Discriminative Sparse Image Models for Class-Specific Edge Detection and Image Interpretation

Julien Mairal[1,4], Marius Leordeanu[2], Francis Bach[1,4],
Martial Hebert[2], and Jean Ponce[3,4]

[1] INRIA, Paris-Rocquencourt
[2] Carnegie Mellon University, Pittsburgh
[3] Ecole Normale Supérieure, Paris
[4] WILLOW project-team, ENS/INRIA/CNRS UMR 8548.

Abstract. Sparse signal models learned from data are widely used in audio, image, and video restoration. They have recently been generalized to discriminative image understanding tasks such as texture segmentation and feature selection. This paper extends this line of research by proposing a multiscale method to minimize least-squares reconstruction errors and discriminative cost functions under ℓ_0 or ℓ_1 regularization constraints. It is applied to edge detection, category-based edge selection and image classification tasks. Experiments on the Berkeley edge detection benchmark and the PASCAL VOC'05 and VOC'07 datasets demonstrate the computational efficiency of our algorithm and its ability to learn local image descriptions that effectively support demanding computer vision tasks.

1 Introduction

Introduced in [21], learned sparse representations have recently been the focus of much attention and have led to many state-of-the-art algorithms for various signal and image processing tasks [8, 15, 22]. Different frameworks have been developed, which exploit learned sparse decompositions, nonparametric dictionary learning techniques [1, 9, 11], convolutional neural networks [24], probabilistic models [25], each of them being applied to the learning of natural image bases. Recently, a novel discriminative approach of the dictionary learning techniques has been proposed in [14], and it has been applied on texture segmentation and category-based feature selection.

In this paper, we present a method for learning overcomplete bases, which combines ideas from [1, 9, 11] but with a slightly better convergence speed. It is also compatible with ℓ_0 and ℓ_1 regularization sparsity constraints.[5] We use this algorithm in a multiscale extension of the discriminative framework of [14] and apply it to the problem of edge detection, with raw results very close to the state of the art on the Berkeley segmentation dataset [19]. Following [23], we also learn a class-specific edge detector and show that using it as a preprocessing stage to a state-of-the-art edge-based classifier [12] can dramatically improve its performance of the latter.

[5] The ℓ_2 norm of a vector $\mathbf{x}$ in $\mathbb{R}^n$ is defined as $||\mathbf{x}||_2 = (\sum_{i=1}^n \mathbf{x}[i]^2)^{\frac{1}{2}}$, the ℓ_1-norm as $||\mathbf{x}||_1 = \sum_{i=1}^n |\mathbf{x}[i]|$, and $||\mathbf{x}||_0$, the ℓ_0-pseudo-norm of $\mathbf{x}$, counts its number of nonzero elements.

2 Background

Consider a signal $\mathbf{x}$ in $\mathbb{R}^n$. We say that $\mathbf{x}$ admits a sparse representation over a dictionary $\mathbf{D}$ in $\mathbb{R}^{n\times k}$ composed of k unit vectors (atoms) of $\mathbb{R}^n$ when one can find a linear combination of a few atoms from $\mathbf{D}$ that is "close" to the original signal $\mathbf{x}$. Given an input matrix $X = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ in $\mathbb{R}^{n\times m}$ of m signals, learning such a dictionary can be formulated as an optimization problem over a dictionary $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_k]$ in $\mathbb{R}^{n\times k}$ and the sparse representation matrix $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_m]$ in $\mathbb{R}^{k\times m}$, namely

$$\min_{\mathbf{D},\boldsymbol{\alpha}} \sum_{l=1}^{m} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l||_2^2 \quad \begin{array}{l}\text{subject to } \; ||\mathbf{d}_j||_2^2 = 1 \;\text{ and }\; \phi(\boldsymbol{\alpha}_l) \leq 0 \\ \text{for } \; j = 1, \ldots, k \;\text{ and }\; l = 1, \ldots, m.\end{array} \tag{1}$$

Here, $\phi(\boldsymbol{\alpha}_l) \leq 0$ is a *sparsity constraint* guaranteeing that only a few of the components of each vector $\boldsymbol{\alpha}_l$ are nonzero. The most popular sparsity constraints in the literature involve the ℓ_0-pseudo-norm, and the ℓ_1-norm (see [11] for other sparsity functions). In the first case, we simply take $\phi(\boldsymbol{\alpha}_l) = ||\boldsymbol{\alpha}_l||_0 - L$, where L is the maximum number of nonzero coefficients allowed. In the second case, we take $\phi(\boldsymbol{\alpha}_l) = ||\boldsymbol{\alpha}_l||_1 - T$, where T is an arbitrary parameter. It is well known in the statistics, optimization and compressed sensing communities that the ℓ_1 constraint yields a sparse solution [6], but there is no analytic link between the value of T and the effective sparsity L that it yields.

A number of practical methods have been developed for solving problem (1). This includes the K-SVD algorithm of Aharon et al. [1] and the method of optimal directions (MOD) of Engan et al. [9] for its ℓ_0 formulation, and the algorithm of Lee et al. [11] for its ℓ_1 variant. All these techniques [1, 9, 11] are iterative approaches designed to minimize the energy (1). After an initialization of the dictionary $\mathbf{D}$ in $\mathbb{R}^{n\times k}$, e.g., from random signals, they iterate between a *sparse coding* step where $\mathbf{D}$ is fixed and the matrix $\boldsymbol{\alpha}$ is computed, and a *dictionary update* step, where $\mathbf{D}$ is updated with $\boldsymbol{\alpha}$ fixed in [9, 11] and variable in [1]. Given a signal $\mathbf{x}_l$ in $\mathbb{R}^n$ and a fixed dictionary $\mathbf{D}$ in $\mathbb{R}^{n\times k}$, sparse coding amounts to solving the following optimization over $\boldsymbol{\alpha}_l$ in $\mathbb{R}^k$:

$$\min_{\boldsymbol{\alpha}_l \in \mathbb{R}^k} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l||_2^2, \;\; \text{s.t. } \; \phi(\boldsymbol{\alpha}_l) \leq 0. \tag{2}$$

In the ℓ_0 case, solving this problem and finding the corresponding nonzero coefficients—what we will call the *sparsity pattern* in the rest of this paper—is NP-hard. In the applications addressed in this paper as well as in [1, 9], the dimension n of the signals is quite small and so is the sparsity factor L (typically $n \leq 1000$ and $L \leq 10$), which makes it reasonable to use a greedy algorithm called orthogonal matching pursuit (OMP) [17] to effectively find an approximate solution. Using the ℓ_1 formulation of sparse coding "convexifies" this problem, and the LARS-Lasso algorithm [7], can be used to find its global optimum. In general, the question of whether to prefer an ℓ_0 or ℓ_1 formulation has not been settled yet. The main advantages of the ℓ_1 norm are that it is easy to optimize and, since it is piecewise smooth, it provides more stable decompositions. In practice, with the small signals and low sparsity typical of our applications, OMP usually provides sparser solutions but is not as stable in the sense that a small variation of the input data may result in a completely different sparsity pattern.

When the coefficient matrix $\boldsymbol{\alpha}$ in $\mathbb{R}^{k\times m}$ is fixed, updating the dictionary is a linear least-squares problem under quadratic constraints. For a given data matrix X in $\mathbb{R}^{n\times m}$,

it can be formulated as the following optimization problem over $\mathbf{D}$ in $\mathbb{R}^{n\times k}$:

$$\min_{\mathbf{D}} \sum_{l=1}^{m} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l||_2^2 \quad \text{subject to } \; ||\mathbf{d}_j||_2^2 = 1 \;\text{ for }\; j = 1, \ldots, k. \tag{3}$$

This constrained optimization problem can be addressed using several methods, including, as noted in [11], gradient descent with iterative projection, or a dual version derived from its Lagrangian. This is the approach followed in the ℓ_1 formulation by the method of Lee et al. [11]. Engan et al for the MOD algorithm have chosen to solve the problem (3) without constraints and normalize the $\mathbf{d}_j$ a posteriori since multiplying a column of $\mathbf{D}$ while dividing the j-th line of $\boldsymbol{\alpha}$ by the same value does not change the energy in Eq. (1). The K-SVD of Aharon et al. [1] uses a different strategy where the columns of $\mathbf{D}$ are updated one at a time but only the sparsity pattern is fixed while the values of the nonzero coefficients of $\boldsymbol{\alpha}$ are allowed to change as well. This allows for larger steps at the cost of more complex calculations in general.

The approach proposed in this paper combines several of the aspects of the methods reviewed so far. In particular, as will be shown in the next section, it can handle both ℓ_0 and ℓ_1 formulations of problem (1), takes advantage of the LARS-Lasso or OMP sparse coding speed when appropriate, and enjoys fast dictionary updates steps similar to [9, 11], while letting the $\boldsymbol{\alpha}$ coefficients change for faster convergence, similar to K-SVD [1]. It also generalizes to discriminative tasks in a straightforward way.

3 A method for sparse model learning

In this section, we present how to learn reconstructive and discriminative sparse representations and a multiscale extension of the latter.

3.1 Learning reconstructive dictionaries

In our experiments, the MOD and K-SVD algorithms present very similar performances in terms of convergence and speed. Both algorithms suffer from using the same expensive sparse coding step, even with efficient Cholesky-based implementations. With sets of parameters commonly used in image, signal and video processing, the K-SVD dictionary update, which relies on k truncated SVDs of matrices of size roughly $n \times \frac{mL}{k}$, is slower than the MOD one (one inversion of a $k \times k$ symmetric matrix) except when k is very large, but performs larger steps.

The algorithm we propose here extends [9, 11] and thus enjoys fast dictionary updates, while exploiting fast updates of the nonzero coefficients of $\boldsymbol{\alpha}$ by fixing only the sparsity pattern like in the K-SVD. Note that such a trick to accelerate convergence has already been used successfully in [20], but in a different context. The overall process is outlined in Figure 1. Instead of solving a single instance of Eq. (5) with $\boldsymbol{\alpha}$ fixed to update $\mathbf{D}$, we alternate between this update, which we call *partial dictionary update*, and a fast update of the nonzero coefficients of $\boldsymbol{\alpha}$ with $\mathbf{D}$ fixed (*partial fast sparse coding*). This allows us to reduce the number of calls of the expensive full sparse coding step.

In the ℓ_0 case, a *partial dictionary update* can be $\mathbf{D}(\Gamma) = \mathbf{X}\boldsymbol{\alpha}^T(\boldsymbol{\alpha}\boldsymbol{\alpha}^T)^{-1}$ as in the MOD. In the ℓ_1 case, it can be done efficiently like in [11] by using a Newton method

Input: $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}^{n \times m}$ (input data); k (number of atoms); $\phi : \mathbb{R}^k \rightarrow \mathbb{R}$ (constraint on the coefficients); J (number of iterations).
Output: $\mathbf{D} \in \mathbb{R}^{n \times k}$ (dictionary); $\boldsymbol{\alpha} \in \mathbb{R}^{k \times m}$ (coefficients).
Initialization: Choose randomly some $\mathbf{x}_l$ to initialize the columns of $\mathbf{D}$.
Loop: Repeat J times:

– *Sparse coding:* Fix $\mathbf{D}$ and compute, using OMP (ℓ_0) or LARS-Lasso (ℓ_1):

$$\text{for all } \; l = 1, \ldots, m, \;\; \boldsymbol{\alpha}_l = \underset{\boldsymbol{\alpha}'_l \in \mathbb{R}^k}{\arg\min} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}'_l||^2, \;\; \text{s.t. } \; \phi(\boldsymbol{\alpha}'_l) \leq 0. \tag{4}$$

– *Dictionary update:* Repeat until convergence:
- *Partial dictionary update:* Fix $\boldsymbol{\alpha}$ and solve:

$$\mathbf{D} = \underset{\mathbf{D}' \in \mathbb{R}^k}{\arg\min} ||\mathbf{X} - \mathbf{D}'\boldsymbol{\alpha}||_F^2 \;\; \text{s.t. } \; ||\mathbf{d}'_j||_2 = 1, \;\; \text{for all } \; j = 1, \ldots, k \tag{5}$$

- *Partial fast sparse coding:* Fix $\mathbf{D}$ and update the nonzero coefficients of $\boldsymbol{\alpha}$ to minimize Eq. (4) while keeping the same sparsity pattern.

Fig. 1. *Proposed algorithm for learning reconstructive dictionaries.*

to solve the dual problem arising from the Lagrangian of this constrained optimization problem. In particular, it is easy to show that the optimal vector $\Gamma^\star$ of Lagrange multipliers must satisfy

$$\Gamma^\star = \underset{\Gamma \in \mathbb{R}^k}{\arg\max} \left[||\mathbf{X} - \mathbf{D}(\Gamma)\boldsymbol{\alpha}||_F^2 + \sum_{j=1}^{k} \Gamma_j (\mathbf{d}_j^T \mathbf{d}_j - 1) \right], \tag{6}$$

where $\mathbf{D}(\Gamma) = \mathbf{X}\boldsymbol{\alpha}^T(\boldsymbol{\alpha}\boldsymbol{\alpha}^T + \text{diag}(\Gamma))^{-1}$.

Assuming that the sparsity pattern is fixed, partial fast sparse coding becomes much less costly than full sparse coding: given a fixed dictionary $\mathbf{D}$ in $\mathbb{R}^{n \times k}$, a signal $\mathbf{x}_l$ in $\mathbb{R}^n$, and the active dictionary $\mathbf{D}_a$ in $\mathbb{R}^{n \times L}$ composed of the atoms corresponding to the L nonzero coefficients of $\boldsymbol{\alpha}_l$, the vector $\tilde{\boldsymbol{\alpha}}_l$ in $\mathbb{R}^L$ composed of the nonzero values from $\boldsymbol{\alpha}_l$ can be updated as follows:

– In the ℓ_0 case, $\tilde{\boldsymbol{\alpha}}_l$ is the minimum of $||\mathbf{x}_l - \mathbf{D}_a\tilde{\boldsymbol{\alpha}}_l||_2^2$, and the corresponding linear least-squares system is solved using a conjugate gradient method.
– In the ℓ_1 case, we prevent the sign of the values in $\tilde{\boldsymbol{\alpha}}_l$ from strictly changing, but we allow them to be zero. We denote by ϵ_a in $\{-1, 1\}^L$ the sign of the initial $\tilde{\boldsymbol{\alpha}}_l$ and we address the problem

$$\min_{\tilde{\boldsymbol{\alpha}}_l} ||\mathbf{x}_l - \mathbf{D}_a\tilde{\boldsymbol{\alpha}}_l||_2^2 \;\; \text{s.t. } \; \epsilon_a^T \tilde{\boldsymbol{\alpha}}_l = T \;\; \text{and} \;\; \begin{array}{l} \epsilon_a[j]\tilde{\boldsymbol{\alpha}}_l[j] \geq 0 \\ \text{for } \; j = 1, \ldots, L \end{array} \tag{7}$$

using reduced projected gradient descent [13] with optimal steps. We repeat until convergence:
- Gradient computation: $\mathbf{g} = -2\mathbf{D}_a^T(\mathbf{x}_l - \mathbf{D}\tilde{\boldsymbol{\alpha}}_l)$.

- Projection of $\mathbf{g}$ so that $\sum_j \tilde{\mathbf{g}}[j] = 0$: $\tilde{\mathbf{g}} = \big(\mathbf{I} - \frac{\epsilon_a \epsilon_a^T}{\epsilon_a^T \epsilon_a}\big)\mathbf{g}$.
- Computation of the optimal step, which prevents the sign of the coefficients to change: $t = \min(\frac{\tilde{\boldsymbol{\alpha}}_l[1]}{\tilde{\mathbf{g}}[1]}, \ldots, \frac{\tilde{\boldsymbol{\alpha}}_l[L]}{\tilde{\mathbf{g}}[L]}, \frac{\tilde{\mathbf{g}}^T \mathbf{g}}{\tilde{\mathbf{g}}^T \mathbf{D}_a^T \mathbf{D}_a \tilde{\mathbf{g}}})$.
- Steepest descent: $\tilde{\boldsymbol{\alpha}}_l' = \tilde{\boldsymbol{\alpha}}_l - t\tilde{\mathbf{g}}$,
- If $\tilde{\boldsymbol{\alpha}}_l[j] = 0$, it is removed from $\tilde{\boldsymbol{\alpha}}_l$. Note that for simplicity reasons we chose not to allow a coefficient which has been removed from $\tilde{\boldsymbol{\alpha}}_l$ to change again. Thus, this descent algorithm stops before the exact solution of Eq. (7).

This approach for learning sparse representations extends [9, 11], since the only difference with these algorithms is the idea of using a partial fast sparse coding to accelerate the convergence, with often less computational effort than the K-SVD. Note that it also extends the K-SVD in some sense in the ℓ_0 case (since K-SVD is per se not compatible with an ℓ_1 constraint). Suppose that we perform our dictionary update on a single atom, by keeping the other atoms fixed. Then, the alternating iterations between what we call partial dictionary update and partial fast sparse coding do exactly the same as the power method, which performs the truncated SVD used in the K-SVD algorithm.

3.2 Learning discriminative dictionaries

An effective method for learning discriminative dictionaries under ℓ_0 constraints has been introduced in [14] using an energy formulation that contains both sparse reconstruction and class discrimination components, jointly optimized towards the learning of the dictionaries. Given N classes S_i of signals, $i = 1, \ldots, N$, the goal is to learn N discriminative dictionaries $\mathbf{D}_i$, each of them being adapted to reconstructing a specific class better than others. As shown in [14], this yields the following optimization problem:

$$\min_{\{\mathbf{D}_j\}_{j=1}^N} \sum_{i=1}^{N} \sum_{l \in S_i} \mathcal{C}_i^{\lambda}\big(\{\mathcal{R}^{\star}(\mathbf{x}_l, \mathbf{D}_j)\}_{j=1}^N\big) + \lambda\gamma \mathcal{R}^{\star}(\mathbf{x}_l, \mathbf{D}_i), \tag{8}$$

$$\text{where } \mathcal{R}^{\star}(\mathbf{x}_l, \mathbf{D}) \equiv \min_{\boldsymbol{\alpha}_l} ||\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l||_2^2 \text{ s.t. } \phi(\boldsymbol{\alpha}_l) \leq 0. \tag{9}$$

Here, $\mathcal{R}^{\star}(\mathbf{x}_l, \mathbf{D}_i)$ is the reconstruction error of the signal $\mathbf{x}_l$ using the dictionary $\mathbf{D}_i$ and $\mathcal{C}_i^{\lambda}$ is a softmax discriminative cost function, which is the multiclass version of the logistic regression function. Its purpose is to make the dictionary $\mathbf{D}_i$ better at reconstructing the signals from class S_i than the dictionaries $\mathbf{D}_j$ for j different than i. In this equation, λ is a parameter of the cost function, and γ controls the trade-off between reconstruction and discrimination. More details on this formulation and on the choices of the parameters λ and γ are given in [14].

The optimization procedure in this case uses the same iterative (i) sparse coding and (ii) dictionary update scheme as the MOD and K-SVD. Nevertheless, due to the different nature of the energy, the dictionary update is slightly different from the MOD or K-SVD. It is implemented as a *truncated Newton* iteration with respect to the dictionaries. It is shown in [14] that performing this truncated Newton iteration to update the

j-th dictionary $\mathbf{D}_j$ is equivalent to solving a problem of the form:

$$\min_{\mathbf{D}' \in \mathbb{R}^{n \times k}} \sum_{i=1}^{N} \sum_{l \in S_i} w_l ||\mathbf{x}_l - \mathbf{D}' \boldsymbol{\alpha}_{lj}||_2^2, \tag{10}$$

where the $\boldsymbol{\alpha}_{lj}$'s are the coefficients of the decompositions of $\mathbf{x}_l$ using the dictionary $\mathbf{D}_j$, and the w_l's are weights coming from a local linear approximation of $\mathcal{C}_i^\lambda$. They depend on the derivatives of $\mathcal{C}_i^\lambda$ and therefore have to be recomputed at each step of the optimization procedure.

It is thus clear that this formulation can easily be adapted and generalized to the framework proposed in the previous section, allowing us to use the ℓ_1 as well as the ℓ_0 formulation, which might be more suitable for some applications. The *partial fast sparse coding* remains unchanged and the *partial dictionary update* becomes:

$$\begin{aligned} \mathbf{D}(\Gamma) &= \sum_{i=1}^{N} \sum_{l \in S_i} w_l \mathbf{x}_l \boldsymbol{\alpha}_{lj}^T \big(\sum_{i=1}^{N} \sum_{l \in S_i} w_l \boldsymbol{\alpha}_{lj} \boldsymbol{\alpha}_{lj}^T + \operatorname{diag}(\Gamma) \big)^{-1} \\ \Gamma &= \arg\max_{\Gamma \in \mathbb{R}^k} \sum_{i=1}^{N} \sum_{l \in S_i} w_l ||\mathbf{x}_l - \mathbf{D}(\Gamma) \boldsymbol{\alpha}_l j||_2^2 + \sum_{j=1}^{k} \Gamma_j (\mathbf{d}_j^T \mathbf{d}_j - 1). \end{aligned} \tag{11}$$

Note that interestingly, our framework allows us to update all the atoms at the same time when the K-SVD does some sequential optimization. When this is always a benefit in the reconstructive framework, the discriminative one relies on a local linear approximation of the cost function, which is linked to the weights w_l introduced above. In the ℓ_0 case, while our procedure improves upon the MOD-like algorithm from [14] since it achieves faster the same reconstruction error, the more computationally expensive K-SVD-like algorithm updates sequentially the atoms of the dictionary but also *updates the local linear approximations of the cost function* (recomputes the w_l's) between each update of the atoms, which has proven experimentally to converge toward a better local minimum. Therefore, the choice between our new discriminative framework and the K-SVD-like dictionary update from [14] in the ℓ_0 case becomes a matter of trade-off between speed and quality of the optimization. In the ℓ_1 case, the K-SVD can not be applied, but the partial dictionary update stage of our discriminative framework can alternatively be replaced by a sequential update of the columns of the dictionary while interlacing updates of the $\mathbf{w}_l$'s. The update of the j-th atom when the $\boldsymbol{\alpha}$ are fixed should then be written:

$$\mathbf{d}'_j = \frac{\sum_{i=1}^{N} \sum_{l \in S_i} w_l \boldsymbol{\alpha}_l[j] (\mathbf{x}_l - \sum_{p \neq j} \boldsymbol{\alpha}_l[p] \mathbf{d}_p)}{|| \sum_{i=1}^{N} \sum_{l \in S_i} w_l \boldsymbol{\alpha}_l[j] (\mathbf{x}_l - \sum_{p \neq j} \boldsymbol{\alpha}_l[p] \mathbf{d}_p) ||_2},$$

which is the solution of:

$$\min_{d'_j} \sum_{i=1}^{N} \sum_{l \in S_i} w_l \left\| \mathbf{x}_l - \sum_{p \neq j} \boldsymbol{\alpha}_l[p] \mathbf{d}_p - \boldsymbol{\alpha}_l[j] \mathbf{d}'_j \right\|_2^2 \quad \text{s.t. } ||\mathbf{d}'_j||_2^2 = 1. \tag{12}$$

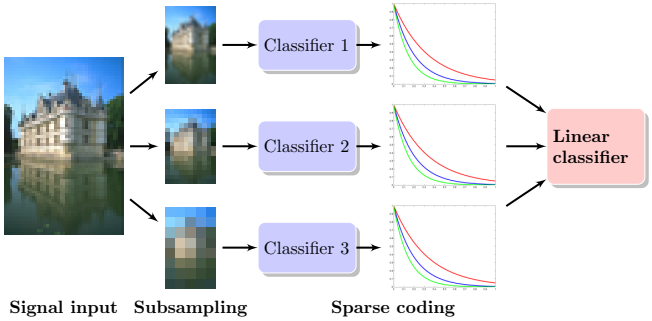

Fig. 2. Multiscale classifier using discriminative sparse coding. The signal input is subsampled in different signal sizes. Then, each classifier outputs N curves of reconstruction errors as functions of a sparsity constraint, one curve per dictionary. A linear classifier provides a confidence value.

3.3 A new multiscale feature space

In this subsection, we present a multiscale extension and some improvements to the classification procedure outlined in [14], which have proven to improve noticeably the performance of our classifier. Although it is presented for illustrative purposes when the signals are image patches, its scope goes beyond vision tasks and similar concepts could be applied in other situations. An important assumption, commonly and successfully used in image processing, is the existence of multiscale features in images, which we exploit using a multi-layer approach, presented in Figure 2. It allows us to work with images at different resolutions, with different sizes of patches and avoids the choice of the hyperparameters L (ℓ_0 case) or T (ℓ_1 case) during the testing phase.

In [14], the class i_0 for some patch $\mathbf{x}$ is taken to be $i_0 = \arg\min_{i=1\ldots N} \mathcal{R}^\star(\mathbf{x}, \mathbf{D}_i)$. However, $\mathcal{R}^\star$ is a reconstruction error obtained with an arbitrary ℓ_0 or ℓ_1 constraint, which does not take into account the different characteristics of the patches. Patches with a high amount of information need indeed a lot of atoms to achieve a correct representation, and should therefore benefit from being classified with a high sparsity factor. On the other hand, some patches admit extremely sparse representations, and should be classified with a small sparsity factor. To cope with this effect, we have chosen when testing a given patch to compute many reconstruction errors with different constraints (different values of L or T). Thanks to the nature of the OMP and LARS-Lasso, this can be done without additional computations since both these algorithms can plot the reconstruction error as a function of the given constraint value in one pass. The two curves produced by two different dictionaries on a patch can then be incorporated into a logistic regression classifier or a linear SVM [26] as feature vectors.

The same idea can be used to combine the output of different classifiers, working at different resolutions and with different sizes of patches. Suppose you train P discriminative classifiers with different sizes of patches and different resolutions. Testing a signal $\mathbf{x}$ consists of sending $\mathbf{x}$ to each classifier independently, cropping and subsampling it so that its size and resolution match the classifier. Each classifier produces N curves representing the reconstruction errors using the N dictionaries and different

sparsity constraints. Then, a linear classifier (logistic regression or SVM) permits to combine these outputs.

4 Combining geometry and local appearance of edges

Considering the edge detection task as a pixelwise classification problem, we have applied our patch-based discriminative framework to learn a local appearance model of "patches centered on an edge pixel" against "patches centered on a non-edge pixel" and therefore have a confidence value for each pixel of being on an edge or not. An evaluation of this scheme is detailed in the experimental part.

Then, once we have trained an edge detector, we propose to use this generic method for class-specific edge detection. Suppose we have N classes of images. After finding the edges of all the images, we can then learn N classifiers to discriminate "patches centered on a pixel located on an edge from an object A" against "patches centered on a pixel located on the other edges". If this method should not be enough for recognizing an object by itself, we show in the experimental part how crucial this local analysis can be when used as a preprocessing of a global contour-based recognition framework.

We now show how to use our edge detector for object recognition by combining it with the shape-based method for category recognition from [12]. Their algorithm learns the shape of a specific category in a discriminative fashion by selecting from training images the pieces of contours that are most relevant for a specific category. The method exploits the pairwise geometric relationships between simple features that include relative angle and distance information. Thus, features are selected based on how discriminative they are together, as a group, and not on an individual basis (for more details on learning these models, see [12]). After the models are learned, they are matched against contours extracted from novel images by formulating the task as a graph matching problem, where the focus is on pairwise relationships between features and not their local appearance. While the authors of [12] make the point that shape is stronger than local appearance for category recognition we want to demonstrate that there is a natural way of combining shape and local appearance that produces a significant increase in the recognition performance.

While shape is indeed important for category recognition, the toughest challenge for such shape-based algorithms on difficult databases is the change in view point, which makes the use of $2D$ shape less powerful. Therefore, it is important to be able to help the shape recognizer by local appearance methods that are geometry independent and thus less sensitive to such changes in viewpoint. Our proposed approach of combining local appearance with shape is to first learn a class specific edge detector on pieces of contours. Next this class specific edge detector is used to filter out the irrelevant edges while learning the shape-based classifier based on [12]. Similarly, at testing time, the shape-based algorithm is applied to the contours that survive after filtering them with our class dependent edge detector. The outputs of both the shape-based classifier and the real values given by our detector are later combined for the final recognition score. This framework provides a natural way of combining the lower-level, appearance based edge detection and contour filtering with the more higher level, shape-based approach. The algorithm can be described in more detail as follows:

• **Contour Training**: learn a class specific edge classifier using our proposed method. For each image, we apply our general edge detector, then obtain pieces of contours obtained as in [12]. Next, we train class specific detectors on such contours belonging to the positive class vs. all other classes.
• **Shape Training**: the output of the class specific edge detector on each training image (contours with average scores less than 0.5 are removed) is given to the shape-based algorithm from [12]. Thus the shape classifier is trained on images that were first preprocessed with our class dependent contour classification.
• **Testing**: each testing image is first preprocessed at the individual contours level in the same way as it is done at training time. The edge classifier it is used to filter out contours that had an average score less than 0.5 (over all pixels belonging to that contour). The contours that survived are then used by the shape-based classifier, to obtain the final recognition score.

5 Experimental validation

5.1 Sparse coding with learned bases

In this experiment, we show that our approach slightly improves upon [1, 9, 11] in terms of convergence speed. Comparing the speed of algorithms is a very delicate issue. Plotting the residual error as a function of the number of iterations would not be fair since the amount of computation per iteration is different from one algorithm to another. Computing the exact number of elementary operations (flops) for one iteration could help, but it is often extremely far from the real computation time. Therefore, we have chosen to base our measures on our careful implementations of the above algorithms. Both our implementations of OMP and LARS-Lasso are efficient parallel Cholesky-based ones. All computations are done on an Intel Quad-Core 2.4Ghz processor.

The comparison is reported in Figure 3 and we report the average ℓ_2-norm of the residuals as a function of the running time of the algorithm for $100\,000$ patches of size 8×8 taken randomly from the Berkeley segmentation database, a dictionary of size $k = 256$, sparsity constraints $L = 6$ for ℓ_0 and $T = 0.1$ for the ℓ_1 case, which are typical settings in the sparse coding literature [8]. In these experiments, our approach proves to slightly outperform those of [1, 9, 11]. Nevertheless, with different parameters or different implementations, results could differ due to the nature of these algorithms. Let us also note that the times we report here are far lower than any of those reported in the literature [8, 11, 15] for comparable tasks.

5.2 Edge Detection

We have chosen to train and evaluate our multiscale discriminative framework on the Berkeley segmentation benchmark [19]. To accelerate the procedure and obtain thin edges, we first process all the images with a Canny edge detector [3] without thresholding. Then, we use the manually segmented images from the training set to classify the pixels from these Canny edges into two classes: S_1 for the ones that are close to a human edge, and S_2 for the others (bad Canny edges). As in [14], RGB patches are concatenated into vectors. The size k of all of our dictionaries are 256. 14 local classifiers using

7 different sizes of patches' edges $e = 5, 7, 9, 11, 15, 19, 23$, and 2 resolutions (full and half) are trained independently and we perform $J = 25$ iterations with a sparsity constraint of $L = 6$, on a sample of $150\,000$ random patches from S_1 and $150\,000$ patches from S_2. This maximum size of patches associated with the half-resolution version of the images allows us to capture sufficient neighborhood context around each pixel, which has proven to be crucial in [5]. A new sample of the training set is encoded using each trained dictionary and we compute the curves of the reconstruction error as function of the sparsity constraint ($L = 1, 2, \ldots, 15$ for the ℓ_0 case), ($T = 0.1, 0.2, \ldots, 2.0$ for the ℓ_1 case). All the curves corresponding to a given patch are concatenated into a single feature vector, which are used to train a linear logistic classifier.

During the test phase, we have chosen to compute independently a confidence value per pixel on the Canny edges without doing any post-processing or spatial regularization, which by itself is also a difficult problem. Precision-recall curves are presented on Figure 4 and are compared with Pb [18], BEL [5], UCM [2] and the recent gPb [16], which was published after that this paper was accepted. Note that with no post-processing, our generic method achieves similar performance as [5] just behind [2] in terms of F-measure (see [18, 19] for its definition), although it was not specifically designed for this task. Compared to Pb, BEL and UCM, our method performs slightly better for high recalls, but is slightly behind for lower recalls, where our edges map contain many small nonmeaningful edges (noise). Recently, gPb has outperformed all of these previous methods by adding global considerations on edges. Examples of dictionaries and results are also presented on Figure 4. Interestingly, we have observed two phenomenons. First, while the dictionaries are learned on RGB color images, most of the learned atoms appear gray, which has already been observed in [15]. Second, we have often noticed color atoms composed of two complementary colors: red/cyan, green/magenta and blue/yellow.

5.3 Category-based edge detection and object recognition

In this section, we use our edge detector on all the images from Pascal VOC'05 and VOC'07 [10] and postprocess them to remove nonmeaningful edges using the same grouping method as [12]. Then, we train our class-specific edge detector on the training set of each dataset, using the same training set as [12] for VOC'05 and the training set of VOC'07. For each class (4 in VOC'05 and 20 in VOC'07) a one-vs-all classifier is trained using the exact same parameters as for the edge detection, which allows us to give a confidence value for each edge as being part of a specific object type. Some examples of filtered edges maps are presented in Figure 5.

In our first set of recognition experiments we want to quantify the benefit of using our class-specific edge detector for object category recognition (shown in Table 1). Thus, we perform the same sets of experiments as [12] and [27], on the same training and testing image sets from Pascal 2005 dataset, on a multiclass classification task. Following the works we compare against, we also use bounding boxes (for more details on the the experiments set up see [12] and [27]). Notice (Tables 1 and 2) that by using our algorithm we are able to reduce the error rate more than 3-fold as compared with the shape alone classifier (3.2% vs. 10.6%) and more than 7-fold when compared to the appearance based method of Winn et al. [27]. We believe that these results are very

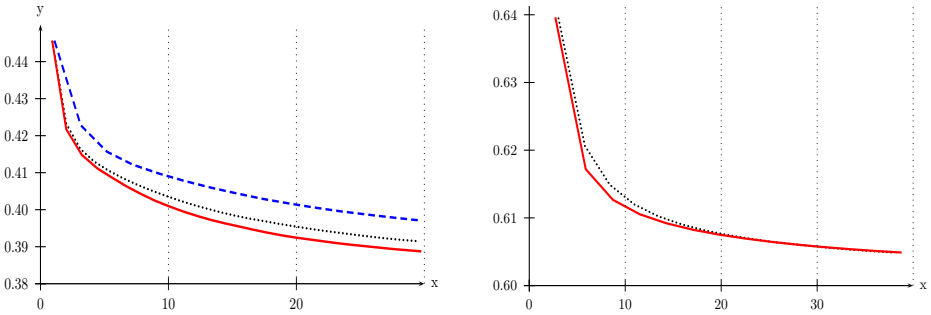

Fig. 3. On the left, the diagram presents an ℓ_0 constrained experiment and on the right, an ℓ_1 one. The curves presented here represents reconstruction errors as a function of the computation time in seconds. Plain red curves represent our algorithm. Black dotted curves represent the MOD [9] in the ℓ_0 case and [11] in the ℓ_1 one. Blue dashed curves represent the K-SVD.

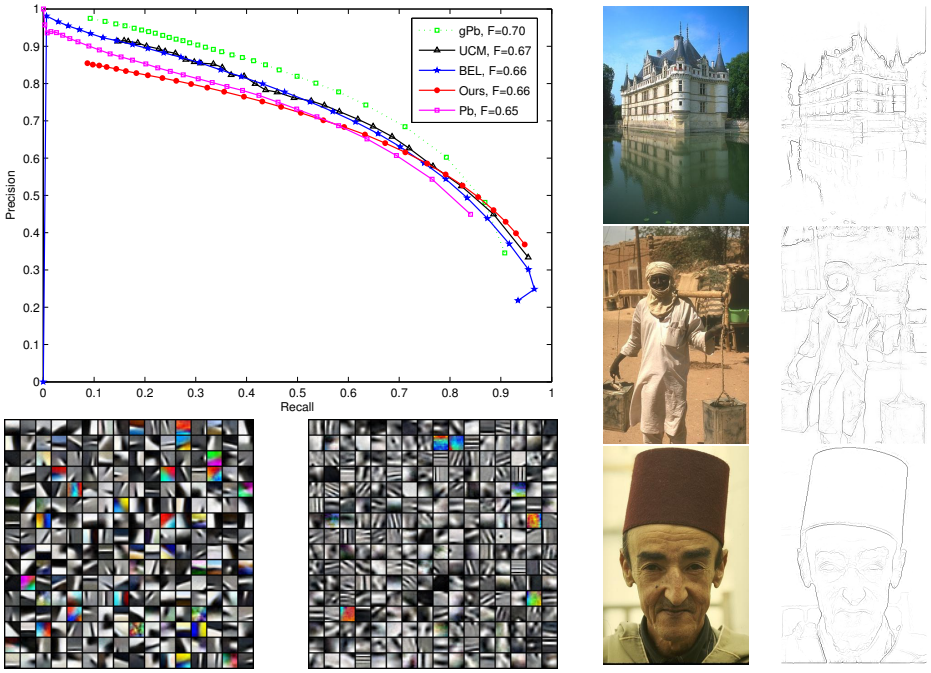

Fig. 4. The curves represents the precision-recall curve for our edge detection task. The two bottom left images are two dictionaries, the left one corresponding to the class S_1 "Good edges" and the right one S_2 "Bad edges". On the right, some images and obtained edge maps. This is a color figure.

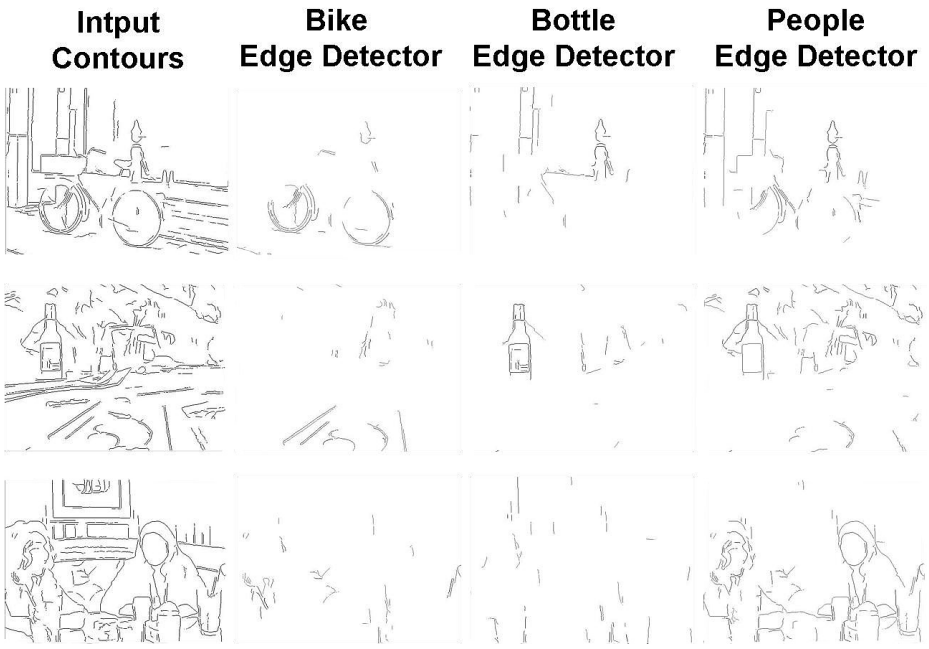

Fig. 5. Examples of filtered edges. Each column shows the edges that survive (score ≥ 0.5) after applying different class specific edge detectors.

encouraging and that they demonstrate the promise of our edge classifier for higher level tasks such as object recognition.

Table 1. Average multiclass recognition rates on the Pascal 2005 Dataset

Algorithm	Ours + [12]	[12]	Winn [27]
Pascal 05 Dataset	96.8%	89.4%	76.9%

In the second set of experiments we want to evaluate how the class-based contour classification can help the shape recognizer on a more challenging dataset, where the objects are undergoing significant changes in viewpoint and scale making their $2D$ shape representation less powerful. To do so, we have chosen the same experimental protocol as for VOC'05 on subset of the VOC'07 dataset composed of 8 object classes (Table 2). For each class we use the training set with the provided bounding boxes for learning both the class specific edge detector and the shape models. But to make the task more challenging, the test set consisted of the full images (no bounding boxes were used) from the official validation set provided in the Pascal 07 challenge (only the images containing one of the eight classes were kept for both testing and training).

Given the difficulty of this dataset we believe that our results are very promising and demonstrate the benefit of combining lower level appearance with higher level, shape based information for object category recognition.

Fusioning these low-level and shape-based classification methods, instead of using them sequentially, by using the sparse representations as learned local geometric features is part of our current effort. The results we have obtained on these preliminary experiments strongly encourage us to pursue that direction.

Table 2. Left: Confusion matrix for Pascal 2005 Dataset (using the bounding boxes). Compare this with the confusion matrix obtained when shape alone is used in [12], on the exact same set of experiments. Right: classification results (recognition performance) at equal error rate for 8 classes from our experiment using the Pascal 07 dataset. Note that this preliminary experiment is different from the official Pascal 07 benchmark [10]. Our filtering method reduces the average error rate by 33%.

Category	Bikes	Cars	Motorbikes	People
Bikes	93.0%	3.5%	1.7%	1.8%
Cars	1.2%	97.7%	0.0%	1.1%
Motorbikes	1.9%	1.8%	96.3%	0%
People	0%	0%	0%	100%

Category	(Ours+[12])	[12]
Aeroplane	71.9%	61.9%
Boat	67.1%	56.4%
Cat	82.6%	53.4%
Cow	68.7%	59.22%
Horse	76.0%	67%
Motorbike	80.6%	73.6%
Sheep	72.9%	58.4%
Tvmonitor	87.7%	83.8%

6 Conclusion

We have presented a multiscale discriminative framework based on learned sparse representations, and have applied it to the problem of edge detection, and class-specific edge detection, which proves to greatly improve the results obtained with a contour-based classifier [12]. Our current efforts are devoted to find a way to use our local appearance model as a preprocessing step for a global recognition framework using bags of words, as we have done for a contour-based classifier. Clustering methods of locally selected patches to define interest regions is an option we are considering, which could eventually allow us to get rid of expensive sliding windows analysis for object detection [4]. Another direction we are pursuing is to use directly the coefficients $\boldsymbol{\alpha}$ of the sparse decompositions as features. Developing a discriminative optimization framework which can use the robust ℓ_1 regularization instead of the ℓ_0 one was a first step, which should make this possible.

Acknowledgments

This paper was supported in part by French grant from the INRIA associated team Thetys and the Agence Nationale de la Recherche (MGA Project).

References

1. Aharon, M., Elad, M., Bruckstein, A.M.: The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. IEEE Trans. SP **54**(11) (2006)
2. Arbelaez, P.: Boundary extraction in natural images using ultrametric contour maps. In: POCV'06, (2006)
3. Canny, J.F.: A computational approach to edge detection. IEEE Trans. PAMI **8**(6) (1986)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. CVPR, (2005)
5. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: Proc. CVPR, (2006)
6. Donoho, D.L.: Compressive sampling. IEEE Trans. IT **52**(4) (2006)
7. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. Ann. Statist. **32**(2) (2004)
8. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. IP **54**(12) (2006)
9. Engan, K., Aase, S.O., Husoy, J.H.: Frame based signal compression using method of optimal directions (MOD). In: Proc. ISCS. (1999)
10. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (2007)
11. Lee, H., Battle, A., Rajat, R., Ng, A.Y.: Efficient sparse coding algorithms. In: Adv. NIPS, (2007)
12. Leordeanu, M., Hebert, M., Sukthankar, R.: Beyond local appearance: Category recognition from pairwise interactions of simple features. In: Proc. CVPR, (2007)
13. Luenberger, D.G.: Linear and Nonlinear Programming. Addison-Wesley (1984)
14. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Discriminative learned dictionaries for local image analysis. In: Proc. CVPR, (2008)
15. Mairal, J., Elad, M., Sapiro, G.: Sparse representation for color image restoration. IEEE Trans. IP **17**(1) (2008)
16. Maire, M., Arbelaez, P., Fowlkes, C., Malik, J.: Using Contours to Detect and Localize Junctions in Natural Images. In: Proc. CVPR, (2008)
17. Mallat, S., Zhang, Z.: Matching pursuit in a time-frequency dictionary. IEEE Trans. SP **41**(12) (1993)
18. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans. PAMI **26**(1) (2004)
19. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. ICCV, (2001)
20. Moghaddam, B., Weiss, Y., Avidan, S.: Spectral bounds for sparse pca: Exact and greedy algorithms. In: Adv. NIPS. (2005)
21. Olshausen, B.A., Field., D.J.: Sparse coding with an overcomplete basis set: A strategy employed by v1? Vision Research **37** (1997)
22. Peyre, G.: Non-negative sparse modeling of textures. In: Proc. SSVM07, (2007)
23. Prasad, M., Zisserman, A., Fitzgibbon, A., Kumar, M.P., Torr, P.: Learning class-specifc edges for object detection and segmentation. In: Proc ICVGIP, (2006)
24. Ranzato, M., Huang, F., Boureau, Y., LeCun, Y.: Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: Proc. CVPR, (2007)
25. Roth, S., Black, M.J.: Fields of experts: A framework for learning image priors. In: Proc. CVPR, (2005)
26. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. (2004)
27. Winn, J., Criminisi, A., Minka, T.: Object categorization by learned universal visual dictionary. In: Proc. ICCV, (2005)