric dictionaries for image representation instead of using off-the-shelf ones, can significantly improve image restoration, *e.g.*, [6, 30, 31, 38].

Consider a signal $\mathbf{x}$ in $\mathbb{R}^n$. We say that $\mathbf{x}$ admits a sparse approximation over a dictionary $\mathbf{D}$ in $\mathbb{R}^{n\times k}$, composed of k elements (atoms), when we can find a linear combination of a "few" atoms from $\mathbf{D}$ that is "close" to the original signal $\mathbf{x}$. A number of practical algorithms have been developed for learning such dictionaries like the K-SVD algorithm [2] and the method of optimal directions (MOD) [7]. This approach has led to several restoration algorithms, which equal or exceed the state of the art in tasks such as image and video denoising, inpainting, demosaicing [6, 19], and texture synthesis [27]. Alternative models that learn image representations can be found in [10, 12].

The computer vision community has also been interested in extracting sparse information from images for recognition, *e.g.*, by designing local texture models [15, 37], which have proven to be discriminative enough to be used in image segmentation [18, 33]. Introducing learning into the feature extraction task has been part of the motivation for some recent works: *e.g.*, in [29], image features are learned using convolutional neural networks; while in [13, 39], discriminative strategies for learning visual codebooks for nearest-neighbor search are presented.

Sparse decompositions have also been used for face recognition [40, 41], signal classification [8, 9] and texture classification [14, 27, 34]. Interestingly, while discrimination is the main goal of these papers, the optimization (dictionary design) is purely generative, based on a criteria which does not explicitly include the actual discrimination task, which is one of the key contributions of our work.

The framework introduced in this paper addresses the learning of multiple dictionaries which are simultaneously reconstructive and discriminative, and the use of the reconstruction errors of these dictionaries on image patches to derive a pixelwise classification. The novelty of the proposed approach is twofold: First, redundant non-parametric dictionaries are learned, in contrast with the more common use of predefined features and dictionaries [9, 40, 41]. Second, the sparse local representations are learned with an explicit discriminative goal, making the proposed model very different from traditional reconstructive ones [8, 27, 34]. We illustrate the benefits of this approach in a texture segmentation task on the Brodatz dataset [28] for which it significantly improves over the state of the art [16, 17, 34], and also present preliminary results showing that it can be used to learn discriminative key patches from the Pascal VOC06 [26] database, and perform the weakly supervised form of feature selection advocated in [25, 36] on images from the Graz02 dataset [24].

Section 2 presents the classical framework for learning dictionaries for the sparse representation of overlapping image patches. The discriminative framework is introduced in Section 3, along with the corresponding optimization procedure. Section 4 is devoted to experimental results and applications, and Section 5 concludes this paper.

[5]WILLOW project-team, Laboratoire d'Informatique de l'Ecole Normale Supérieure, ENS/INRIA/CNRS UMR 8548.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

2. Dictionary learning for reconstruction

2.1. Learning reconstructive dictionaries

We now briefly describe for completeness the K-SVD [2] and MOD [7] algorithms for learning dictionaries for natural images. Since images are usually large, these techniques are designed to work with overlapping patches instead of whole images (one patch centered at every pixel). We denote by n the number of pixels per patch, and write patches as vectors $\mathbf{x}$ in $\mathbb{R}^n$. Learning an over-complete dictionary with a fixed number k of atoms, that is adapted to M patches of size n from natural images, and with a sparsity constraint (each patch has less than L atoms in its decomposition), is addressed by solving the following minimization problem:

$$\min_{\boldsymbol{\alpha},\mathbf{D}} \sum_{l=1}^{M} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l||_2^2 \text{ s.t. } ||\boldsymbol{\alpha}_l||_0 \leq L. \tag{1}$$

In this equation, $\mathbf{x}_l$ is an image patch written as a column vector. $\mathbf{D}$ in $\mathbb{R}^{n\times k}$ is a dictionary to be learned, each of its atoms (columns) is a unit vector in the ℓ_2 norm. The problem is to find the optimal dictionary that leads to the lowest reconstruction error given a fixed sparsity factor L. The vector $\boldsymbol{\alpha}_l$ in $\mathbb{R}^k$ is the sparse representation for the l-th patch using the dictionary $\mathbf{D}$. $||\mathbf{x}||_0$ denotes the "ℓ_0-norm" of the vector $\mathbf{x}$, which is not a formal norm, but a measure of sparsity counting the number of non-zero elements in a vector. Then one can introduce

$$\begin{aligned} \boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D}) &\equiv \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^k} ||\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}||_2^2, \text{ s.t. } ||\boldsymbol{\alpha}||_0 \leq L, \\ \mathcal{R}(\mathbf{x},\mathbf{D},\boldsymbol{\alpha}) &\equiv ||\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}||_2^2, \\ \mathcal{R}^\star(\mathbf{x},\mathbf{D}) &\equiv ||\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})||_2^2. \end{aligned} \tag{2}$$

$\mathcal{R}^\star(\mathbf{x},\mathbf{D})$ represents the best representation error of $\mathbf{x}$ on $\mathbf{D}$ with a sparsity factor L (number of coefficients in the representation), in terms of ℓ_2-norm. It will be used later in this paper as a discriminative function.

Both K-SVD and MOD are iterative approaches designed to minimize the energy (1). First, a dictionary $\mathbf{D}$ in $\mathbb{R}^{n\times k}$ is initialized, *e.g.*, from random patches of natural images. Then the main loop is composed of two stages:

• *Sparse coding:* During this step, $\mathbf{D}$ is fixed and one addresses the NP-hard problem of finding the best decomposition of each patch l:

$$\boldsymbol{\alpha}_l = \boldsymbol{\alpha}^\star(\mathbf{x}_l,\mathbf{D}). \tag{3}$$

A greedy orthogonal matching pursuit [22] is used, which has been shown to be very efficient [35], although being theoretically suboptimal.[1]

• *Dictionary update:* Here is the only difference between K-SVD and MOD. In the case of MOD, the decompositions $\boldsymbol{\alpha}_l$ are fixed and a least-squares problem is solved updating all the atoms simultaneously:

[1]Other strategies can be employed like convexification via the replacement of the ℓ_0-norm by the ℓ_1-norm (basis pursuit approach [5]).

Input: $\mathbf{D} \in \mathbb{R}^{n\times k}$ (dictionary from the previous iteration); M vectors $\mathbf{x}_l \in \mathbb{R}^n$ (input data); $\boldsymbol{\alpha} \in \mathbb{R}^{k\times M}$ (coefficients from the previous iteration).
Output: $\mathbf{D}$ and $\boldsymbol{\alpha}$.
Loop: For $j = 1 \dots k$, update $\mathbf{d}$, j-th column of $\mathbf{D}$,
• Select the set of patches that use $\mathbf{d}$:

$$\omega \leftarrow \{l \in 1,\dots,M \text{ s.t. } \boldsymbol{\alpha}_l[j] \neq 0\}. \tag{5}$$

• For each patch $l \in \omega$, compute the residual of the decomposition of $\mathbf{x}_l$: $\mathbf{r}_l = \mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l$.
• Compute a new atom $\mathbf{d}' \in \mathbb{R}^n$ and the associated coefficients $\boldsymbol{\beta} \in \mathbb{R}^{|\omega|}$ that minimize the residual error on the selected set ω, using a truncated SVD:

$$\min_{||\mathbf{d}'||_2=1,\boldsymbol{\beta}\in\mathbb{R}^{|\omega|}} \sum_{l\in\omega} ||\mathbf{r}_l + \boldsymbol{\alpha}_l[j]\mathbf{d} - \beta_l\mathbf{d}'||_2^2. \tag{6}$$

• Update $\mathbf{D}$ using the new atom $\mathbf{d}'$, and replace the scalars $\boldsymbol{\alpha}_l[j] \neq 0$ from Eq. (5) in $\boldsymbol{\alpha}$ using $\boldsymbol{\beta}$.

Figure 1. *K-SVD step for updating* $\mathbf{D}$ *and* $\boldsymbol{\alpha}$.

$$\min_{\mathbf{D}\in\mathbb{R}^{n\times k}} \sum_{l=1}^{M} \mathcal{R}(\mathbf{x}_l,\mathbf{D},\boldsymbol{\alpha}_l). \tag{4}$$

In the case of K-SVD, the values of the non-zero coefficients in the $\boldsymbol{\alpha}_l$ are not fixed, and are updated at the same time as the dictionary $\mathbf{D}$. Finding which are the L non-zero coefficients in a patch decomposition, or in other words which are the atoms that take part, is a difficult problem that is addressed during the sparse coding step. On the other hand, setting the values of these non-zero coefficients, once they are selected, is an easy task, which is addressed at the same time as the update of $\mathbf{D}$. To do so, and as detailed in Figure 1, each atom (column) of the dictionary and the coefficients associated with it (the non-zero coefficients in $\boldsymbol{\alpha}$) are updated sequentially[2]. In practice, it has been observed that K-SVD converges with less iterations than MOD [6], motivating the selection of K-SVD for our proposed framework, (though MOD could be used instead). Typically 10 or 20 iterations are enough for K-SVD, with $M = 100\,000$ patches of size $n = 64$ (8×8) and $k = 256$ atoms.

2.2. A reconstructive approach to discrimination

Assume that we have N sets S_i of training patches, $i = 1 \dots N$, belonging to N different classes. The simplest strategy for using dictionaries for discrimination consists of first learning N dictionaries $\mathbf{D}_i, i = 1 \dots N$, one for each class. Approximating each patch using a constant sparsity L and the N different dictionaries provides N different residual errors, which can then be used as classification features. This is essentially the strategy employed in [27, 34]. Thus, the first naive way of estimating the class i_0 for some patch

$\mathbf{x}$ is to write (as in [41], but with learned dictionaries):

$$\hat{\imath}_0 = \underset{i=1\ldots N}{\arg\min}\, \mathcal{R}^\star(\mathbf{x}, \mathbf{D}_i). \quad (7)$$

Instead of this *reconstruction-based* approach, we show that better results can be achieved, in general, by learning *discriminative* sparse representations, while keeping the same robustness against noise or occlusions (see also [9]), which discriminative methods may be sensitive to [40].[2]

3. Learning discriminative dictionaries

3.1. Discriminative model

The main goal of our paper is to propose a model that learns *discriminative* dictionaries and not only *reconstructive* ones. We still want to use the residual error $\mathcal{R}^\star(\mathbf{x}, \mathbf{D}_i)$ as a discriminant, which has already been shown to be relatively efficient in classification [27, 34, 40, 41], but we want also to increase the discriminative power of $\mathcal{R}^\star(\mathbf{x}, \mathbf{D}_i)$, using the idea that a dictionary $\mathbf{D}_i$ associated to a class S_i should be "good" at reconstructing this class, and at the same time "bad" for the other classes. To this effect, we introduce a discriminative term in Eq. (1), using classical *softmax* discriminative cost functions, for $i = 1 \ldots N$,

$$\mathcal{C}_i^\lambda(y_1, y_2, ..., y_N) \equiv \log\Big(\sum_{j=1}^{N} e^{-\lambda(y_j - y_i)}\Big), \quad (8)$$

which is close to zero when y_i is the smallest value among the y_j, and provides an asymptotic linear penalty cost $\lambda(y_i - \min_j y_j)$ otherwise. These are the multiclass versions of the logistic function presented on Figure 2, which is differentiable and enjoys properties similar to the hinge loss function from the support vector machine (SVM) literature. Increasing the value of the new parameter $\lambda > 0$ provides a higher relative penalty cost for each misclassified patch, at the same time making the cost function less smooth, in terms of maxima of its second derivatives. Denoting $\{\mathbf{D}_j\}_{j=1}^N$ the set of these N dictionaries and $\{\mathcal{R}^\star(\mathbf{x}, \mathbf{D}_j)\}_{j=1}^N$ the set of the N different reconstruction errors provided by the N dictionaries, we want to solve

$$\min_{\{\mathbf{D}_j\}_{j=1}^N} \sum_{\substack{i=1\ldots N \\ l \in S_i}} \mathcal{C}_i^\lambda\big(\{\mathcal{R}^\star(\mathbf{x}_l, \mathbf{D}_j)\}_{j=1}^N\big) + \lambda\gamma\mathcal{R}^\star(\mathbf{x}_l, \mathbf{D}_i), \quad (9)$$

where the parameter $\gamma \geq 0$ controls the trade-off between reconstruction and discrimination. With a high value for γ, our model is close to the classical *reconstructive* one, loosing discriminative power. Note that an optional normalization factor $1/|S_i|$ in front of each term can be added to ensure the same weight to each class. Solving this problem will explicitly tend to make each dictionary $\mathbf{D}_i$ not only good for its own class S_i, but also better for S_i than any of the other dictionaries $\mathbf{D}_j$, $j \neq i$, that are being simultaneously learned for the other classes. At first sight, it might seem that this new optimization problem suffers from several difficulties. Not only is it non-convex and non-differentiable because of the ℓ_0-norm in the function $\mathcal{R}^\star$, but it also seems to lose the simplicity of possible least-squares-based (Eq. [4]) or SVD-based (Eq. [6]) dictionary updates. We actually present next a MOD/K-SVD type of iterative scheme to address this problem, which includes the automatic tuning of the parameters λ and γ.

[2]Note also that due to the over-completeness of the dictionaries, and possible correlations between the classes, sparse representations of members of one class with dictionaries from a different class can still be very efficient and produce small values for the residual error $\mathcal{R}^\star$.

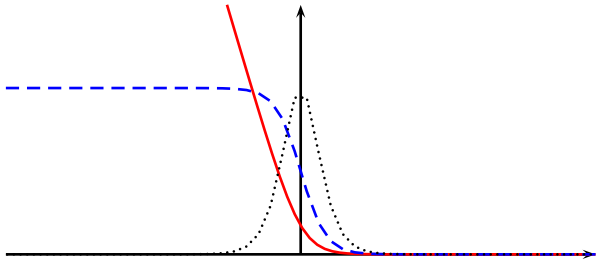


Figure 2. *The logistic function (red, continuous), its first derivative (blue, dashed), and its second derivative (black, dotted).*

3.2. Optimization procedure

We want to apply an iterative scheme similar to those effectively used for the purely reconstructive dictionary learning techniques. First, one initializes the dictionaries $\mathbf{D}_i$ in $\mathbb{R}^{n\times k}$, for $i = 1 \ldots N$, using a few iterations of K-SVD on each class separately (reconstructive approach). Then the main loop at iteration k is composed of:

• *Sparse coding:* This step remains the same. The dictionaries $\mathbf{D}_i$ are fixed, and one computes the decompositions $\boldsymbol{\alpha}_{li} = \boldsymbol{\alpha}^\star(\mathbf{x}_l, \mathbf{D}_i)$, for each patch l and dictionary i. Note that while this step was directly involved in the minimization of Eq. (1) in the reconstructive approach, its purpose here is not to minimize Eq. (9), but to compute the $\boldsymbol{\alpha}^\star$ and $\mathcal{R}^\star$ that are necessary in the following step.[3]

• *Dictionary update:* This step is the one that will make the dictionaries more discriminative. Following the same ideas as in K-SVD, we want to compute new dictionaries $\mathbf{D}_i$. This is done by updating each single atom (column) of each dictionary sequentially, while letting the corresponding $\boldsymbol{\alpha}$ coefficients, associated with an atom during the previous sparse coding, to change as well. This update step raises several questions, which are discussed below. The full procedure is given in Figure 3 at the end of this section.

Choice of the parameters λ and γ. These parameters are critical to ensure the stability and the efficiency of our pro-

[3]The possible use of a discriminative component, on top of the traditional reconstructive one, inside the sparse coding stage itself is the subject of ongoing parallel efforts.

posed scheme. To automatically choose the value of λ that gives the best result in term of classification performance, we have opted to use a varying λ and to replace it by an ascending series. At iteration p, the dictionary update step should now use the value λ_p. Starting from a low value for λ, it is possible to check every few iterations whether increasing this value provides a better classification rate. Experimentally, this simple idea has proven to always give faster convergence and significantly better results than using a fixed parameter.

When $\gamma = 0$, one addresses a pure discriminative task, which can be difficult to stabilize in our scheme. Indeed, one difference between our framework and K-SVD is that the sparse coding and dictionary update steps do not attempt to minimize exactly the same thing. While the first one addresses the *reconstructive* problem of Eq. (2), the second one addresses the (mostly) *discriminative* learning task of Eq. (9). To cope with this potential source of instability, adding some weight (through a bigger γ) to the reconstructive term will draw the minimization problem toward the reconstructive (and stable) one from Eq. (1). Therefore, one has to choose γ large enough to ensure the stability of the scheme, but as small as possible to enforce the discriminative power.[4] Practically, using a varying γ by building a descending series starting with a high value and updating the same way as for λ, gives stability to the algorithm. To simplify the notation, we will omit these variations of λ and γ in the following.

Updating the dictionary in a MOD-like fashion. Let us first consider how to solve the dictionary update step like in the MOD algorithm, with varying λ and varying reconstruction term γ as detailed above. This minimization problem can be written as

$$\min_{\{\mathbf{D}_j\}_j} \sum_{\substack{i=1\dots N \\ l\in S_i}} \mathcal{C}_i^\lambda\big(\{\mathcal{R}(\mathbf{x}_l,\mathbf{D}_j,\boldsymbol{\alpha}_{lj})\}_{j=1}^N\big) + \gamma\lambda\mathcal{R}(\mathbf{x}_l,\mathbf{D}_i,\boldsymbol{\alpha}_{li}), \tag{10}$$

with the $\boldsymbol{\alpha}_{li}$ being fixed and computed during the previous sparse coding step. The proposed MOD-like approach consists of updating sequentially each dictionary using a *truncated Newton* iteration: Let us denote by $\mathcal{C}(\mathbf{D})$ the cost function optimized with respect to a dictionary $\mathbf{D}$ in Eq. (10). Performing one *Newton* iteration to solve $\nabla\mathcal{C}(\mathbf{D}) = 0$ is equivalent to minimizing a second-order Taylor approximation of $\mathcal{C}$ (see [3], p. 484). When computing the Hessian of $\mathcal{C}$, we have chosen to do a local linear approximation[5] of the softmax functions $\mathcal{C}_i$ by neglecting their second derivatives, keeping only the second-order terms that come from $\mathcal{R}$. This use of an approximated Hessian in a Newton iteration is indeed an instance of a truncated Newton iteration, which can be performed very efficiently in our case.

As already noticed, the softmax functions $\mathcal{C}_i^\lambda$ are mainly composed of quasi-linear parts, making the local linear approximation suitable for most of the patches. This is illustrated in Figure 2 for the logistic function (softmax function with only 2 classes). When patches are correctly classified, their cost is rapidly close to 0 (right part of the figure). When they are "misclassified enough," they lie on a nonconstant linear part of the cost function (left part of the figure). It can be shown that performing our *truncated Newton* iteration to update the p-th dictionary is equivalent to solve

$$\min_{\mathbf{D}'\in\mathbb{R}^{n\times k}} \sum_{i=1}^N \sum_{l\in S_i} w_l \mathcal{R}(\mathbf{x}_l,\mathbf{D}',\boldsymbol{\alpha}_{lp}) \text{ where} \tag{11}$$

$$w_l \equiv \frac{\partial \mathcal{C}_i^\lambda}{\partial y_p}\big(\{\mathcal{R}^\star(\mathbf{x}_l,\mathbf{D}_j)\}_{j=1}^N\big) + \lambda\gamma\mathbf{1}_p(i), \tag{12}$$

the variables y_p are defined in Eq. (8), and $\mathbf{1}_p(i)$ is equal to 1 if $i = p$ and 0 otherwise. This formally resembles a *weighted least squares problem*, but it is different since some of the weights may be negative. It admits a unique solution if and only if the Hessian matrix $\mathbf{A} = \sum_l w_l \boldsymbol{\alpha}_{lp}\boldsymbol{\alpha}_{lp}^T$ is positive definite. The reconstructive term ($\gamma > 0$) plays an important role here. By adding weight exclusively to the diagonal of the matrix $\mathbf{A}$, we have experimentally observed that $\mathbf{A}$ is almost always positive definite, which is a key concept behind the classical *Levenberg-Marquardt* algorithm for nonlinear least-squares [23].

From MOD to K-SVD. Now that we have shown how to use a MOD-like dictionary update (no change on $\boldsymbol{\alpha}$ during this step), we present the main ideas behind the use of a K-SVD-like update. Recall that this sequentially improves each atom of each dictionary, while allowing for the coefficients $\boldsymbol{\alpha}$ to change (improve) at the same time. Experimentally, we have observed that this approach converges faster and gives better results than the MOD-like update in our discriminative framework. The main idea we use is the local linear approximation of the softmax function, which provides us with a much easier function to minimize. The detailed algorithm is presented in Figure 3. The first two steps are identical to the K-SVD for reconstruction (Figure 1); These address the selection of the patches that use the atom being updated. The other steps come from the following modification of Eq. (11) from the MOD-like dictionary update. Updating $\mathbf{d}$, j-th atom of the dictionary i, while allowing its corresponding coefficients to change, can be written as Eq. (15), which is a simple eigenvalue problem, where $\mathbf{d}'$ is the eigenvector associated to the largest eigenvalue of

$$\mathbf{B} = \sum_{p=1}^N \sum_{l\in S_p\cap\omega} w_l(\mathbf{r}_l + \boldsymbol{\alpha}_{li}[j]\mathbf{d})(\mathbf{r}_l + \boldsymbol{\alpha}_{li}[j]\mathbf{d})^T. \tag{13}$$

[4] Interestingly, we have observed that this strategy has consequences that are similar to the ideas that bring robustness to the Levenberg-Marquardt algorithm [23], as noticed later in this paper.

[5] This kind of approximation is classical in optimization and derives from the same ideas that motivate the use of Gauss-Newton methods for nonlinear least-squares.

This way, for each atom sequentially, we address the problem of Eq. (9), under the only assumption that the softmax functions can be locally linearly approximated.

Input: N dictionaries $\mathbf{D}_i \in \mathbb{R}^{n\times k}$; M vectors $\mathbf{x}_l \in \mathbb{R}^n$ (input data); S_i (classification of the input data); $\boldsymbol{\alpha} \in \mathbb{R}^{k\times MN}$ (coefficients from the previous iteration).
Output: $\mathbf{D}$ and $\boldsymbol{\alpha}$.
Loop: For $i = 1 \dots N$, for $j = 1 \dots k$, update $\mathbf{d}$, the j-th column of $\mathbf{D}_i$:
• Select the set of patches that uses $\mathbf{d}$:

$$\omega \leftarrow \{l \in 1 \dots M | \boldsymbol{\alpha}_{li}[j] \neq 0\}. \tag{14}$$

• For each patch l in ω, compute the residual of the decomposition of $\mathbf{x}_l$: $\mathbf{r}_l = \mathbf{x}_l - \mathbf{D}_i\boldsymbol{\alpha}_{li}$.
• Compute the same weights w_l as in Equation (12), for all $p = 1 \dots N$, for all l in $S_p \cap \omega$.
• Compute a new atom $\mathbf{d}' \in \mathbb{R}^n$ and the associated coefficients $\boldsymbol{\beta} \in \mathbb{R}^{|\omega|}$ that minimize the residual error on the selected set ω, using Eq. (13)

$$\min_{\substack{||\mathbf{d}'||_2=1 \\ \boldsymbol{\beta}\in\mathbb{R}^{|\omega|}}} \sum_{\substack{p=1\dots N \\ l\in S_p\cap\omega}} w_l||\mathbf{r}_l + \boldsymbol{\alpha}_{li}[j]\mathbf{d} - \beta_l\mathbf{d}'||_2^2. \tag{15}$$

• Update $\mathbf{D}_i$ and $\boldsymbol{\alpha}$ using the new atom $\mathbf{d}'$, and replace the scalars $\boldsymbol{\alpha}_{li}[j] \neq 0$ from Eq. (14) in $\boldsymbol{\alpha}$ using $\boldsymbol{\beta}$.

Figure 3. *Dictionary update step for the proposed discriminative K-SVD, provided the parameters λ and γ.*

3.3. Data issues

Before presenting experiments, we discuss here some data related issues. Depending on the particular application of the proposed framework, different prefiltering operations can be applied to the input data. We mention first the possibility to apply a Gaussian mask to the input patches (by multiplying element-wise the patches by the mask), in order to give more weight to the center of the patches, since the framework is designed for local discrimination. We mention also the possibility to pre-process the data using a Laplacian filter, which has proven to give more discriminatory power. Since a Laplacian filter can be represented by a difference of Gaussians, this step is consistent with previous works on local descriptors. Both proposed pre-filtering can be simultaneously used depending on the chosen application.

The presented framework is flexible in the sense that it is very easy to take into account different types of vectorial information. For instance, using color patches for the K-SVD has been addressed in [19] by concatenating R,G,B information from a patch into single vectors. This can be directly applied here. One could also opt to only include the mean color of a patch, if we consider that the geometrical structure is more meaningful for discrimination. It is therefore possible to work with vectors representing grayscale patches and just 3 average R,G,B values. This permits to take into account the color information without multiplying by 3 the dimensionality of the patches. Depending on the data, other types of information could be added this way.

4. Experimental results and applications

4.1. Texture segmentation of the Brodatz dataset

Texture segmentation and classification is a natural application of our framework, since it can be formulated as a local feature extraction and patch classification process. We have chosen to evaluate our method on the Brodatz dataset, introduced in [28], which provides a set of "patchwork" images composed of textures from different classes, and a training sample for each class. The suite of the 12 images is presented in [17].

In our experiments, following the same methodology as [34], each of the patches of the training images were used as a training set. We use patches of size $n = 144$ (12×12), dictionaries of size $k = 128$ and a sparsity factor $L = 4$. A Gaussian mask of standard deviation 4 (element-wise multiplication) and a Laplacian filter were applied on each patch as a prefiltering. Then 30 iterations of our discriminative framework are performed. After this training stage, each one of the 12×12 patches from the test images are classified using the learned dictionaries (by comparing the corresponding representation error $\mathcal{R}^\star$ for each dictionary). Smoothing follows to obtain the segmentation. We present two alternatives, either using a simple Gaussian filtering with a standard deviation of 12, or applying a graph-cut alpha-expansion algorithm, [4, 11], based on a classical Potts model with an 8-neighborhood system. The cost associated to a patch $\mathbf{x}$ and a class S_i is 0 if $\mathbf{x}$ has been classified as part of S_i, and 1 otherwise. A constant regularization cost between two adjacent patches of 1.75 has proven to be appropriate. Table 1 reports the results.

From these experiments, we observe that our method significantly outperforms those reported in [16, 17, 28, 34], regardless of the selected regularization method, and performs best for all but two of the images. Moreover, while the purely *reconstructive* framework already provides good results, we observe that the *discriminative* one noticeably improves the classification rate except for examples 5 and 12. In these two particular cases, it has proven to be an artefact from the image smoothing. The classification rate before and after smoothing are indeed not fully correlated. Smoothing will better remove isolated misclassified patches and sometimes the misclassified patches from the reconstructive approach are more isolated than with the discriminative one. Note that our model under-performs at image 2, where the size of the patches we had chosen has proven to be particularly not adapted, which is a motivation for developing a multiscale framework. Some qualitative results are presented in Figure 4.

♯	[28]	[17]	[34]	[16]	R1	R2	D1	D2
1	7.2	6.7	5.5	3.37	2.22	1.69	1.89	**1.61**
2	18.9	14.3	**7.3**	16.05	24.66	36.5	16.38	16.42
3	20.6	10.2	13.2	13.03	10.20	5.49	9.11	**4.15**
4	16.8	9.1	5.6	6.62	6.66	4.60	3.79	**3.67**
5	17.2	8.0	10.5	8.15	5.26	**4.32**	5.10	4.58
6	34.7	15.3	17.1	18.66	16.88	15.50	12.91	**9.04**
7	41.7	20.7	17.2	21.67	19.32	21.89	11.44	**8.80**
8	32.3	18.1	18.9	21.96	13.27	11.80	14.77	**2.24**
9	27.8	21.4	21.4	9.61	18.85	21.88	10.12	**2.04**
10	0.7	0.4	NA	0.36	0.35	**0.17**	0.20	**0.17**
11	**0.2**	0.8	NA	1.33	0.58	0.73	0.41	0.60
12	2.5	5.3	NA	1.14	1.36	**0.37**	1.97	0.78
Av.	18.4	10.9	NA	10.16	9.97	10.41	7.34	**4.50**

Table 1. *Error rates for the segmentation/classification task for the Brodatz dataset. The proposed framework is compared with a number of reported state-of-the-art results [16, 17, 34] and the best results reported in [28]. R1 and R2 denote the reconstructive approach, while D1 and D2 stand for the discriminative one. A Gaussian regularization has been used for R1 and D1, a graph-cut-based one for R2 and D2. The best results for each image are in bold.*

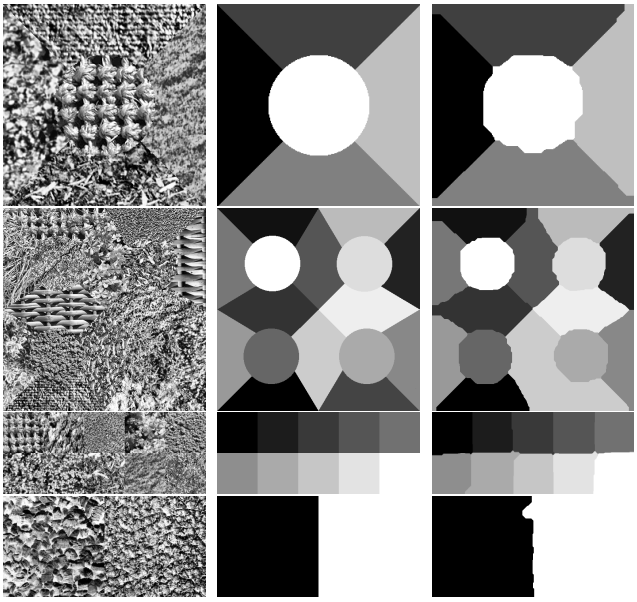


Figure 4. *Subset of the Brodatz dataset with various number of classes: From top to bottom, images* 4*,* 7*,* 9 *and* 12*. The ground-truth segmentation is displayed in the middle and the resulting segmentation on the right side with a graph-cut regularization. Note that the segmentation is in general very precise but fails at separating two classes on image* 7*.*

4.2. Learning discriminant images patches

To assess the promise of our local appearance model, we verify its ability to learn discriminative patches for object categories from a very general database with a high variability. To that effect, we have chosen some classes from the Pascal VOC06 database [26] and conducted the following qualitative experiment: Given one object class A (*e.g.*, bicycle, sheep, car, cow), we build two sets of patches S_1 and S_2. The first one is composed of $200\,000$ 12×12 patches, extracted from bounding boxes that contain one object of class A. The second one was composed of $200\,000$ 12×12 background patches from an image that contains one object of class A. This way, classification at the patch level is difficult since the overlap between S_1 and S_2 is important: (i) many small patches from an object look like some patches from the background and vice-versa; (ii) some patches from an object's bounding boxes do not necessarily fully overlap with the object.

Analyzing globally the patches as a whole, or using a multiscale framework like in [1, 36], to capture global appearance of objects, are among the possibilities that could make this problem more tractable, but these are beyond the scope of this paper. Instead, we want to show here that our purely local model can deal with this overlap between classes and learn the local parts of objects from class A that are discriminative, when observed at the patch level. The experiment we did consists of learning two discriminative dictionaries, $\mathbf{D}_1$ for S_1 and $\mathbf{D}_2$ for S_2, with $k = 128$, $L = 4$ and 15 iterations of our algorithm, and then to pursue the discriminative learning during 15 additional iterations, but at each new iteration, pruning the set S_1 by keeping the 90% "best classified patches." This way, one hopes to remove the overlap between S_1 and S_2 and to enforce the learning on the key-patches of the objects. Examples with various classes of the Pascal dataset are presented in Figure 5. All the images we used in our test procedure are from the official validation set and are not used during the training. The test images are rescaled so that the maximum between the height and the width of the image is less than 256 pixels. The same prefiltering as for the texture segmentation is applied and the average color of each patch is taken into account. The learned key-patches focus on parts of the object that stand as locally discriminative compared to the background. These eventually could be used as inputs to other algorithms of the "bags-of-words" type. Figure 6 shows examples of the learned dictionaries obtained with the discriminative and the reconstructive approaches.

4.3. Weakly-supervised feature selection

Using the same methodology as in [25, 36], we evaluate quantitatively our pixelwise classification on the "bike" category from the Graz02 dataset [24], in a weakly supervised fashion (*without using any ground truth information or bounding box during the training*), with the same parameters, pre-processing and algorithm as in the previous subsection (which prunes iteratively the training set after iteration 15), except that we use a patch size of $n = 15 \times 15$ and process the images at half resolution to capture more context around each pixel. We use the first 300 images of the classes "bike" and "background" and use odd images for training, keeping the even images for testing. To produce a confidence value per pixel we have chosen to measure the reconstruction errors of the tested patches with different sparsity factors $L = 1, \ldots, 15$ and use these values

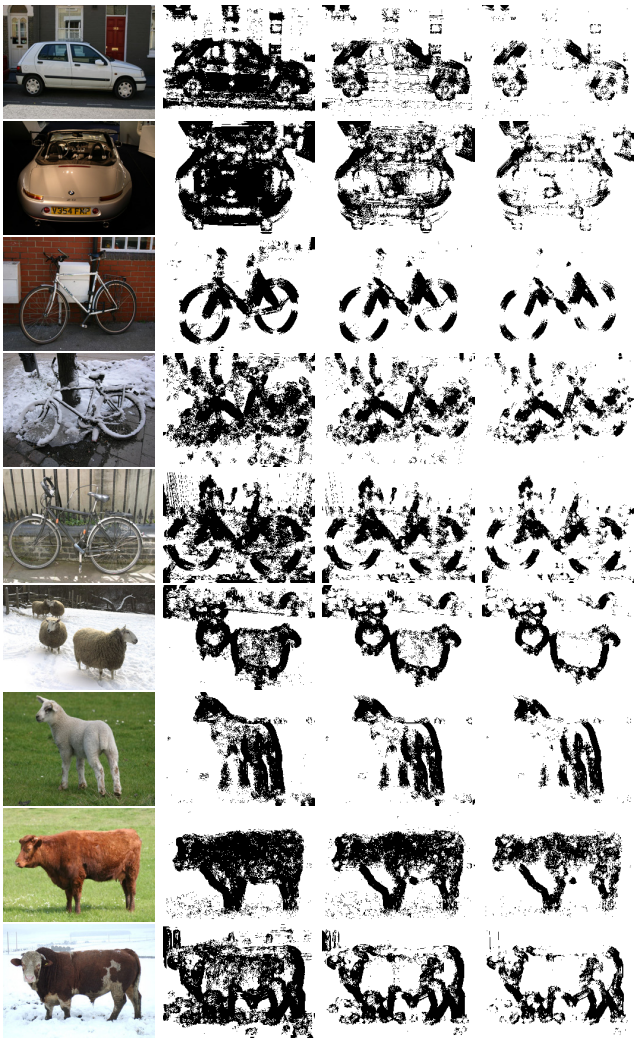
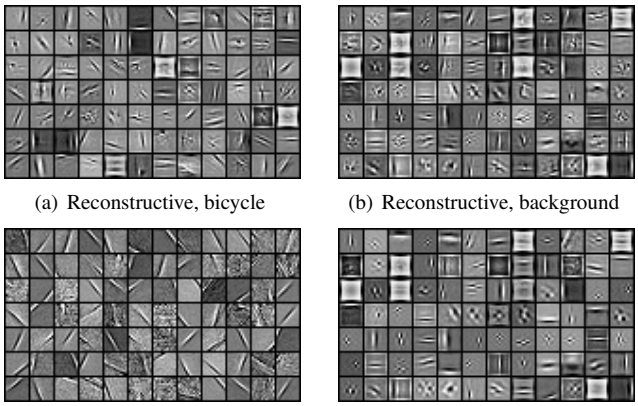


Figure 5. *Learning of key-patches from the Pascal VOC06 dataset. Column 1 presents the test image. Columns 2,3,4 present the raw pixelwise classification results obtained respectively at iterations 20,25 and 30 of the procedure, during the pruning of the dataset. Interestingly, the vertical and horizontal edges of the bicycles are not considered as locally discriminative in an urban environment.*

as feature vectors in a logistic linear classifier. A Gaussian regularization similar to that used in our texture segmentation experiments is applied and has proven to improve noticeably the classification performance. Corresponding precision-recall curves are presented on Figure 7 and compared with [25, 36]. As one can see, our algorithm produces the best results. Nevertheless, a more exhaustive study with different classes and datasets with a more precise ground truth would be needed to draw general conclusions about the relative performance of these three methods.

5. Conclusion and future directions

We have introduced a novel framework for using learned sparse image representations in local classification tasks.



(a) Reconstructive, bicycle (b) Reconstructive, background

(c) Discriminative, bicycle (d) Discriminative, background

Figure 6. *Parts of the dictionaries, learned on the class 'bicycle' from the Pascal VOC06 dataset. The left part has been learned on bounding boxes containing a bicycle, the right part on background regions. The resulting dictionaries from the two approaches, reconstructive and discriminative, are presented. Visually, the dictionaries produced by the discriminative approach are less similar to each other than with the reconstructive one.*

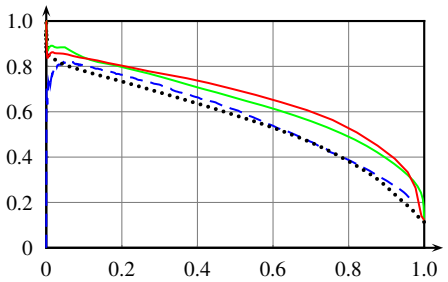


Figure 7. *Precision-recall curve obtained by our framework for the bikes, without pruning of the training dataset (green, continuous), and after* 5 *pruning iterations (red, continuous), compared with the one from [25] (blue, dashed) and [36] (black, dotted).*

Using a local sparsity prior on images, our algorithm learns the local appearance of object categories in a discriminative framework. This is achieved via an efficient optimization of an energy function, leading to the learning of overcomplete and non-parametric dictionaries that are explicitly optimized to be both representative and discriminative. We have shown that the proposed approach leads to state-of-the-art segmentation results on the Brodatz dataset, with significant improvements over previously published methods for most examples. Applied to more general image datasets, mainly of natural images, it permits to learn some key-patches of objects and to perform local discrimination/segmentation.

We are also currently pursuing a discriminative multiscale analysis. This could be embedded into a graph-cut-based segmentation framework, which should take into account both the local classification and the more global image characteristics, as in [32]. In general, we would like to build a model that enjoys both global and local image anal-

ysis capabilities, using for example the coefficients of the decompositions and/or the reconstruction error as local discriminants, combined with more global learned geometric constraints between the patches, as currently being investigated in the scene analysis community.

Acknowledgements

This paper was supported in part by the NSF under grant IIS-0535152, ANR under grant MGA, and an EADS industrial chair to ENS. The work of Guillermo Sapiro is partially supported by ONR, NGA, NSF, ARO, and DARPA.

References

[1] A. Agarwal and B. Triggs. Hyperfeatures - multilevel local coding for visual recognition. In *Proc. ECCV*, 2006.

[2] M. Aharon, M. Elad, and A. M. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Trans. SP*, 54(11), 2006.

[3] S. Boyd and L. Vandenberghe. *Convex optimization. Cambridge University Press*, 2004.

[4] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *PAMI*, 20(12), 2001.

[5] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Sci. Comp.*, 20(1), 1998.

[6] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. IP*, 54(12), 2006.

[7] K. Engan, S. O. Aase, and J. H. Husøy. Frame based signal compression using method of optimal directions (MOD). In *IEEE Intern. Symp. Circ. Syst.*, 1999.

[8] R. Grosse, R. Raina, H. Kwong and A. Y. Ng. Shift-invariant sparse coding for audio classification, In *Proc. UAI*, 2007.

[9] K. Huang and S. Aviyente. Sparse representation for signal classification. In *Adv. NIPS*, 2006.

[10] N. Jojic, B. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *Proc. ICCV*, 2003.

[11] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2), 2004.

[12] J. Lasserre, A. Kannan, and J. Winn. Hybrid learning of large jigsaws. In *Proc. CVPR*, 2007.

[13] S. Lazebnik and M. Raginsky. Supervised learning of quantizer codebooks by information loss minimization. Preprint, 2007.

[14] T-W. Lee and M. S. Lewicki Unsupervised Image Classification, Segmentation, and Enhancement Using ICA Mixture Models. In *IEEE Trans. IP*, 11(3), 2002.

[15] T. Leung and J. Malik Representing and Recognizing the Visual Appearance of Materials using Three-Dimensional Textons. In *IJCV*, 43(1), 2001.

[16] A. D. Lillo, G. Motta, and J. A. Storer. Texture classification based on discriminative features extracted in the frequency domain. In *Proc. ICIP*, 2007.

[17] T. Mäenpää, M. Pietikäinen, and T. Ojala. Texture classification by multi-predicate local binary pattern operators In *Proc. ICPR*, 2000.

[18] J. Malik, S. Belongie, T. Leung and J. Shi. Contour and texture analysis for image segmentation. In *IJCV*, 43(1), 2001

[19] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Trans. IP.*, 17(1), 2008

[20] J. Mairal, G. Sapiro, and M. Elad. Multiscale sparse image representation with learned dictionaries. In *Proc. ICIP*, 2007.

[21] S. Mallat. *A wavelet tour of signal processing, second edition*. Academic Press, New York, 1999.

[22] S. Mallat and Z. Zhang. Matching pursuit in a time-frequency dictionary. *IEEE Trans. SP*, 41(12), 1993.

[23] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Num. Anal.* 15(5), 1978.

[24] A. Opelt and A. Pinz. Object localization with boosting and weak supervision for generic object recognition. In *SCIA*, 2005.

[25] C. Pantofaru, G. Dorkó, C. Schmid and M. Hebert. Combining regions and patches for object class localization. *The Beyond Patches Workshop, CVPR*, 2006.

[26] M. Everingham, A. Zisserman, C. K. I. Williams and L. Van Gool The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results.

[27] G. Peyré. Sparse modeling of textures. *Preprint Ceremade 2007-15*, 2007.

[28] T. Randen and J. H. Husoy. Filtering for texture classification: A comparative study. *PAMI*, 21(4), 1999.

[29] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. CVPR*, 2007.

[30] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. In *Adv. NIPS*, 2006.

[31] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *Proc. CVPR*, 2005.

[32] J. Shotton, J. Winn, C. Rother and A. Criminisi. TextonBoost: joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proc ECCV*, 2006.

[33] F. Schroff, A. Criminisi, and A. Zisserman. Single-histogram class models for image segmentation. In *ICGVIP*, 2006.

[34] K. Skretting and J. H. Husoy. Texture classification using sparse frame-based representations. *EURASIP J. Appl. Signal Process.*, 2006(1), 2006.

[35] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. IT*, 50(10), 2004.

[36] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *Proc. ICCV*, 2007.

[37] M. Varma and A. Zisserman Texture Classification: Are Filter Banks Necessary? In *CVPR*, 2003.

[38] Y. Weiss and W. T. Freeman. What makes a good model of natural images ? In *Proc. CVPR*, 2007.

[39] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proc. ICCV*, 2005.

[40] J. Wright, A. Ganesh, A. Y. Yang, and Y. Ma. Robust face recognition via sparse representation. to appear in *PAMI*.

[41] A. Y. Yang, J. Wright, Y. Ma, and S. Sastry. Feature selection in face recognition: A sparse representation perspective. *UC Berkeley Tech Report UCB/EECS-2007-99*, 2007.