HAL
archives-ouvertes.fr

Task-Driven Dictionary Learning

Julien Mairal, Francis Bach, Jean Ponce

▶ **To cite this version:**

Julien Mairal, Francis Bach, Jean Ponce. Task-Driven Dictionary Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, Institute of Electrical and Electronics Engineers, 2012, 34 (4), pp.30. 10.1109/TPAMI.2011.156 . inria-00521534v2

HAL Id: inria-00521534
https://hal.inria.fr/inria-00521534v2

Submitted on 9 Sep 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Task-Driven Dictionary Learning

Julien Mairal, Francis Bach, and Jean Ponce

Abstract—Modeling data with linear combinations of a few elements from a *learned* dictionary has been the focus of much recent research in machine learning, neuroscience and signal processing. For signals such as natural images that admit such sparse representations, it is now well established that these models are well suited to *restoration* tasks. In this context, learning the dictionary amounts to solving a large-scale matrix factorization problem, which can be done efficiently with classical optimization tools. The same approach has also been used for learning features from data for other purposes, e.g., image classification, but tuning the dictionary in a supervised way for these tasks has proven to be more difficult. In this paper, we present a general formulation for supervised dictionary learning adapted to a wide variety of tasks, and present an efficient algorithm for solving the corresponding optimization problem. Experiments on handwritten digit classification, digital art identification, nonlinear inverse image problems, and compressed sensing demonstrate that our approach is effective in large-scale settings, and is well suited to supervised and semi-supervised classification, as well as regression tasks for data that admit sparse representations.

Index Terms—Basis pursuit, Lasso, dictionary learning, matrix factorization, semi-supervised learning, compressed sensing.

1 Introduction

The linear decomposition of data using a few elements from a *learned* dictionary instead of a predefined one—based on wavelets [1] for example—has recently led to state-of-the-art results in numerous low-level signal processing tasks such as image denoising [2], [3], [4], audio processing [5], [6], as well as classification tasks [7], [8], [9], [10], [11], [12]. Unlike decompositions based on principal component analysis (PCA) and its variants, these sparse models do not impose that the dictionary elements be orthogonal, allowing more flexibility to adapt the representation to the data.

Consider a vector $\mathbf{x}$ in $\mathbb{R}^m$. We say that it admits a sparse approximation over a *dictionary* $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_p]$ in $\mathbb{R}^{m \times p}$, when one can find a linear combination of a "few" columns from $\mathbf{D}$ that is "close" to the vector $\mathbf{x}$. Experiments have shown that modeling signals with such sparse decompositions (*sparse coding*) is very effective in many signal processing applications [13]. For natural images, predefined dictionaries based on various types of wavelets [1] have been used for this task. Initially introduced by Olshausen and Field [14] for modeling the spatial receptive fields of simple cells in the mammalian visual cortex, the idea of learning the dictionary from data instead of using off-the-shelf bases has been shown to significantly improve signal reconstruction [2].

This classical data-driven approach to dictionary learning is well adapted to reconstruction tasks, such as restoring a noisy signal. These dictionaries, which are good at reconstructing clean signals, but bad at reconstructing noise, have indeed led to state-of-the-art denoising algorithms [2], [3], [4]. Unsupervised dictionary learning has also been used for other purposes than pure signal reconstruction, such as classification [5], [7], [11], [12], [15], but recent works have shown that better results can be obtained when the dictionary is tuned to the specific task (and not just data) it is intended for. Duarte-Carvajalino and Sapiro [16] have for instance proposed to learn dictionaries for compressed sensing, and in [8], [9], [10] dictionaries are learned for signal classification. In this paper, we will refer to this type of approach as *task-driven* dictionary learning.

Whereas purely data-driven dictionary learning has been shown to be equivalent to a large-scale matrix factorization problem that can be effectively addressed with several methods [14], [17], [18], [19], its task-driven counterpart has proven to be much more difficult to optimize. Presenting a general efficient framework for various task-driven dictionary learning problems is the main topic of this paper. Even though it is different from existing machine learning approaches, it shares similarities with many of them.

For instance, Blei et al. [20] have proposed to learn a latent topic model intended for document classification. In a different context, Argyriou et al. [21] introduced a convex formulation for multi-task classification problems where an orthogonal linear transform of input features is jointly learned with a classifier. Learning compact features has also been addressed in the literature of neural networks, with restricted Boltzmann machines (RBM's) and convolutional neural networks for example (see [22], [23], [24], [25], [26] and references therein). Interestingly, the question of learning the data representation in an unsupervised or supervised way has also been investigated for these approaches. For instance, a supervised topic model is proposed in [27] and tuning latent data representations for minimizing a cost function is often achieved with *backpropagation* in neural networks [28].

• *When this work was achieved, all authors were with INRIA - Willow Project-Team, Laboratoire d'Informatique de l'Ecole Normale Supérieure. (INRIA/ENS/CNRS UMR 8548), 23, avenue d'Italie, 75214 Paris. France. Jean Ponce is also with Ecole Normale Supérieure, Paris. Francis Bach is now with INRIA - Sierra Project-Team. Julien Mairal is now with the statistics department of the University of California, Berkeley. E-mail:* `firstname.lastname@inria.fr`

1.1 Contributions

This paper makes three main contributions:

- It introduces a supervised formulation for learning dictionaries adapted to various tasks instead of dictionaries only adapted to data reconstruction.
- It shows that the resulting optimization problem is smooth under mild assumptions, and empirically that stochastic gradient descent addresses it efficiently.
- It shows that the proposed formulation is well adapted to semi-supervised learning, can exploit unlabeled data when they admit sparse representations, and leads to state-of-the-art results for various machine learning and signal processing problems.

1.2 Notation

Vectors are denoted by bold lower case letters and matrices by upper case ones. We define for $q \geq 1$, the ℓ_q-norm of a vector $\mathbf{x}$ in $\mathbb{R}^m$ as $\|\mathbf{x}\|_q \triangleq (\sum_{i=1}^m |\mathbf{x}[i]|^q)^{1/q}$, where $\mathbf{x}[i]$ denotes the i-th entry of $\mathbf{x}$, and $\|\mathbf{x}\|_\infty \triangleq \max_{i=1,\ldots,m} |\mathbf{x}[i]| = \lim_{q\to\infty} \|\mathbf{x}\|_q$. We also define the ℓ_0-pseudo-norm as the number of nonzero elements in a vector. We consider the Frobenius norm of a matrix $\mathbf{X}$ in $\mathbb{R}^{m\times n}$: $\|\mathbf{X}\|_{\mathrm{F}} \triangleq (\sum_{i=1}^m \sum_{j=1}^n \mathbf{X}[i,j]^2)^{1/2}$. We also write for a sequence of vectors $\mathbf{x}_t$ and scalars u_t, $\mathbf{x}_t = O(u_t)$, when there exists a constant $K > 0$ independent of t so that for all t, $\|\mathbf{x}_t\|_2 \leq K u_t$, and use a similar notation for matrices (note that for finite-dimensional vector spaces, the choice of norm is irrelevant). When $\Lambda \subseteq \{1,\ldots,m\}$ is a finite set of indices, $\mathbf{x}_\Lambda$ denotes the vector in $\mathbb{R}^{|\Lambda|}$ that carries the entries of $\mathbf{x}$ indexed by Λ. Similarly, when $\mathbf{X}$ is a matrix in $\mathbb{R}^{m\times n}$ and $\Lambda \subseteq \{1,\ldots,n\}$, $\mathbf{X}_\Lambda$ is the matrix in $\mathbb{R}^{m\times|\Lambda|}$ whose columns are those of $\mathbf{X}$ indexed by Λ.

The rest of this paper is organized as follows: Section 2 presents the data-driven dictionary learning framework. Section 3 is devoted to our new task-driven framework, and Section 4 to efficient algorithms to addressing the corresponding optimization problems. Section 5 presents several dictionary learning experiments for signal classification, signal regression, and compressed sensing.

2 Data-Driven Dictionary Learning

Classical dictionary learning techniques [14], [17], [18] consider a finite training set of signals $\mathbf{X} = [\mathbf{x}_1,\ldots,\mathbf{x}_n]$ in $\mathbb{R}^{m\times n}$ and minimize the empirical cost function

$$g_n(\mathbf{D}) \triangleq \frac{1}{n}\sum_{i=1}^n \ell_u(\mathbf{x}_i,\mathbf{D}),$$

with respect to a dictionary $\mathbf{D}$ in $\mathbb{R}^{m\times p}$, each column representing a dictionary element. ℓ_u is a loss function such that $\ell_u(\mathbf{x},\mathbf{D})$ should be small if $\mathbf{D}$ is "good" at representing the signal $\mathbf{x}$ in a sparse fashion. As emphasized by the index u of ℓ_u, this optimization problem is *unsupervised*. As others (see, e.g., [18]), we define $\ell_u(\mathbf{x},\mathbf{D})$ as the optimal value of a sparse coding problem. We choose here the elastic-net formulation of [29]:

$$\ell_u(\mathbf{x},\mathbf{D}) \triangleq \min_{\boldsymbol{\alpha}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda_1\|\boldsymbol{\alpha}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\alpha}\|_2^2, \quad (1)$$

where λ_1 and λ_2 are regularization parameters. When $\lambda_2 = 0$, this leads to the ℓ_1 sparse decomposition problem, also known as *basis pursuit* [13], or *Lasso* [30]. Here, our choice of the elastic-net formulation over the Lasso is mainly for stability reasons. Using a parameter $\lambda_2 > 0$ makes the problem of Eq. (1) strongly convex and, as shown later in this paper, ensures its unique solution to be Lipschitz with respect to $\mathbf{x}$ and $\mathbf{D}$ with a constant depending on λ_2. Whereas the stability of this solution is not necessarily an issue when learning a dictionary for a reconstruction task, it has turned out to be empirically important in some of our experiments with other tasks.

To prevent the ℓ_2-norm of $\mathbf{D}$ from being arbitrarily large, which would lead to arbitrarily small values of $\boldsymbol{\alpha}$, it is common to constrain its columns $\mathbf{d}_1,\ldots,\mathbf{d}_p$ to have ℓ_2-norms less than or equal to one. We will call $\mathcal{D}$ the convex set of matrices satisfying this constraint:

$$\mathcal{D} \triangleq \{\mathbf{D}\in\mathbb{R}^{m\times p} \ \text{ s.t. } \ \forall j\in\{1,\ldots,p\}, \ \ \|\mathbf{d}_j\|_2 \leq 1\}. \quad (2)$$

As pointed out by Bottou and Bousquet [31], one is usually not interested in a perfect minimization of the *empirical* cost $g_n(\mathbf{D})$, but instead in the minimization with respect to $\mathbf{D}$ of the *expected* cost

$$g(\mathbf{D}) \triangleq \mathbb{E}_{\mathbf{x}}[\ell_u(\mathbf{x},\mathbf{D})] \overset{\text{a.s.}}{=} \lim_{n\to\infty} g_n(\mathbf{D}), \quad (3)$$

where the expectation is taken relative to the (unknown) probability distribution $p(\mathbf{x})$ of the data, and is supposed to be finite.[1] In practice, dictionary learning problems often involve a large amount of data. For instance when the vectors $\mathbf{x}$ represent image patches, n can be up to several millions in a single image. In this context, online learning techniques have shown to be very efficient for obtaining a stationary point of this optimization problem [19]. In this paper, we propose to minimize an expected cost corresponding to a *supervised* dictionary learning formulation, which we now present.

3 Proposed Formulation

We introduce in this section a general framework for learning dictionaries adapted to specific supervised tasks, e.g., classification, as opposed to the unsupervised formulation of the previous section, and present different extensions along with possible applications.

3.1 Basic Formulation

Obtaining a good performance in classification tasks is often related to the problem of finding a good data representation. Sparse decompositions obtained with data-driven learned dictionaries have been used for that

1. We use "a.s." (almost surely) to denote convergence with probability one.

purpose in [5] and [7], showing promising results for audio data and natural images. We present in this section a formulation for learning a dictionary in a *supervised* way for regression or classification tasks.

Given a dictionary $\mathbf{D}$ obtained using the approach presented in the previous section, a vector $\mathbf{x}$ in $\mathcal{X} \subseteq \mathbb{R}^m$ can be represented as a sparse vector $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$, defined as the solution of an elastic-net problem [29]:

$$\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D}) \triangleq \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^p} \frac{1}{2}\|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda_1 \|\boldsymbol{\alpha}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\alpha}\|_2^2. \quad (4)$$

We now assume that each signal $\mathbf{x}$ in $\mathcal{X}$ is associated to a variable $\mathbf{y}$ in $\mathcal{Y}$, which we want to predict from $\mathbf{x}$. Concretely, the set $\mathcal{Y}$ can either be *a finite set of labels* in classification tasks, or a *subset of* $\mathbb{R}^q$ *for some integer* q in regression tasks. We can now use the sparse vector $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$ as a feature representation of a signal $\mathbf{x}$ in a classical expected risk minimization formulation:

$$\min_{\mathbf{W} \in \mathcal{W}} f(\mathbf{W}) + \frac{\nu}{2}\|\mathbf{W}\|_{\mathrm{F}}^2, \quad (5)$$

where $\mathbf{W}$ are model parameters which we want to learn, $\mathcal{W}$ is a convex set, ν is a regularization parameter, and f is a convex function defined as

$$f(\mathbf{W}) \triangleq \mathbb{E}_{\mathbf{y},\mathbf{x}}[\ell_s\big(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})\big)]. \quad (6)$$

In this equation, ℓ_s is a convex loss function that measures how well one can predict $\mathbf{y}$ by observing $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$ given the model parameters $\mathbf{W}$. For instance, it can be the square, logistic, or hinge loss from support vector machines (see [32]). The index s of ℓ_s indicates here that the loss is adapted to a *supervised* learning problem. The expectation is taken with respect to the unknown probability distribution $p(\mathbf{y}, \mathbf{x})$ of the data. So far, the dictionary $\mathbf{D}$ is be obtained in an unsupervised way. However, Mairal et al. [9], and Bradley and Bagnell [10] have shown that better results can be achieved when the dictionary is obtained in a fully supervised setting, tuned for the prediction task. We now introduce the task-driven dictionary learning formulation, that consists of *jointly* learning $\mathbf{W}$ and $\mathbf{D}$ by solving

$$\min_{\mathbf{D} \in \mathcal{D}, \mathbf{W} \in \mathcal{W}} f(\mathbf{D}, \mathbf{W}) + \frac{\nu}{2}\|\mathbf{W}\|_{\mathrm{F}}^2, \quad (7)$$

where $\mathcal{D}$ is a set of constraints defined in Eq. (2), and f has the form

$$f(\mathbf{D}, \mathbf{W}) \triangleq \mathbb{E}_{\mathbf{y},\mathbf{x}}[\ell_s\big(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})\big)]. \quad (8)$$

The main difficulty of this optimization problem comes from the non-differentiability of $\boldsymbol{\alpha}^\star$, which is the solution of a nonsmooth optimization problem (4). Bradley and Bagnell [10] have tackled this difficulty by introducing a smooth approximation of the sparse regularization which leads to smooth solutions, allowing the use of implicit differentiation to compute the gradient of the cost function they have introduced. This approximation encourages some coefficients in $\boldsymbol{\alpha}^\star$ to be small, and does not produce true zeros. It can be used when "true" sparsity is not required. In a different formulation, Mairal et al. [9] have used nonsmooth sparse regularization, but used heuristics to tackle the optimization problem. We show in Section 4 that better optimization tools than these heuristics can be used, while keeping a nonsmooth regularization for computing $\boldsymbol{\alpha}^\star$.

A difference between supervised and unsupervised dictionary learning is that overcompleteness—that is, the dictionaries have more elements than the signal dimension, has not empirically proven to be necessary. It is indeed often advocated for image processing applications that having $p > m$ provides better reconstruction results [2], [4], but for discriminative tasks, perfect reconstruction is not always required as long as discriminative features are captured by the sparse coding procedure.

Minor variants of the formulation (7) can also be considered: Non-negativity constraints may be added on $\boldsymbol{\alpha}^\star$ and $\mathbf{D}$, leading to a supervised version of nonnegative matrix factorization [33], regularized with a sparsity-inducing penalty. The function ℓ_s could also take extra arguments such as $\mathbf{D}$ and $\mathbf{x}$ instead of just $\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star$. For simplicity, we have omitted these possibilities, but the formulations and algorithms we present in this paper can easily be extended to these cases.

Before presenting extensions and applications of the formulation we have introduced, let us first discuss the assumptions under which our analysis holds.

3.1.1 Assumptions

From now on, we assume that:

- **(A)** The data $(\mathbf{y}, \mathbf{x})$ admits a probability density p with a compact support $K_\mathcal{Y} \times K_\mathcal{X} \subseteq \mathcal{Y} \times \mathcal{X}$. This is a reasonable assumption in audio, image, and video processing applications, where it is imposed by the data acquisition process, where values returned by sensors are bounded. To simplify the notation we assume from now on that $\mathcal{X}$ and $\mathcal{Y}$ are compact.[2]
- **(B)** When $\mathcal{Y}$ is a subset of a finite-dimensional real vector space, p is continuous and ℓ_s is twice continuously differentiable.
- **(C)** When $\mathcal{Y}$ is a finite set of labels, for all $\mathbf{y}$ in $\mathcal{Y}$, $p(\mathbf{y}, .)$ is continuous and $\ell_s(\mathbf{y}, .)$ is twice continuously differentiable.[3]

Assumptions **(B)** and **(C)** allow us to use several loss functions such as the square, logistic, or softmax losses.

3.2 Extensions

We now present two extensions of the previous formulations. The first one includes a linear transform of the input data, and the second one exploits unlabeled data in a semi-supervised setting.

2. Even though images are acquired in practice after a quantization process, it is a common assumption in image processing to consider pixel values in a continuous space.

3. For a given value of $\mathbf{y}$ and function g, $g(\mathbf{y}, .)$ denotes the function which associates to a vector $\mathbf{x}$ the value $g(\mathbf{y}, \mathbf{x})$.

3.2.1 Learning a Linear Transform of the Input Data

In this section, we add to our basic formulation a linear transform of the input features, represented by a matrix $\mathbf{Z}$. Our motivation for this is twofold: It can be appealing to reduce the dimension of the feature space via such a linear transform, and/or it can make the model richer by increasing the numbers of free parameters. The resulting formulation is the following:

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{W}\in\mathcal{W},\mathbf{Z}\in\mathcal{Z}} f(\mathbf{D},\mathbf{W},\mathbf{Z}) + \frac{\nu_1}{2}\|\mathbf{W}\|_{\mathrm{F}}^2 + \frac{\nu_2}{2}\|\mathbf{Z}\|_{\mathrm{F}}^2, \quad (9)$$

where ν_1 and ν_2 are two regularization parameters, $\mathcal{Z}$ is a convex set and

$$f(\mathbf{D},\mathbf{W},\mathbf{Z}) \triangleq \mathbb{E}_{\mathbf{y},\mathbf{x}}[\ell_s\big(\mathbf{y},\mathbf{W},\boldsymbol{\alpha}^\star(\mathbf{Z}\mathbf{x},\mathbf{D})\big)]. \quad (10)$$

It is worth noticing that the formulations of Eq. (7) and Eq. (9) can also be extended to the case of a cost function depending on several dictionaries involving several sparse coding problems, such as the one used in [8] for signal classification. Such a formulation is not developed here for simplicity reasons, but algorithms to address it can easily be derived from this paper.

3.2.2 Semi-supervised Learning

As shown in [7], sparse coding techniques can be effective for learning good features from unlabeled data. The extension of our task-driven formulation to the semi-supervised learning setting is natural and takes the form

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{W}\in\mathcal{W}} (1-\mu)\mathbb{E}_{\mathbf{y},\mathbf{x}}[\ell_s\big(\mathbf{y},\mathbf{W},\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})\big)] + \mu\mathbb{E}_{\mathbf{x}}[\ell_u(\mathbf{x},\mathbf{D})] + \frac{\nu}{2}\|\mathbf{W}\|_{\mathrm{F}}^2, \quad (11)$$

where the second expectation is taken with respect to the marginal distribution of $\mathbf{x}$. The function ℓ_u is the loss function defined in Eq. (1), and μ in $[0,1]$ is a new parameter controlling the trade-off between the unsupervised and supervised learning cost functions.

3.3 Applications

For illustration purposes, we present a few applications of our task-driven dictionary learning formulations. Our approach is of course not limited to these examples.

3.3.1 Regression

In this setting, $\mathcal{Y}$ is a subset of a q-dimensional real vector space, and the task is to predict variables $\mathbf{y}$ in $\mathcal{Y}$ from the observation of vectors $\mathbf{x}$ in $\mathcal{X}$. A typical application is for instance the restoration of clean signals $\mathbf{y}$ from observed corrupted signals $\mathbf{x}$. Classical signal restoration techniques often focus on removing additive noise or solving inverse linear problems [34]. When the corruption results from an unknown nonlinear transformation, we formulate the restoration task as a general regression problem. This is the case for example in the experiment presented in Section 5.3.

We define the task-driven dictionary learning formulation for regression as follows:

$$\min_{\mathbf{W}\in\mathcal{W},\mathbf{D}\in\mathcal{D}} \mathbb{E}_{\mathbf{y},\mathbf{x}}\Big[\frac{1}{2}\|\mathbf{y}-\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})\|_2^2\Big] + \frac{\nu}{2}\|\mathbf{W}\|_{\mathrm{F}}^2. \quad (12)$$

At test time, when a new signal $\mathbf{x}$ is observed, the estimate of the corresponding variable $\mathbf{y}$ provided by this model is $\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ (plus possibly an intercept which we have omitted here for simplicity reasons). Note that we here propose to use the square loss for estimating the difference between $\mathbf{y}$ and its estimate $\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$, but any other twice differentiable loss can be used.

3.3.2 Binary Classification

In this section and in the next one, we propose to learn dictionaries adapted to classification tasks. Our approach follows the formulation presented in [9], but is slightly different and falls into our task-driven dictionary learning framework. In this setting, the set $\mathcal{Y}$ is equal to $\{-1;+1\}$. Given a vector $\mathbf{x}$, we want to learn the parameters $\mathbf{w}$ in $\mathbb{R}^p$ of a linear model to predict y in $\mathcal{Y}$, using the sparse representation $\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ as features, and jointly optimize $\mathbf{D}$ and $\mathbf{w}$. For instance, using the logistic regression loss, our formulation becomes

$$\min_{\mathbf{w}\in\mathbb{R}^p,\mathbf{D}\in\mathcal{D}} \mathbb{E}_{y,\mathbf{x}}\Big[\log\big(1+e^{-y\mathbf{w}^\top\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})}\big)\Big] + \frac{\nu}{2}\|\mathbf{w}\|_2^2, \quad (13)$$

Once $\mathbf{D}$ and $\mathbf{w}$ have been learned, a new signal $\mathbf{x}$ is classified according to the sign of $\mathbf{w}^\top\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$. For simplicity reasons, we have omitted the intercept in the linear model, but it can easily be included in the formulation. Note that instead of the logistic regression loss, any other twice differentiable loss can be used.

As suggested in [9], it is possible to extend this approach with a bilinear model by learning a matrix $\mathbf{W}$ so that a new vector $\mathbf{x}$ is classified according to the sign of $\mathbf{x}^\top\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$. In this setting, our formulation becomes

$$\min_{\mathbf{W}\in\mathbb{R}^{m\times p},\mathbf{D}\in\mathcal{D}} \mathbb{E}_{y,\mathbf{x}}\Big[\log\big(1+e^{-y\mathbf{x}^\top\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})}\big)\Big] + \frac{\nu}{2}\|\mathbf{W}\|_{\mathrm{F}}^2. \quad (14)$$

This bilinear model requires learning pm parameters as opposed to the p parameters of the linear one. It is therefore richer and can sometimes offer a better classification performance when the linear model is not rich enough to explain the data, but it might be more subject to overfitting.

Note that we have naturally presented the binary classification task using the logistic regression loss, but as we have experimentally observed, the square loss is also an appropriate choice in many situations.

3.3.3 Multi-class Classification

When $\mathcal{Y}$ is a finite set of labels in $\{1,\ldots,q\}$ with $q>2$, extending the previous formulation to the multi-class setting can be done in several ways, which we briefly describe here. The simplest possibility is to use a set of binary classifiers presented in Section 3.3.2 in a "one-vs-all" or "one-vs-one" scheme. Another possibility is

to use a multi-class cost function such as the soft-max function, to find linear predictors $\mathbf{w}_k$, k in $\{1,\ldots,q\}$ such that for a vector $\mathbf{x}$ in $\mathcal{X}$, the quantities $\mathbf{w}_y^\top \boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ are encouraged to be greater than $\mathbf{w}_k^\top \boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ for all $k \neq y$. Another possibility is to turn the multi-class classification problem into a regression one and consider that $\mathcal{Y}$ is a set of q binary vectors of dimension q such that the $k-$th vector has 1 on its k-th coordinate, and 0 elsewhere. This allows using the regression formulation of Section 3.3.1 to solve the classification problem.

We remark that for classification tasks, scalability issues should be considered when choosing between a one-vs-all scheme (learning independent dictionaries for every class) and using a multi-class loss function (learning a single dictionary shared between all classes). The one-vs-all scheme requires keeping into memory qpm parameters, where q is the number of classes, which is feasible when q is reasonably small. For classifications problems with many classes (for instance $q \geq 1\,000$), using a single (larger) dictionary and a multi-class loss function is more appropriate, and would in addition allow feature sharing between the classes.

3.3.4 Compressed sensing

Let us consider a signal $\mathbf{x}$ in $\mathbb{R}^m$, the theory of compressed sensing [35], [36] tells us that under certain assumptions, the vector $\mathbf{x}$ can be recovered exactly from a few measurements $\mathbf{Z}\mathbf{x}$, where $\mathbf{Z}$ in $\mathbb{R}^{r\times m}$ is called a "sensing" matrix with $r \ll m$. Unlike classical signal processing methods, such a linear transformation is sometimes included physically in the data acquisition process itself [37], meaning that a sensor can provide measurements $\mathbf{Z}\mathbf{x}$ without directly measuring $\mathbf{x}$.

In a nutshell, the recovery of $\mathbf{x}$ has been proven to be possible when $\mathbf{x}$ admits a sparse representation on a dictionary $\mathbf{D}$, and the sensing matrix $\mathbf{Z}$ is incoherent with $\mathbf{D}$, meaning that the rows of $\mathbf{Z}$ are sufficiently uncorrelated with the columns of $\mathbf{D}$ (see [35], [36] for more details).[4] To ensure that this condition is satisfied, $\mathbf{Z}$ is often chosen as a random matrix, which is incoherent with any dictionary with high probability.

The choice of a random matrix is appealing for many reasons. In addition to the fact that it provides theoretical guarantees of incoherence, it is well suited to the case where m is large, making it impossible to store a deterministic matrix $\mathbf{Z}$ into memory, whereas it is sufficient to store the seed of a random process to generate a random matrix. On the other hand, large signals can often be cut into smaller parts that still admit sparse decompositions, e.g., image patches, which can be treated independently with a deterministic smaller matrix $\mathbf{Z}$. When this is the case or when m has a reasonable size, the question of whether to use a deterministic matrix $\mathbf{Z}$ or a random one arises, and it has been empirically observed that learned matrices $\mathbf{Z}$ can outperform random projections: For example, it is shown in [38] that classical dimensionality reduction techniques such as principal component analysis (PCA) or independent component analysis (ICA) could do better than random projections in noisy settings, and in [16] that jointly learning sensing matrices and dictionaries can do even better in certain cases. A Bayesian framework for learning sensing matrices in compressed sensing applications is also proposed in [39].

Following the latter authors, we study the case where $\mathbf{Z}$ is not random but learned at the same time as the dictionary, and introduce a formulation which falls into out task-driven dictionary learning framework:

$$\min_{\substack{\mathbf{D}\in\mathcal{D}\\ \mathbf{W}\in\mathbb{R}^{m\times p}\\ \mathbf{Z}\in\mathbb{R}^{r\times m}}} \mathbb{E}_{\mathbf{y},\mathbf{x}}\Big[\frac{1}{2}\|\mathbf{y}-\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{Z}\mathbf{x},\mathbf{D})\|_2^2\Big]+\frac{\nu_1}{2}\|\mathbf{W}\|_{\mathrm{F}}^2+\frac{\nu_2}{2}\|\mathbf{Z}\|_{\mathrm{F}}^2, \tag{15}$$

where we learn $\mathbf{D}$, $\mathbf{W}$ and $\mathbf{Z}$ so that the variable $\mathbf{y}$ should be well reconstructed when encoding the "sensed" signal $\mathbf{Z}\mathbf{x}$ with a dictionary $\mathbf{D}$. In a noiseless setting, $\mathbf{y}$ is naturally set to the same value as $\mathbf{x}$. In a noisy setting, it can be a corrupted version of $\mathbf{x}$.

After having presented our general task-driven dictionary learning formulation, we present next a strategy to address the corresponding nonconvex optimization problem.

4 Optimization

We first show that the cost function f of our basic formulation (7) is differentiable and compute its gradient. Then, we refine the analysis for the different variations presented in the previous section, and describe an efficient online learning algorithm to address them.

4.1 Differentiability of f

We analyze the differentiability of f as defined in Eq. (7) with respect to its two arguments $\mathbf{D}$ and $\mathbf{W}$. We consider here the case where $\mathcal{Y}$ is a compact subset of a finite dimensional real vector space, but all proofs and formulas are similar when $\mathcal{Y}$ is a finite set of labels. The purpose of this section is to show that even though the sparse coefficients $\boldsymbol{\alpha}^\star$ are obtained by solving a non-differentiable optimization problem, f is differentiable on $\mathcal{W}\times\mathcal{D}$, and one can compute its gradient.

The main argument in the proof of Propositions 1 and 2 below is that, although the function $\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ is not differentiable, it is uniformly Lipschitz continuous, and differentiable almost everywhere. The only points where $\boldsymbol{\alpha}^\star$ is not differentiable are points where the set of nonzero coefficients of $\boldsymbol{\alpha}^\star$ change (we always denote this set by Λ in this paper). Considering optimality conditions of the elastic-net formulation of Eq. (1), these points are easy to characterize. The details of the proof have been relegated to the Appendix (Lemma 1 and Proposition 3) for readability purposes. With these results in hand, we then show that f admits a first-order

4. The assumption of "incoherence" between $\mathbf{D}$ and $\mathbf{Z}$ can be replaced with a different but related hypothesis called *restricted isometry property*. Again the reader should refer to [35], [36] for more details.

Taylor expansion meaning that it is differentiable, the sets where $\boldsymbol{\alpha}^\star$ is not differentiable being negligible in the expectation from the definition of f in Eq. (8). We can now state our main result:

Proposition 1 (Differentiability and gradients of f): Assume $\lambda_2 > 0$, (**A**), (**B**) and (**C**). Then, the function f defined in Eq. (7) is differentiable, and

$$\begin{cases} \nabla_{\mathbf{W}} f(\mathbf{D}, \mathbf{W}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}[\nabla_{\mathbf{W}} \ell_s(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star)], \\ \nabla_{\mathbf{D}} f(\mathbf{D}, \mathbf{W}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}[-\mathbf{D}\boldsymbol{\beta}^\star \boldsymbol{\alpha}^{\star\top} + (\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}^\star)\boldsymbol{\beta}^{\star\top}], \end{cases} \tag{16}$$

where $\boldsymbol{\alpha}^\star$ is short for $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$, and $\boldsymbol{\beta}^\star$ is a vector in $\mathbb{R}^p$ that depends on $\mathbf{y}, \mathbf{x}, \mathbf{W}, \mathbf{D}$ with

$$\boldsymbol{\beta}^\star_{\Lambda^C} = 0 \quad \text{and} \quad \boldsymbol{\beta}^\star_\Lambda = (\mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda + \lambda_2 \mathbf{I})^{-1} \nabla_{\boldsymbol{\alpha}_\Lambda} \ell_s(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star), \tag{17}$$

where Λ denotes the indices of the nonzero coefficients of $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$.

The proof of this proposition is given in Appendix. We have shown that the function defined in Eq. (7) is smooth, and computed its gradients. The same can be done for the more general formulation of Eq. (10):

Proposition 2 (Differentiability, extended formulation): Assume $\lambda_2 > 0$, (**A**), (**B**) and (**C**). Then, the function f defined in Eq. (10) is differentiable. The gradients of f are

$$\begin{cases} \nabla_{\mathbf{W}} f(\mathbf{D}, \mathbf{W}, \mathbf{Z}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}[\nabla_{\mathbf{W}} \ell_s(\mathbf{y}, \mathbf{W}, \boldsymbol{\alpha}^\star)], \\ \nabla_{\mathbf{D}} f(\mathbf{D}, \mathbf{W}, \mathbf{Z}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}[-\mathbf{D}\boldsymbol{\beta}^\star \boldsymbol{\alpha}^{\star\top} + (\mathbf{Z}\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}^\star)\boldsymbol{\beta}^{\star\top}], \\ \nabla_{\mathbf{Z}} f(\mathbf{D}, \mathbf{W}, \mathbf{Z}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}[\mathbf{D}\boldsymbol{\beta}^\star \mathbf{x}^\top], \end{cases} \tag{18}$$

where $\boldsymbol{\alpha}^\star$ is short for $\boldsymbol{\alpha}^\star(\mathbf{Z}\mathbf{x}, \mathbf{D})$, and $\boldsymbol{\beta}^\star$ is defined in Eq. (17).

The proof is similar to the one of Proposition 1 in Appendix, and uses similar arguments.

4.2 Algorithm

Stochastic gradient descent algorithms are typically designed to minimize functions whose gradients have the form of an expectation as in Eq. (16). They have been shown to converge to stationary points of (possibly nonconvex) optimization problems under a few assumptions that are a bit stricter than the ones satisfied in this paper (see [31] and references therein).[5] As noted in [19], these algorithms are generally well suited to unsupervised dictionary learning when their learning rate is well tuned.

The method we propose here is a projected first-order stochastic gradient algorithm (see [40]), and it is given in Algorithm 1. It sequentially draws i.i.d samples $(\mathbf{y}_t, \mathbf{x}_t)$ from the probability distribution $p(\mathbf{y}, \mathbf{x})$. Obtaining such i.i.d. samples may be difficult since the density $p(\mathbf{y}, \mathbf{x})$ is unknown. At first approximation, the vectors $(\mathbf{y}_t, \mathbf{x}_t)$ are obtained in practice by cycling over a randomly permuted training set, which is often done in similar machine learning settings [41].

Algorithm 1 Stochastic gradient descent algorithm for task-driven dictionary learning.

Require: $p(\mathbf{y}, \mathbf{x})$ (a way to draw i.i.d samples of p), $\lambda_1, \lambda_2, \nu \in \mathbb{R}$ (regularization parameters), $\mathbf{D} \in \mathcal{D}$ (initial dictionary), $\mathbf{W} \in \mathcal{W}$ (initial parameters), T (number of iterations), t_0, ρ (learning rate parameters).

1: **for** $t = 1$ to T **do**
2: Draw $(\mathbf{y}_t, \mathbf{x}_t)$ from $p(\mathbf{y}, \mathbf{x})$.
3: Sparse coding: compute $\boldsymbol{\alpha}^\star$ using a modified LARS [42].

$$\boldsymbol{\alpha}^\star \leftarrow \operatorname*{arg\,min}_{\boldsymbol{\alpha} \in \mathbb{R}^p} \frac{1}{2}||\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}||_2^2 + \lambda_1 ||\boldsymbol{\alpha}||_1 + \frac{\lambda_2}{2}||\boldsymbol{\alpha}||_2^2.$$

4: Compute the active set:

$$\Lambda \leftarrow \{j \in \{1, \ldots, p\} : \boldsymbol{\alpha}^\star[j] \neq 0\}.$$

5: Compute $\boldsymbol{\beta}^\star$: Set $\boldsymbol{\beta}^\star_{\Lambda^C} = 0$ and

$$\boldsymbol{\beta}^\star_\Lambda = (\mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda + \lambda_2 \mathbf{I})^{-1} \nabla_{\boldsymbol{\alpha}_\Lambda} \ell_s(\mathbf{y}_t, \mathbf{W}, \boldsymbol{\alpha}^\star).$$

6: Choose the learning rate $\rho_t \leftarrow \min\left(\rho, \rho \frac{t_0}{t}\right)$.
7: Update the parameters by a projected gradient step

$$\mathbf{W} \leftarrow \Pi_{\mathcal{W}}\Big[\mathbf{W} - \rho_t\big(\nabla_{\mathbf{W}} \ell_s(\mathbf{y}_t, \mathbf{W}, \boldsymbol{\alpha}^\star) + \nu \mathbf{W}\big)\Big],$$

$$\mathbf{D} \leftarrow \Pi_{\mathcal{D}}\Big[\mathbf{D} - \rho_t\big(-\mathbf{D}\boldsymbol{\beta}^\star \boldsymbol{\alpha}^{\star\top} + (\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}^\star)\boldsymbol{\beta}^{\star\top}\big)\Big],$$

where $\Pi_{\mathcal{W}}$ and $\Pi_{\mathcal{D}}$ are respectively orthogonal projections on the sets $\mathcal{W}$ and $\mathcal{D}$.
8: **end for**
9: **return** $\mathbf{D}$ (learned dictionary).

At each iteration, the sparse code $\boldsymbol{\alpha}^\star(\mathbf{x}_t, \mathbf{D})$ is computed by solving the elastic-net formulation of [29]. We have chosen to use the LARS algorithm, a homotopy method [42], which was originally developed to solve the Lasso formulation—that is, $\lambda_2 = 0$, but which can be modified to solve the elastic-net problem. Interestingly, it admits an efficient implementation that provides a Cholesky decomposition of the matrix $(\mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda + \lambda_2 \mathbf{I})^{-1}$ (see [29], [42]) as well as the solution $\boldsymbol{\alpha}^\star$. In this setting, $\boldsymbol{\beta}^\star$ can be obtained without having to solve from scratch a new linear system.

The learning rate ρ_t is chosen according to a heuristic rule. Several strategies have been presented in the literature (see [28], [43] and references therein). A classical setting uses a learning rate of the form ρ/t, where ρ is a constant.[6] However, such a learning rate is known to

5. As often done in machine learning, we use stochastic gradient descent in a setting where it is not guaranteed to converge in theory, but is has proven to behave well in practice, as shown in our experiments. The convergence proof of Bottou [31] for non-convex problems indeed assumes three times differentiable cost functions.

6. A $1/t$-asymptotic learning rate is usually used for proving the convergence of stochastic gradient descent algorithms [31].

decrease too quickly in many practical cases, and one sometimes prefers a learning rate of the form $\rho/(t+t_0)$, which requires tuning two parameters. In this paper, we have chosen a learning rate of the form $\min(\rho, \rho t_0/t)$—that is, a constant learning rate ρ during t_0 iterations, and a $1/t$ annealing strategy when $t > t_0$, a strategy used by [43] for instance. Finding good parameters ρ and t_0 also requires in practice a good heuristic. The one we have used successfully in all our experiments is $t_0 = T/10$, where T is the total number of iterations. Then, we try several values of ρ during a few hundreds of iterations and keep the one that gives the lowest error on a small validation set.

In practice, one can also improve the convergence speed of our algorithm with a mini-batch strategy—that is, by drawing $\eta > 1$ samples at each iteration instead of a single one. This is a classical heuristic in stochastic gradient descent algorithms and, in our case, this is further motivated by the fact that solving η elastic-net problems with the same dictionary $\mathbf{D}$ can be accelerated by the precomputation of the matrix $\mathbf{D}^\top\mathbf{D}$ when η is large enough. Such a strategy is also used in [19] for the classical data-driven dictionary learning approach. In practice, the value $\eta = 200$ has given good results in all our experiments (a value found to be good for the unsupervised setting as well).

As many algorithms tackling non-convex optimization problems, our method for learning supervised dictionaries can lead to poor results if is not well initialized. The classical unsupervised approach of dictionary learning presented in Eq. (3) has been found empirically to be better behaved than the supervised one, and easy to initialize [19]. We therefore have chosen to initialize our dictionary $\mathbf{D}$ by addressing the unsupervised formulation of Eq. (3) using the SPAMS toolbox [19].[7] With this initial dictionary $\mathbf{D}$ in hand, we optimize with respect to $\mathbf{W}$ the cost function of Eq (5), which is convex. This procedure gives us a pair $(\mathbf{D}, \mathbf{W})$ of parameters which are used to initialize Algorithm 1.

4.3 Extensions

We here present the slight modifications to Algorithm 1 necessary to address the two extensions discussed in Section 3.2.

The last step of Algorithm 1 updates the parameters $\mathbf{D}$ and $\mathbf{W}$ according to the gradients presented in Eq. (18). Modifying the algorithm to address the formulation of Section 3.2.1 also requires updating the parameters $\mathbf{Z}$ according to the gradient from Proposition 2:

$$\mathbf{Z} \leftarrow \Pi_{\mathcal{Z}}\Big[\mathbf{Z} - \rho_t(\mathbf{D}\boldsymbol{\beta}^\star\mathbf{x}^\top + \nu_2\mathbf{Z})\Big],$$

where $\Pi_{\mathcal{Z}}$ denotes the orthogonal projection on the set $\mathcal{Z}$.

The extension to the semi-supervised formulation of Section 3.2.2 assumes that one can draw samples from the marginal distribution $p(\mathbf{x})$. This is done in practice by cycling over a randomly permuted set of unlabeled vectors. Extending Algorithm 1 to this setting requires the following modifications: At every iteration, we draw one pair $(\mathbf{y}_t, \mathbf{x}_t)$ from $p(\mathbf{y}, \mathbf{x})$ and one sample $\mathbf{x}'_t$ from $p(\mathbf{x})$. We proceed exactly as in Algorithm 1, except that we also compute $\boldsymbol{\alpha}^{\star\prime} \triangleq \boldsymbol{\alpha}^\star(\mathbf{x}'_t, \mathbf{D})$, and replace the update of the dictionary $\mathbf{D}$ by

$$\mathbf{D} \leftarrow \Pi_{\mathcal{D}}\Big[\mathbf{D} - \rho_t\Big((1-\mu)\big(-\mathbf{D}\boldsymbol{\beta}^\star\boldsymbol{\alpha}^{\star\top} + (\mathbf{x}_t - \mathbf{D}\boldsymbol{\alpha}^\star)\boldsymbol{\beta}^{\star\top}\big) + \mu\big(-(\mathbf{x}'_t - \mathbf{D}\boldsymbol{\alpha}^{\star\prime})\boldsymbol{\alpha}^{\star\prime\top}\big)\Big)\Big], \quad (19)$$

where the term $-(\mathbf{x}'_t - \mathbf{D}\boldsymbol{\alpha}^{\star\prime})\boldsymbol{\alpha}^{\star\prime\top}$ is in fact the gradient $\nabla_{\mathbf{D}}\ell_u(\mathbf{x}_t, \mathbf{D})$, as shown in [19].

5 Experimental Validation

Before presenting our experiments, we briefly discuss the question of choosing the parameters in our formulation.

5.1 Choosing the Parameters

Performing cross-validation on the parameters λ_1, λ_2 (elastic-net parameters), ν (regularization parameter) and p (size of the dictionary) would of course be cumbersome, and we use a few simple heuristics to either reduce the search space for these parameters or fix arbitrarily some of them. We have proceeded in the following way:

- Since we want to exploit sparsity, we often set λ_2 to 0, even though $\lambda_2 > 0$ is necessary in our analysis for proving the differentiability of our cost function. This has proven to give satisfactory results in most experiments, except for the experiment of Section 5.5, where choosing a small positive value for λ_2 was necessary for our algorithm to converge.
- We have empirically observed that natural image patches (that are preprocessed to have zero-mean and unit ℓ_2-norm) are usually well reconstructed with values of λ_1 around 0.15 (a value used in [9] for instance), and that one only needs to test a few different values, for instance $\lambda_1 = 0.15 + 0.025k$, with $k \in \{-3, \ldots, 3\}$.
- When there is a lot of training data, which is often the case for natural image patches, the regularization with ν becomes unnecessary and this parameter can arbitrarily set to a small value, e.g., $\nu = 10^{-9}$ for normalized input data. When there are not many training points, this parameter is set up by cross-validation.
- We have also observed that a larger dictionary usually means a better performance, but a higher computational cost. Setting the size of the dictionary is therefore often a trade-off between results quality and efficiency. In our experiments, we often try the values p in $\{50, 100, 200, 400\}$.

We show in this section several applications of our method to real problems, starting with handwritten digits classification, then moving to the restoration of images damaged by an unknown nonlinear transformation, digital art authentification, and compressed sensing.

7. http://www.di.ens.fr/willow/SPAMS/

5.2 Handwritten Digits Classification

We consider here a classification task using the MNIST [44] and USPS [45] handwritten datasets. MNIST contains $70\,000$ 28×28 images, $60\,000$ for training, $10\,000$ for testing, whereas USPS has $7\,291$ training images and $2\,007$ test images of size 16×16.

We address this multiclass classification problem with a one-vs-all strategy, learning independently one dictionary and classifier per class, using the formulation of Section 3.3.2. This approach has proven here to be faster than learning a single large dictionary with a multi-class loss function, while providing very good results. In this experiment, the Lasso [30] is preferred to the elastic-net formulation [29], and λ_2 is thus set to 0. All digits are preprocessed to have zero-mean and are normalized to have unit ℓ_2-norm. For the reasons mentioned earlier, we try the parameters $\lambda_1 = 0.15 + 0.025k$, with $k \in \{-3, \ldots, 3\}$, and and ν is chosen in $\{10^{-1}, \ldots, 10^{-6}\}$. We select the parameters on MNIST by keeping the last $10\,000$ digits of the training set for validation, while training on the first $50\,000$ ones. For USPS, we similarly keep 10% of the training set for validation. Note that a cross-validation scheme may give better results, but would be computationally more expensive.

Most effective digit recognition techniques use features with shift invariance properties [24], [46]. Since our formulation is less sophisticated than for instance the convolutional network architecture of [24] and does not enjoy such properties, we have artificially augmented the size of the training set by considering versions of the digits that are shifted by one pixel in every direction. This is of course not an optimal way of introducing shift invariance in our framework, but it is fairly simple.

After choosing the parameters using the validation set, we retrain our model on the full training set. Each experiment is performed with $40\,000$ iterations of our algorithm with a mini-batch of size 200. We report the performance on the test set achieved for different dictionary sizes, with p in $\{50, 100, 200, 300\}$ for the two datasets, and observe that learning $\mathbf{D}$ in a supervised way significantly improves the performance of the classification. Moreover our method achieves state-of-the-art results on MNIST with a 0.54% error rate, which is similar to the 0.60% error rate of [24].[8] Our 2.84% error rate on USPS is slightly behind the 2.4% error rate of [46].

We remark that a digit recognition task was also carried out in [9], where a similar performance is reported.[9] Our conclusions about the advantages of supervised versus unsupervised dictionary learning are consistent with [9], but our approach has two main advantages. First it is much easier to use since it does not requires complicated heuristic procedures to select the parameters, and second it applies to a wider spectrum of applications such as to regression tasks.

8. It is also shown in [24] that better results can be achieved by considering deformations of the training set.

9. The error rates in [9] are slightly higher but the dataset used in their paper is not augmented with shifted versions of the digits.

TABLE 1
Test error in percent of our method for the digit recognition task for different dictionary sizes p.

D	unsupervised				supervised			
p	50	100	200	300	50	100	200	300
MNIST	5.27	3.92	2.95	2.36	.96	.73	.57	.54
USPS	8.02	6.03	5.13	4.58	3.64	3.09	2.88	2.84

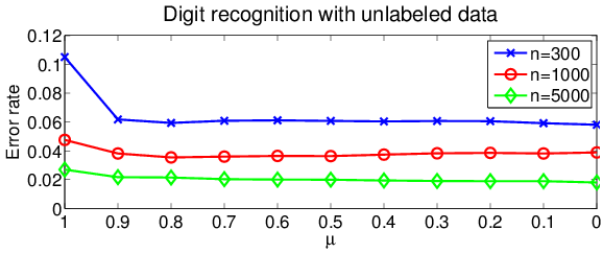

Fig. 1. Error rates on MNIST when using n labeled data, for various values of μ.

Our second experiment follows [24], where only a few samples are labelled. We use the semi-supervised formulation of Section 3.2.2 which exploits unlabeled data. Unlike the first experiment where the parameters are chosen using a validation set, and following [24], we make a few arbitrary choices. Indeed, we use $p = 300$, $\lambda_1 = 0.075$, and $\nu = 10^{-5}$, which were the parameters chosen in the previous experiment. As in the previous experiment, we have observed that these parameters lead to sparse vectors $\boldsymbol{\alpha}^\star$ with about 15 non-zero coefficients. The dictionaries associated with each digit class are initialized using the unsupervised formulation of Section 2. To test our algorithm with different values of μ, we use a continuation strategy: Starting with $\mu = 1.0$, we sequentially decrease its value by 0.1 until we have $\mu = 0$, learning with $10\,000$ iterations for each new value of μ. We report the error rates in Figure 1, showing that our approach offers a competitive performance similar to [24]. The best error rates of our method for $n = 300, 1000, 5000$ labeled data are respectively $5.81, 3.55$ and 1.81%, which is similar to [24] who has reported $7.18, 3.21$ and 1.52% with the same sets of labeled data.

5.3 Learning a Nonlinear Image Mapping

We now illustrate our method in a regression context by considering a classical image processing task called "inverse halftoning". With the development of several *binary* display technologies in the 70s (including, for example, printers and PC screens), the problem of converting a grayscale continuous-tone image into a binary one that looks perceptually similar to the original one ("halftoning") was posed to the image processing community. Examples of halftoned images obtained with the classical Floyd-Steinberg algorithm [47] are presented in the second column of Figure 2, with original images in the first column. Restoring these binary images to

continuous-tone ones ("inverse halftoning") has become a classical problem (see [48] and references therein).

Unlike most image processing approaches that explicitly model the halftoning process, we formulate it as a regression problem, *without exploiting any prior on the task*. We use a database of 36 images; 24 are high-quality images from the Kodak PhotoCD dataset[10] and are used for training, and 12 are classical images often used for evaluating image processing algorithms;[11] the first four (house, peppers, cameraman, lena) are used for validation and the remaining eight for testing.

We apply the Floyd-Steinberg algorithm implemented in the LASIP Matlab toolbox[12] to the grayscale continuous-tone images in order to build our training/validation/testing set. We extract all pairs of patches from the original/halftoned images in the training set, which provides us with a database of approximately 9 millions of patches. We then use the "signal regression" formulation of Eq. (12) to learn a dictionary $\mathbf{D}$ and model parameters $\mathbf{W}$, by performing two passes of our algorithm over the 9 million training pairs.

At this point, we have learned how to restore a small patch from an image, but not yet how to restore a full image. Following other patch-based approaches to image restoration [2], we extract from a test image all patches including overlaps, and restore each patch *independently* so that we get different estimates for each pixel (one estimate for each patch the pixel belongs to). These estimates are then averaged to reconstruct the full image, which has proven to give very good results in many image restoration tasks (see, e.g., [2], [4]). The final image is then post-processed using the denoising algorithm of [4] to remove possible artefacts.

We then measure how well it reconstructs the continuous-tone images from the halftoned ones in the test set. To reduce the number of hyperparameters, we have made a few arbitrary choices: We first use the Lasso formulation for encoding the signals—that is, we set $\lambda_2 = 0$. With millions of training samples, our model is unlikely to overfit and the regularization parameter ν is set to 0 as well. The remaining free parameters are the size m of the patches, the size p of the dictionary and the regularization parameter λ_1. These parameters are selected by minimizing the mean-squared error reconstruction on the validation set. We have tried patches of size $m = l \times l$, with $l \in \{6, 8, 10, 12, 14, 16\}$, dictionaries of sizes $p = 100$, 250 and 500 , and determined λ_1 by first trying values on the logarithmic scale 10^i, $i = -3, 2$, then refining this parameter on the scale $0.1, 0.2, 0.3, \ldots, 1.0$. The best parameters found are $m = 10 \times 10$, $p = 500$ and $\lambda_1 = 0.6$. Since the test procedure is slightly different from the training one (the test includes an averaging step to restore a full image whereas the training one does not), we have refined the value of λ_1, trying different values one an additive scale $\{0.4, 0.45, \ldots, 0.75, 0.8\}$, and selected the value $\lambda_1 = 0.55$, which has given the best result on the validation set.

Note that the largest dictionary has been chosen, and better results could potentially be obtained using an even larger dictionary, but this would imply a higher computational cost. Examples of results are presented in Figure 2. Halftoned images are binary but look perceptually similar to the original image. Reconstructed images have very few artefacts and most details are well preserved. We report in Table 2 a quantitative comparison between our approach and various ones from the literature, including the state-of-the-art algorithm of [48], which had until now the best results on this dataset. Even though our method does not explicitly model the transformation, it leads to better results in terms of PSNR.[13] We also present in Figure 3 the results obtained by applying our algorithm to various binary images found on the web, from which we do not know the ground truth, and which have not necessarily been obtained with the Floyd-Steinberg algorithm. The results are qualitatively rather good.

From this experiment, we conclude that our method is well suited to (at least, some) nonlinear regression problems on natural images, paving the way to new applications of sparse image coding.

5.4 Digital Art Authentification

Recognizing authentic paintings from imitations using statistical techniques has been the topic of a few recent works [52], [53], [54]. Classical methods compare, for example, the kurtosis of wavelet coefficients between a set of authentic paintings and imitations [52], or involve more sophisticated features [53]. Recently, Hugues et al. [54] have considered a dataset of 8 authentic paintings from Pieter Bruegel the Elder, and 5 imitations.[14] They have proposed to learn dictionaries for sparse coding, and use the kurtosis of the sparse coefficients as discriminative features. We use their dataset, which they kindly provided to us.[15]

The supervised dictionary learning approach we have presented is designed for classifying relatively small signals, and should not be directly applicable to the classification of large images, for which classical computer vision approaches based on bags of words may be better adapted (see [12], [55] for such approaches). However, we show that, for this particular dataset, a simple voting scheme based on the classification of small image patches with our method leads to good results.

The experiment we carry out consists of finding which painting is authentic, and which one is fake, in a pair

10. http://r0k.us/graphics/kodak/

11. The list of these images can be found in [4], where they are used for the problem of image denoising.

12. http://www.cs.tut.fi/~lasip/

13. Denoting by MSE the mean-squared-error for images whose intensities are between 0 and 255, the PSNR is defined as PSNR $= 10 \log_{10}(255^2/\text{MSE})$ and is measured in dB. A gain of 1dB reduces the MSE by approximately 20%.

14. The origin of these paintings is assessed by art historians.

15. It would have been interesting to use the datasets used in [52], [53], but they are not publicly available.

TABLE 2
Inverse halftoning experiments. Results are reported in PSNR (higher is better). SA-DCT refers to [48], LPA-ICI to [49], FIHT2 to [50] and WInHD to [51]. Best results for each image are in bold.

	Validation set				Test set							
Image	1	2	3	4	5	6	7	8	9	10	11	12
FIHT2	30.8	25.3	25.8	31.4	24.5	28.6	29.5	28.2	29.3	26.0	25.2	24.7
WInHD	31.2	26.9	26.8	31.9	25.7	29.2	29.4	28.7	29.4	28.1	25.6	26.4
LPA-ICI	31.4	27.7	26.5	32.5	25.6	29.7	30.0	29.2	30.1	28.3	26.0	27.2
SA-DCT	32.4	28.6	27.8	**33.0**	**27.0**	30.1	30.2	29.8	30.3	28.5	**26.2**	27.6
Ours	**33.0**	**29.6**	**28.1**	**33.0**	26.6	**30.2**	**30.5**	**29.9**	**30.4**	**29.0**	**26.2**	**28.0**

Fig. 2. From left to right: Original images, halftoned images, reconstructed images. Even though the halftoned images (center column) perceptually look relatively close to the original images (left column), they are binary. Reconstructed images (right column) are obtained by restoring the halftoned binary images. Best viewed by zooming on a computer screen.

Fig. 3. Results on various binary images publicly available on the Internet. No ground truth is available for these images from old computer games, and the algorithm that has generated these images is unknown. Input images are on the left. Restored images are on the right. Best viewed by zooming on a computer screen.

known to contain one of each.[16] We proceed in a leave-one-out fashion, where we remove for testing one authentic and one imitation paintings from the dataset, and learn on the remaining ones. Since the dataset is small and does not have an official training/test set, we measure a cross-validation score, testing all possible pairs. We consider 12×12 color image patches, without any pre-processing, and classify each patch from the test images independently. Then, we use a simple majority vote among the test patches to decide which image is the authentic one in the pair test, and which one is the imitation.[17]

For each pair of authentic/imitation paintings, we build a dataset containing $200\,000$ patches from the authentic images and $200\,000$ from the imitations. We use the formulation from Eq. (13) for binary classification, and arbitrarily choose dictionaries containing $p = 100$ dictionary elements. Since the training set is large, we set the parameter ν to 0, also choose the Lasso formulation for decomposing the patches by setting $\lambda_2 = 0$, and cross-validate on the parameter λ_1, trying values on a grid $\{10^{-4}, 10^{-3}, \ldots, 10^{0}\}$, and then refining the result on a grid with a logarithmic scale of 2. We also compare

16. This task is of course considerably easier than classifying each painting as authentic or fake. We do not claim to propose a method that readily applies to forgery detection.

17. Note that this experimental setting is different from [54], where only authentic paintings are used for training (and not imitations). We therefore do not make quantitive comparison with this work.

Eq. (13) with the logistic regression loss and the basic formulation of Eq. (5) where $\mathbf{D}$ is learned unsupervised.

For classifying individual patches, the cross-validation score of the supervised formulation is a classification rate of $54.04 \pm 2.26\%$, which slightly improves upon the "unsupervised" one that achieves $51.94 \pm 1.92\%$. The task of classifying independently small image patches is difficult since there is significant overlap between the two classes. On the other hand, finding the imitation in a pair of (authentic,imitation) paintings with the voting scheme is easier and the "unsupervised formulation" only fails for one pair, whereas the supervised one has always given the right answer in our experiments.

5.5 Compressed Sensing

In this experiment, we apply our method to the problem of learning dictionaries and projection matrices for compressed sensing. As explained in Section 3.3.4, our formulation and the conclusions of this section hold for relatively small signals, where the sensing matrix can be stored into memory and learned. Thus, we consider here small image patches of natural images of size $m = 10 \times 10$ pixels. To build our training/validation/test set, we have chosen the Pascal VOC 2006 database of natural images [56]: Images 1 to 3000 are used for training; images 3001 to 4000 are used for validation, and the remaining 1304 images are kept for testing. Images are downsampled by a factor 2 so that the JPEG compression artefacts present in this dataset become visually imperceptible, thereby improving its quality for our experiment.

We compare different settings where the task is to reconstruct the patches $\mathbf{y}$ of size $m = 10 \times 10$, from an observation $\mathbf{Zx}$ of size $r \ll m$ (for instance $r = 20$ linear measurements), where $\mathbf{Z}$ in $\mathbb{R}^{r \times m}$ is a sensing matrix. For simplicity reasons, we only consider here the noiseless case, where $\mathbf{y} = \mathbf{x}$. At test time, as explained in Section (3.3.4), our estimate of $\mathbf{y}$ is $\mathbf{W}\boldsymbol{\alpha}^\star(\mathbf{Zx}, \mathbf{D})$, where $\mathbf{D}$ in $\mathbb{R}^{r \times p}$ is a dictionary for representing $\mathbf{Zx}$, and $\mathbf{W}$ in $\mathbb{R}^{m \times p}$ can be interpreted here as a dictionary for representing $\mathbf{y}$. We evaluate several strategies for obtaining $(\mathbf{Z}, \mathbf{D}, \mathbf{W})$:

- We consider the case, which we call RANDOM, where the entries of $\mathbf{Z}$ are i.i.d. samples of the Gaussian distribution $\mathcal{N}(0, 1/\sqrt{m})$. Since the purpose of $\mathbf{Z}$ is to reduce the dimensionality of the input data, it is also natural to consider the case where $\mathbf{Z}$ is obtained by principal component analysis on natural image patches (PCA). Finally, we also learn $\mathbf{Z}$ with the supervised learning formulation of Eq. (15), (SL), but consider the case where it is initialized randomly (SL1) or by PCA (SL2).
- The matrix $\mathbf{D}$ can either be fixed or learned. A typical setting would be to set $\mathbf{D} = \mathbf{ZD}'$, where $\mathbf{D}'$ is discrete-cosine-transform matrix (DCT), often used in signal processing applications [2]. It can also be learned with an unsupervised learning formulation (UL), or a supervised one (SL).
- $\mathbf{W}$ is always learned in a supervised way.

Since our training set is very large (several millions of patches), we arbitrarily set the regularization parameters ν_1 and ν_2 to 0. We measure reconstruction errors with dictionaries of various levels of overcompleteness by choosing a size p in $\{100, 200, 400\}$. The remaining free parameters are the regularization parameters λ_1 and λ_2 for obtaining the coefficients $\boldsymbol{\alpha}^\star$. We try the values $\lambda_1 = 10^i$, with i in $\{-5, \ldots, 0\}$. Unlike what we have done in the experiments of Section 5.3, it is absolutely necessary in this setting to use $\lambda_2 > 0$ (according to the theory), since using a zero value for this parameter has led to instabilities and prevented our algorithm from converging. We have tried the values $\lambda_2 = 10^i \lambda_1$, with i in $\{-2, -1, 0\}$. Each learning procedure is performed by our algorithm in one pass on 10 millions of patches randomly extracted from our training images. The pair of parameters (λ_1, λ_2) that gives the lowest reconstruction error on the validation set is selected, and we report in Table 3 the result obtained on a test set of $500\,000$ patches randomly extracted from the 1304 test images. The conclusions of this compressed sensing experiment on natural image patches are the following:

- When $\mathbf{Z}$ is initialized as a Gaussian random matrix (case RANDOM), learning $\mathbf{D}$ and $\mathbf{Z}$ significantly improves the reconstruction error (case SL1). A similar observation was made in [16].
- Results obtained with PCA are in general much better than those obtained with random projections, which is consistent with the conclusions of [38].
- However, PCA does better than SL1. When PCA is used for initializing our supervised formulation, results can be slightly improved (case SL2). This illustrates either the limits of the non-convex optimization procedure, or that PCA is particularly well adapted to this problem.
- Learned dictionaries (cases UL and SL) outperform classical DCT dictionaries.

6 Conclusion

We have presented in this paper a general formulation for learning sparse data representations tuned to specific tasks. Unlike classical approaches that learn a dictionary adapted to the reconstruction of the input data, our method learns features in a supervised way. We have shown that this approach is effective for solving classification and regression tasks in a large-scale setting, allowing the use of millions of training samples, and is able of exploiting successfully unlabeled data, when only a few labeled samples are available. Experiments on handwritten digits classification, non-linear inverse image mapping, digital art authentification, and compressed sensing have shown that our method leads to state-of-the-art results for several real problems. Future work will include adapting our method to various image processing problems such as image deblurring and image super-resolution, and other inverse problems.

TABLE 3
Compressed sensing experiment on small natural image patches. The mean squared error (MSE) measured on a test set is reported for each scenario with standard deviations, obtained by reproducing 5 times each experiment, randomizing the algorithm initializations and the sampling of the training images. Each patch is normalized to have unit ℓ_2 norm, and the mean squared reconstruction error is multiplied by 100 for readability purposes. Here, r is the number of rows of the matrix $\mathbf{Z}$. The different scenarios vary with the way $\mathbf{D}$ and $\mathbf{Z}$ are learned (fixed, unsupervised, supervised). See the main text for details.

$\mathbf{Z}$	RANDOM			SL1	PCA			SL2
$\mathbf{D}$	DCT	UL	SL	SL	DCT	UL	SL	SL
$r=5$	77.3 ± 4.0	76.9 ± 4.0	76.7 ± 4.0	54.1 ± 1.3	49.9 ± 0.0	47.6 ± 0.0	47.5 ± 0.1	47.3 ± 0.3
$r=10$	57.8 ± 1.5	56.5 ± 1.5	55.7 ± 1.4	36.5 ± 0.7	33.7 ± 0.0	32.3 ± 0.0	32.3 ± 0.1	31.9 ± 0.2
$r=20$	37.1 ± 1.2	35.4 ± 1.0	34.5 ± 0.9	21.4 ± 0.1	20.4 ± 0.0	19.7 ± 0.0	19.6 ± 0.1	19.4 ± 0.2
$r=40$	19.3 ± 0.8	18.5 ± 0.7	18.0 ± 0.6	10.0 ± 0.3	9.2 ± 0.0	9.1 ± 0.0	9.0 ± 0.0	9.0 ± 0.0

APPENDIX
PROOFS AND LEMMAS

Before giving the proof of Proposition 1, we present two general results on the elastic net formulation of [29].

Lemma 1 (Optimality conditions of the elastic net): The vector $\boldsymbol{\alpha}^\star$ is a solution of Eq. (4) if and only if for all j in $\{1,\dots,p\}$,

$$\begin{aligned} &\mathbf{d}_j^\top(\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star)-\lambda_2\boldsymbol{\alpha}^\star[j]=\lambda_1\operatorname{sign}(\boldsymbol{\alpha}^\star[j]) \quad \text{if} \quad \boldsymbol{\alpha}^\star[j]\neq 0,\\ &|\mathbf{d}_j^\top(\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star)-\lambda_2\boldsymbol{\alpha}^\star[j]|\leq\lambda_1 \quad \text{otherwise.}\end{aligned} \tag{20}$$

Denoting by $\Lambda \triangleq \{j \in \{1,\dots,p\} \text{ s.t. } \boldsymbol{\alpha}^\star[j]\neq 0\}$ the active set, we also have

$$\boldsymbol{\alpha}^\star_\Lambda = (\mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda+\lambda_2\mathbf{I})^{-1}(\mathbf{D}_\Lambda^\top\mathbf{x}-\lambda_1\mathbf{s}_\Lambda), \tag{21}$$

where $\mathbf{s}_\Lambda$ in $\{-1;+1\}^{|\Lambda|}$ carries the signs of $\boldsymbol{\alpha}^\star_\Lambda$.

Proof: Equation (20) can be obtained by considering subgradient optimality conditions as done in [57] for the case $\lambda_2=0$. These can be written as

$$0\in\{-\mathbf{D}^\top(\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star)+\lambda_2\boldsymbol{\alpha}^\star+\lambda_1\mathbf{p}:\mathbf{p}\in\partial\|\boldsymbol{\alpha}^\star\|_1\},$$

where $\partial\|\boldsymbol{\alpha}^\star\|_1$ denotes the subdifferential of the ℓ_1 norm evaluated at $\boldsymbol{\alpha}^\star$. A classical result (see [58], page 238) is that the subgradients $\mathbf{p}$ of this subdifferential are characterized by the fact that for all j in $\{1,\dots,p\}$, $\mathbf{p}[j]=\operatorname{sign}(\boldsymbol{\alpha}^\star[j])$ if $\boldsymbol{\alpha}^\star[j]\neq 0$, and $|\mathbf{p}[j]|\leq 1$ otherwise. This gives directly Eq. (20). The equalities in Eq. (20) define a linear system whose solution is Eq. (21). □

The next proposition exploits these optimality conditions to characterize the regularity of $\boldsymbol{\alpha}^\star$.

Proposition 3 (Regularity of the elastic net solution): Assume $\lambda_2>0$ and (**A**). Then,

1) The function $\boldsymbol{\alpha}^\star$ is uniformly Lipschitz on $\mathcal{X}\times\mathcal{D}$.
2) Let $\mathbf{D}$ be in $\mathcal{D}$, ε be a positive scalar and $\mathbf{s}$ be a vector in $\{-1,0,+1\}^p$, and define $K_\mathbf{s}(\mathbf{D},\varepsilon)\subseteq\mathcal{X}$ as the set of vectors $\mathbf{x}$ satisfying for all j in $\{1,\dots,p\}$,

$$\begin{cases} |\mathbf{d}_j^\top(\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star)-\lambda_2\boldsymbol{\alpha}^\star[j]|\leq\lambda_1-\varepsilon & \text{if} \quad \mathbf{s}[j]=0,\\ \mathbf{s}[j]\boldsymbol{\alpha}^\star[j]\geq\varepsilon & \text{if} \quad \mathbf{s}[j]\neq 0.\end{cases} \tag{22}$$

where $\boldsymbol{\alpha}^\star$ is shorthand for $\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$.
Then, there exists $\kappa>0$ independent of $\mathbf{s}$, $\mathbf{D}$ and ε so that for all $\mathbf{x}$ in $K_\mathbf{s}(\mathbf{D},\varepsilon)$, the function $\boldsymbol{\alpha}^\star$ is twice continuously differentiable on $B_{\kappa\varepsilon}(\mathbf{x})\times B_{\kappa\varepsilon}(\mathbf{D})$, where $B_{\kappa\varepsilon}(\mathbf{x})$ and $B_{\kappa\varepsilon}(\mathbf{D})$ denote the open balls of radius $\kappa\varepsilon$ respectively centered on $\mathbf{x}$ and $\mathbf{D}$.

Proof: The first point is proven in [19]. The proof uses the strong convexity induced by the elastic-net term, when $\lambda_2>0$, and the compactness of $\mathcal{X}$ from Assumption (**A**).

For the second point, we study the differentiability of $\boldsymbol{\alpha}^\star$ on sets that satisfy conditions which are more restrictive than the optimality conditions of Eq. (20). Concretely, let $\mathbf{D}$ be in $\mathcal{D}$, $\varepsilon>0$ and $\mathbf{s}$ be in $\{-1,0,+1\}^p$. The set $K_\mathbf{s}(\mathbf{D},\varepsilon)$ characterizes the vectors $\mathbf{x}$ so that $\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ has the same signs as $\mathbf{s}$ (and same set of zero coefficients), and $\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})$ satisfies the conditions of Eq. (20), but with two additional constraints: (i) The magnitude of the non-zero coefficients in $\boldsymbol{\alpha}^\star$ should be greater than ε. (ii) The inequalities in Eq. (20) should be strict with a margin ε. The reason for imposing these assumptions is to restrict ourselves to points $\mathbf{x}$ in $\mathcal{X}$ that have a stable active set—that is, the set of non-zero coefficients Λ of $\boldsymbol{\alpha}^\star$ should not change for small perturbations of $(\mathbf{x},\mathbf{D})$, when $\mathbf{x}$ is in $K_\mathbf{s}(\mathbf{D},\varepsilon)$.

Proving that there exists a constant $\kappa>0$ satisfying the second point is then easy (if a bit technical): Let us assume that $K_\mathbf{s}(\mathbf{D},\varepsilon)$ is not empty (the case when it is empty is trivial). Since $\boldsymbol{\alpha}^\star$ is uniformly Lipschitz with respect to $(\mathbf{x},\mathbf{D})$, so are the quantities $\mathbf{d}_j^\top(\mathbf{x}-\mathbf{D}\boldsymbol{\alpha}^\star)-\lambda_2\boldsymbol{\alpha}^\star[j]$ and $\mathbf{s}[j]\boldsymbol{\alpha}^\star[j]$, for all j in $\{1,\dots,p\}$. Thus, there exists $\kappa>0$ independent of $\mathbf{x}$ and $\mathbf{D}$ such that for all $(\mathbf{x}',\mathbf{D}')$ satisfying $\|\mathbf{x}-\mathbf{x}'\|_2\leq\kappa\varepsilon$ and $\|\mathbf{D}-\mathbf{D}'\|_F\leq\kappa\varepsilon$, we have for all j in $\{1,\dots,p\}$,

$$\begin{cases} |\mathbf{d}_j^{\top\prime}(\mathbf{x}'-\mathbf{D}'\boldsymbol{\alpha}^{\star\prime})-\lambda_2\boldsymbol{\alpha}^{\star\prime}[j]|\leq\lambda_1-\frac{\varepsilon}{2} & \text{if} \quad \mathbf{s}[j]=0,\\ \mathbf{s}[j]\boldsymbol{\alpha}^{\star\prime}[j]\geq\frac{\varepsilon}{2} & \text{if} \quad \mathbf{s}[j]\neq 0.\end{cases}$$

where $\boldsymbol{\alpha}^{\star\prime}$ is short-hand for $\boldsymbol{\alpha}^\star(\mathbf{x}',\mathbf{D}')$, and $\mathbf{x}'$ is therefore in $K_\mathbf{s}(\mathbf{D}',\varepsilon/2)$. It is then easy to show that the active set Λ of $\boldsymbol{\alpha}^\star$ and the signs of $\boldsymbol{\alpha}^\star$ are stable on $B_{\kappa\varepsilon}(\mathbf{x})\times B_{\kappa\varepsilon}(\mathbf{D})$, and that $\boldsymbol{\alpha}^\star_\Lambda$ is given by the closed

form of Eq. (21). $\boldsymbol{\alpha}^\star$ is therefore twice differentiable on $B_{\kappa\varepsilon}(\mathbf{x}) \times B_{\kappa\varepsilon}(\mathbf{D})$. □

With this proposition in hand, we can now present the proof of Proposition 1:

Proof: The differentiability of f with respect to $\mathbf{W}$ is easy using only the compactness of $\mathcal{Y}$ and $\mathcal{X}$ and the fact that ℓ_s is twice differentiable. We will therefore focus on showing that f is differentiable with respect to $\mathbf{D}$, which is more difficult since $\boldsymbol{\alpha}^\star$ is *not* differentiable everywhere.

Given a small perturbation $\mathbf{E}$ in $\mathbb{R}^{m\times p}$ of $\mathbf{D}$, we compute

$$f(\mathbf{D}+\mathbf{E},\mathbf{W}) - f(\mathbf{D},\mathbf{W}) = \mathbb{E}_{\mathbf{y},\mathbf{x}}\Big[\nabla_{\boldsymbol{\alpha}}\ell_s^\top\big(\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D}+\mathbf{E}) - \boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})\big)\Big] + O(\|\mathbf{E}\|_{\mathrm{F}}^2), \tag{23}$$

where $\nabla_{\boldsymbol{\alpha}}\ell_s$ is short for $\nabla_{\boldsymbol{\alpha}}\ell_s(\mathbf{y},\mathbf{W},\boldsymbol{\alpha}^\star)$, and the term $O(\|\mathbf{E}\|_{\mathrm{F}}^2)$ comes from the fact that $\boldsymbol{\alpha}^\star$ is uniformly Lipschitz and $\mathcal{X}\times\mathcal{D}$ is compact.

Let now choose $\mathbf{W}$ in $\mathcal{W}$ and $\mathbf{D}$ in $\mathcal{D}$. We have characterized in Lemma 3 the differentiability of $\boldsymbol{\alpha}^\star$ on some subsets of $\mathcal{X}\times\mathcal{D}$. We consider the set

$$K(\mathbf{D},\varepsilon) \triangleq \bigcup_{\mathbf{s}\in\{-1,0,1\}^p} K_{\mathbf{s}}(\mathbf{D},\varepsilon),$$

and denoting by $\mathbb{P}$ our probability measure, it is easy to show with a few calculations that $\mathbb{P}(\mathcal{X}\setminus K(\mathbf{D},\varepsilon)) = O(\varepsilon)$. Using the constant κ defined in Lemma 3, we obtain that $\mathbb{P}(\mathcal{X}\setminus K(\mathbf{D},\|\mathbf{E}\|_{\mathrm{F}}/\kappa)) = O(\|\mathbf{E}\|_{\mathrm{F}})$. Since $\nabla_{\boldsymbol{\alpha}}\ell_s(\mathbf{y},\mathbf{W},\boldsymbol{\alpha}^\star)^\top\big(\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D}+\mathbf{E}) - \boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})\big) = O(\|\mathbf{E}\|_{\mathrm{F}})$, the set $\mathcal{X}\setminus K(\mathbf{D},\|\mathbf{E}\|_{\mathrm{F}}/\kappa)$ can be neglected (in the formal sense) when integrating with respect to $\mathbf{x}$ in the expectation of Eq. (23), and it is possible to show that

$$f(\mathbf{D}+\mathbf{E},\mathbf{W}) - f(\mathbf{D},\mathbf{W}) = \mathrm{Tr}\left(\mathbf{E}^\top g(\mathbf{D},\mathbf{W})\right) + O(\|\mathbf{E}\|_F^2),$$

where g has the form given by Eq. (16). This shows that f is differentiable with respect to $\mathbf{D}$, and its gradient $\nabla_{\mathbf{D}} f$ is g. □

ACKNOWLEDGMENTS

This paper was supported in part by ANR under grant MGA ANR-07-BLAN-0311 and the European Research Council (SIERRA and VideoWorld Project). Julien Mairal is now supported by the NSF grant SES-0835531 and NSF award CCF-0939370. The authors would like to thank J.M. Hugues and D.J. Graham and D.N. Rockmore for providing us with the Bruegel dataset used in Section 5.4, and Y-Lan Boureau and Marc'Aurelio Ranzato for providing their experimental setting for the digit recognition task.

REFERENCES

[1] S. Mallat, *A Wavelet Tour of Signal Processing, Second Edition*. Academic Press, New York, September 1999.

[2] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 54, no. 12, pp. 3736–3745, December 2006.

[3] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, January 2008.

[4] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

[5] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng, "Shift-invariant sparse coding for audio classification," in *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

[6] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[7] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: transfer learning from unlabeled data," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2007.

[8] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[9] ——, "Supervised dictionary learning," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. MIT Press, 2009, vol. 21, pp. 1033–1040.

[10] D. M. Bradley and J. A. Bagnell, "Differentiable sparse coding," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., 2009, vol. 21, pp. 113–120.

[11] K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun, "Learning invariant features through topographic filter maps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[12] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[13] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1999.

[14] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[15] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[16] M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1395–1408, 2009.

[17] K. Engan, S. O. Aase, and J. H. Husoy, "Frame based signal compression using method of optimal directions (MOD)," in *Proceedings of the 1999 IEEE International Symposium on Circuits Systems*, vol. 4, 1999.

[18] M. Aharon, M. Elad, and A. M. Bruckstein, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, November 2006.

[19] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[20] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, January 2003.

[21] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Machine Learning*, vol. 73, no. 3, pp. 243–272, 2008.

[22] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[23] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," in *Advances in Neural Information Processing Systems*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. MIT Press, 2007, vol. 19, pp. 1137–1144.

[24] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[25] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F.-J. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar, Eds. MIT Press, 2006.

[26] H. Larochelle and Y. Bengio, "Classification using discriminative restricted boltzmann machines," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

[27] D. Blei and J. McAuliffe, "Supervised topic models," in *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. MIT Press, 2008, vol. 20, pp. 121–128.

[28] Y. LeCun, L. Bottou, G. Orr, and K. Muller, "Efficient backprop," in *Neural Networks: Tricks of the trade*, G. Orr and M. K., Eds. Springer, 1998.

[29] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society Series B*, vol. 67, no. 2, pp. 301–320, 2005.

[30] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society. Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[31] L. Bottou and O. Bousquet, "The trade-offs of large scale learning," in *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. MIT Press, 2008, vol. 20, pp. 161–168.

[32] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, C. U. Press, Ed., 2004.

[33] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, 2001, pp. 556–562.

[34] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint." *Comm. Pure Appl. Math*, vol. 57, pp. 1413–1457, 2004.

[35] E. Candes, "Compressive sampling," in *Proceedings of the International Congress of Mathematicians*, vol. 3, 2006.

[36] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[37] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[38] Y. Weiss, H. Chang, and W. Freeman, "Learning compressed sensing," in *Snowbird Learning Workshop, Allerton, CA*, 2007.

[39] M. W. Seeger, "Bayesian inference and optimal design for the sparse linear model," *Journal of Machine Learning Research*, vol. 9, pp. 759–813, 2008.

[40] H. J. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

[41] L. Bottou, "Online algorithms and stochastic approximations," in *Online Learning and Neural Networks*, D. Saad, Ed., 1998.

[42] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[43] N. Murata, "Statistical study on on-line learning," *On-line learning in neural networks*, pp. 63–92, 1999.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, November 1998.

[45] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*, D. Touretzky, Ed., vol. 2. Morgan Kaufman, 1990.

[46] B. Haasdonk and D. Keysers, "Tangent distance kernels for support vector machines," 2002.

[47] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial grey scale," in *Proceedings of the Society of Information Display*, vol. 17, 1976, pp. 75–77.

[48] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Inverse halftoning by pointwise shape-adaptive DCT regularized deconvolution," in *Proceedings of the International TICSP Workshop on Spectral Methods Multirate Signal Processing (SMMSP)*, 2006.

[49] A. Foi, V. Katkovnik, K. Egiazarian, and J. Astola, "Inverse halftoning based on the anisotropic lpa-ici deconvolution," in *Proceedings of Int. TICSP Workshop Spectral Meth. Multirate Signal Process.*, 2004.

[50] T. D. Kite, N. Damera-Venkata, B. L. Evans, and A. C. Bovik, "A fast, high-quality inverse halftoning algorithm for error diffused halftones," *IEEE Transactions on Image Processing*, vol. 9, no. 9, pp. 1583–1592, 2000.

[51] R. Neelamani, R. Nowak, and R. Baraniuk, "WInHD: Wavelet-based inverse halftoning via deconvolution," *Rejecta Mathematica*, vol. 1, no. 1, pp. 84–103, 2009.

[52] S. Lyu, D. N. Rockmore, and H. Farid, "A digital technique for art authentication," *Proceedings of the National Academy of Science, USA*, vol. 101, no. 49, pp. 17 006–17 010, 2004.

[53] C. R. Johnson, E. H. Hendriks, I. J. Berezhnoy, E. Brevdo, S. M. Hugues, I. Daubechies, J. Li, E. Postma, and J. Z. Want, "Image processing for artist identification," *IEEE Signal Processing Magazine*, no. 37, July 2008.

[54] J. M. Hugues, D. J. Graham, and D. N. Rockmore, "Quantification of artistic style through sparse coding analysis in the drawings of Pieter Bruegel the Elder," *Proceedings of the National Academy of Science, TODO USA*, vol. 107, no. 4, pp. 1279–1283, 2009.

[55] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

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

[57] J. J. Fuchs, "Recovery of exact sparse representations in the presence of bounded noise," *IEEE Transactions on Information Theory*, vol. 51, no. 10, pp. 3601–3608, 2005.

[58] J. M. Borwein and A. S. Lewis, *Convex analysis and nonlinear optimization: Theory and examples*. Springer, 2006.

Julien Mairal received the graduate degree from Ecole Polytechnique, Palaiseau, France in 2005 and the PhD degree from the Ecole Normale Supérieure, Cachan, France in 2010 under the supervision of Jean Ponce and Francis Bach in the INRIA Willow project-team in Paris. He recently joined the department of statistics of the University of California, Berkeley as a post-doctoral researcher. His research interests include machine learning, computer vision and image/signal processing.

Francis Bach is the leading researcher of the Sierra INRIA project-team, in the Computer Science Department of the Ecole Normale Supérieure, Paris, France. He graduated from the Ecole Polytechnique, Palaiseau, France, in 1997, and earned his PhD in 2005 from the Computer Science division at the University of California, Berkeley. His research interests include machine learning, statistics, optimization, graphical models, kernel methods, and statistical signal processing.

Jean Ponce is a computer science professor at Ecole Normale Supérieure (ENS) in Paris, France, where he heads the ENS/INRIA/CNRS Project-team WILLOW. Before joining ENS, he spent most of his career in the US, with positions at MIT, Stanford, and the University of Illinois at Urbana-Champaign, where he was a full professor until 2005. Jean Ponce is the author of over 120 technical publications in computer vision and robotics, including the textbook "Computer Vision: A Modern Approach". He is an IEEE Fellow, served as editor-in-chief for the International Journal of Computer Vision from 2003 to 2008, and chaired the IEEE Conference on Computer Vision and Pattern Recognition in 1997 and 2000, and the European Conference on Computer Vision in 2008.