DOI: 10.1109/tmm.2017.2778568

Non-Negative OPLS for Supervised Design of Filter Banks: Application to Image and Audio Feature Extraction

Sergio Muñoz-Romero, Jerónimo Arenas-García, *Senior Member, IEEE*, and Vanessa Gómez-Verdejo

***Abstract*—Audio or visual data analysis tasks usually have to deal with high-dimensional and non-negative signals. However, most data analysis methods suffer from overfitting and numerical problems when data have more than a few dimensions needing a dimensionality reduction preprocessing. Moreover, interpretability for audio or visual applications is a desired property, specially when energy or spectral signals are involved, and thus the non-negativity has to be kept by the solutions. Because of these two necessities, we propose different methods to reduce the dimensionality of data while the non-negativity and interpretability of the solution are assured. In particular, we propose a generalized methodology to design filter banks in a supervised way for applications which dealing with non-negative data, and we explore different ways of solving the proposed function loss consisting of a non-negative version of OPLS method. We analyze the discriminative power of the features obtained with the proposed methods for two different and widely studied applications: texture and music genre classification. Furthermore, we compare the filter banks achieved by our methods with other state-of-the-art methods for *ad hoc* feature extraction.**

***Index Terms*—Orthonormalized Partial Least Squares, non-negative solution, Gabor filters, filter design, texture classification, music genre classification.**

I. Introduction

In recent years, data analysis methods are increasingly being used in order to automate the extraction of relevant information from data collections. However, most data analysis algorithms suffer from overfitting and numerical problems when patterns have more than a few dimensions. In order to appropriately apply these techniques, a feature extraction pre-processing stage is required, allowing to remove collinearity among variables and to reduce data dimensionality. For this reason, feature extraction techniques, and in particular Multivariate Analysis (MVA) methods [1], [2], have been successfully applied in many different applications of machine learning, such as biomedical engineering [3], [4], remote sensing [5], [6], or chemometrics [7].

MVA aggregates a family of methods that build a new set of features by extracting highly correlated projections of data representations in input and target spaces. Well-known representatives of these methods are Principal Component Analysis (PCA) [8], Partial Least Squares (PLS) approaches [9], or Canonical Correlation Analysis (CCA) [10]. PCA creates a new data representation space by finding the directions of largest input variance, providing an optimal set of features in terms of mean-squared reconstruction error. Unlike other MVA methods, PCA works in an unsupervised manner, i.e., it only considers the input data and does not take into account any target data. The Partial Least Squares (PLS) approach refers to a family of techniques which, in its general form, project both input and target variables to a new space, generating a set of projected features known as latent variables. The criterion used to extract these latent variables varies within the approach, but the general objective is to maximize the covariance of the two projected expressions. In CCA the aim is to find linear projections of the input and target data to maximize correlation between the projected data.

In this paper, we focus on a fourth MVA method known as Orthonormalized PLS (OPLS) [11], also known as semipenalized CCA [7], multilinear regression (MLR) [12], or reduced-rank regression [13]. OPLS is known to be optimal in the mean square error sense for performing multilinear regression [14], [15]; thus, it is very competitive as a pre-processing step in classification and regression problems [5], [15], [16]. Several works have also tried to establish the connections between OPLS and other MVA and discriminative methods (see, e.g., [13], [17] which proved that OPLS and CCA are the same solution in a balanced classification task if the target matrix is binary coded, or [18] that proposes a generalized framework for component analysis).

In this paper, we consider extensions of OPLS that favor the interpretability of the extracted features. In particular, when energy or spectral signals are involved, positive constraints should be imposed on the projection vectors, so that the extracted features can be interpreted as the energy contained in an equalized frequency band, and the projection vectors themselves can be seen as a sort of filter bank. This interpretation is useful, e.g., in audio or image data applications, where data processing is usually carried out in frequency domain. Note that, in this paper, we pursue interpretable solutions, considering interpretability as the machine capability of summarizing the reason for its output using as few variables and input features as possible.

We can find in the recent machine learning literature other algorithms preserving the non-negativity of the solution. One

S. Muñoz-Romero was with the Department of Signal Theory and Communications, Universidad Rey Juan Carlos, 28943, Fuenlabrada, Spain, and with the Center for Computational Simulation, Universidad Politécnica de Madrid, Spain; e-mail: sergio.munoz@urjc.es. ph. +34 91 624 8759.

J. Arenas-García and V. Gómez-Verdejo are with the Department of Signal Theory and Communications, Universidad Carlos III de Madrid, 28911 Leganés, Spain; e-mails: {jarenas,vanessa}@tsc.uc3m.es.

This work has been partly supported by MINECO projects TEC2013-48439-C4-1-R, TEC2014-52289-R, TEC2016-75161-C2-1-R, TEC2016-75161-C2-2-R, TEC2016-81900-REDT, and PRICAM (S2013/ICE-2933).

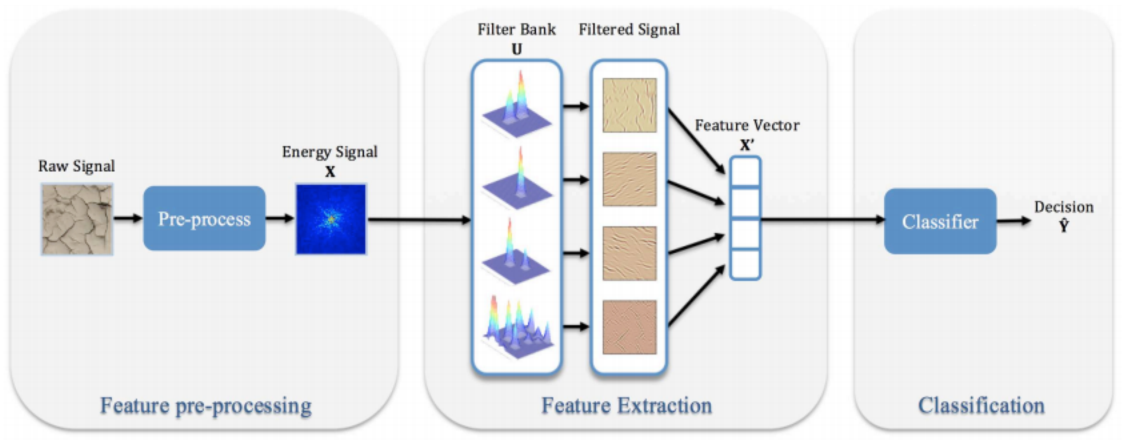

Fig. 1. Texture image recognition scheme from the raw image to the decision. The image is primarily processed to obtain a 2-D frequency representation, which is then passed through the filter bank so that each extracted feature summarizes the energy contained in a certain frequency range. Finally, classification is done based on the extracted features.

of the most popular algorithms is Non-Negative Matrix Factorization (NMF) [19] that has been applied for source separation [20], music transcription [21], or spectral data analysis [22], among others. Another perhaps less explored approach consists of incorporating a non-negativity constraint in the solution of the MVA methods. For instance, non-negative PCA has been used for blind positive source separation [23]; non-negative PLS (NPLS) for understanding Nuclear Magnetic Resonance (NMR) spectroscopy data [24]; non-negative CCA (NCCA) for audiovisual source separation [25]; and the positive constrained OPLS algorithm for music instrument and genre classification [26].

In contrast to NMF approaches, an additional advantage of incorporating non-negativity constraints in MVA methods is the capability to obtain sparse solutions and, indirectly, automatic feature selection. This preference for sparsity has motivated that during the last years many methods incorporate ℓ_0 and ℓ_1 regularizations in their formulations, see among others sparse versions of PCA [27], CCA [28], and OPLS [29]. However, unlike the methods we consider in this paper, neither ℓ_0 nor ℓ_1 regularization alone enforce non-negative solutions.

As we have already mentioned, in this paper we will focus on the design of banks of filters that provide interpretable features for supervised problems. Figure 1 illustrates the whole process when dealing with images, and consists of three well-differentiated blocks: 1) a pre-processing step which converts raw data into a better-fitted data representation in the frequency domain (see Sections III and IV); 2) a feature extraction step where the signal is passed through a bank of filters and, as a result, a feature vector $\boldsymbol{x}'$ is obtained, with each of its components being the energy of the image in a certain equalized frequency range; and 3) a classification stage where the feature vector $\boldsymbol{x}'$ is used to discriminate the class associated to the image.

In most previous works that rely on systems similar to the one depicted in Figure 1, it is the classifier block which is designed in a supervised manner, whereas the filter bank is typically built without any available label information. Instead, a sufficiently rich battery of general purpose filters is used (e.g. Gabor filters [30], [31]) or expert knowledge is exploited. Unlike these schemes, in this paper we will present a non-negative constrained OPLS method that exploits the available labels to design a bank of filters particularly fitted to each supervised task. As we will see, this results in more discriminative filters that can achieve better recognition rates with a smaller number of features. In order to show the versatility of our methods, we review two completely different scenarios: texture classification as a visual application; and music genre classification as an audio application.

Among a large amount of visual tasks, the classification of images by their textures is an interesting application which needs to incorporate a feature extraction stage. Surprisingly, the non-negative methods referred above have not been applied here, perhaps due to the wide and successful use of *ad hoc* feature extraction procedures. One of the most adopted techniques is Gabor Filtering (GB), which was proposed for texture discrimination in [30], [31] and is still used or even improved for efficient feature extraction [32], [33], and for rotation and scale-invariant texture classification [34]. Local Binary Pattern (LBP) is also a successful technique used for texture classification [35] but it does not provide any sort of interpretation to the solution.

Regarding audio-based music classification applications [36], and in particular the music information retrieval (MIR) field, music genre classification has been an active research area in the last years. Despite the great variety of different approaches to solve this problem, feature extraction is a usual stage into these solutions [37] and the use of sparse representations has been recently suggested as a way to boost performance [38]. However, sparse features do not provide any sort of interpretability to the solution, which is a desirable property for understanding music structure. In order to provide this capability, non-negative features can be extracted, as it was proposed by [26] and [39].

This paper proposes a general framework to obtain interpretable filters in classification tasks. For this purpose, starting from a non-negative OPLS formulation, we study four different approaches to solve the proposed optimization problem. The performance of these approaches is compared with classical filter design methods for image and music classification (where their parameters need to be adjusted by an expert user) and, additionally, with some current state of art approaches based on deep learning.

The rest of the paper is organized as follows: In the next section, we propose different ways of designing filter banks in a supervised way as a feature extraction stage for audio or visual applications. In Sections III and IV, we review the most frequently used methods in the visual and audio fields, focusing in particular on the texture and music genre classification problems. In order to prove the general applicability of our methods for applications when energy or spectral data are involved, we analyze the discriminative power of the extracted filter banks (i.e., non-negative features) on several texture and music genre databases in Section V. Finally, Section VI presents the main conclusions of our work.

II. OPLS-based Methods for Supervised Design of Filter Banks

In this section, we formulate several methods to design filter banks in a supervised learning scenario, where the goal is to learn relevant features from input data using a set of N training data $\{\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{y}}_i\}$, for $i = 1, \ldots, N$, where $\bar{\boldsymbol{x}}_i \in \Re^n$ and $\bar{\boldsymbol{y}}_i \in \Re^m$ are considered as the input and target data vectors, respectively. Therefore, n and m denote the dimensions of the input and target spaces. In classification problems, $\bar{\boldsymbol{y}}_i$ will be used to denote the class membership of the ith pattern, e.g., using 1-of-m encoding [40]. Furthermore, in this paper, we assume that all entries of $\bar{\boldsymbol{x}}_i$ are non-negative, which is the case when dealing with applications where the input space consists of spectral features. For notational convenience, we define the input and target data matrices: $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$ and $\mathbf{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N]$, where $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are centered versions of $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{y}}_i$, i.e.,

$$\boldsymbol{x}_i = \bar{\boldsymbol{x}}_i - \boldsymbol{\mu}_x, \qquad \boldsymbol{y}_i = \bar{\boldsymbol{y}}_i - \boldsymbol{\mu}_y,$$

where $\boldsymbol{\mu}_x$ and $\boldsymbol{\mu}_y$ are the means of the input and target data. Sample estimations of the input and target data covariance matrices, as well as of their cross-covariance matrix, can then be calculated as $\mathbf{C_{XX}} = \mathbf{XX}^\top$, $\mathbf{C_{YY}} = \mathbf{YY}^\top$ and $\mathbf{C_{XY}} = \mathbf{XY}^\top$, where we have neglected the scaling factor $\frac{1}{N}$, and superscript $^\top$ denotes vector or matrix transposition.

We denote the desired filter bank as $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{n_f}]$, where each of its columns can be seen as the response of one of the n_f filters in the bank and $\boldsymbol{x}'_i = \mathbf{U}^\top \boldsymbol{x}_i$ represent the extracted features corresponding to pattern $\boldsymbol{x}_i$.

When the input data are (positive) spectral features, we can consider $\mathbf{U}$ as a frequency filter bank whose coefficients are going to be forced to be non-negative, and $\boldsymbol{x}'_i$ can be interpreted as the non-negative output of each of the filters in the bank. However, when matrix $\mathbf{X}$ is centered, $\mathbf{X}' = \mathbf{U}^\top\mathbf{X}$ can also be seen as projections of the centered input data, but its entries are no longer guaranteed to be non-negative. Nevertheless, data centering does not affect the design of the filter bank and is recommended for learning purposes if regression procedures are involved [41]. In fact, it is quite straightforward to illustrate the irrelevance of the centering operation with respect to the extracted features since

$$\mathbf{U}^\top \bar{\boldsymbol{x}}_i = \mathbf{U}^\top \boldsymbol{x}_i - \mathbf{U}^\top \boldsymbol{\mu}_x = \boldsymbol{x}'_i - \boldsymbol{\mu}'_x,$$

where $\boldsymbol{\mu}'_x$ is the mean of the filtered data. Therefore, the interpretation of the filter bank remains valid when working with centered data, and the optimization problems become numerically more stable.

We use OPLS as a starting point in order to design the filter bank. OPLS is optimal in the mean square error (MSE) sense [14], i.e. to find the projection vectors so that the projected data best approximate the target data minimizing $||\mathbf{Y} - \mathbf{WU}^\top\mathbf{X}||_F^2$, where subscript F refers to the Frobenius norm of a matrix and $\mathbf{W}$ is the regression matrix. For this reason, OPLS usually outperforms all other traditional Multivariate Analysis (MVA) algorithms in both regression and classification tasks [2]. However, to enforce non-negative filter coefficients and allow more interpretable solutions, we add a non-negativity constraint to the OPLS objective function. Hence, the minimization problem that we propose for the design of the filter bank is:

$$\begin{array}{ll} \min\limits_{\mathbf{U},\mathbf{W}} & ||\mathbf{Y} - \mathbf{WU}^\top\mathbf{X}||_F^2 \\ \text{s.t.:} & \mathbf{U} \geq \mathbf{0} \end{array} \tag{1}$$

where $\mathbf{U} \geq \mathbf{0}$ indicates that all elements of matrix $\mathbf{U}$ have to be non-negative.

In this paper we propose four different algorithms to solve (1):

1) Non-negative OPLS (NOPLS): based on alternating two coupled convex problems (i.e., solving for $\mathbf{U}$ and $\mathbf{W}$ iteratively) that we proposed in [42] and [43]. Here, the method is adapted to include non-negative constraints.
2) NOPLS with Procrustes approach (P-NOPLS): similar to NOPLS, but solving the iterations for $\mathbf{W}$ using the "Orthogonal Procrustes problem" [44]. Here, we adapt this technique as another alternative to solve (1).
3) Deflated NOPLS: the sequential implementation of NOPLS using deflation.
4) NMF-like OPLS (NMF-OPLS): that can be considered as a supervised version of the NMF problem.

Note that although all algorithms try to solve the same optimization problem, they will in general converge to different solutions. In the rest of this section we will introduce these algorithms, and compare their results in the experimental section. For completeness we will also consider a fifth method known as positive constrained OPLS (POPLS). This method was proposed in [26] and relies on Quadratic Programming (QP) for solving (1).

A. Non-negative OPLS (NOPLS)

The algorithm proposed in this subsection is based on the sparse OPLS (SOPLS) method proposed in [29]. In that paper, an efficient solution of OPLS, also known as reduced rank regression problem in the statistics literature [13], was used in order to propose a feature extraction framework, which allows to easily include any constraints to the projection[1] matrix $\mathbf{U}$. The algorithm consists of the iterative application of two coupled steps: a constrained least squares regression problem, and a standard eigenvalue problem with the same complexity as the target dimensionality. Note that in audio or visual applications, the target dimensionality is usually much smaller than the input dimensionality (i.e., $m \ll n$) and, thus, this latter step is normally quite efficient.

In this subsection, we propose to replace the sparsity constraint that led to SOPLS by the desired non-negativity constraint. Therefore, according the same arguments of [29], we propose the following iterative procedure to solve the objective function (1):

1) $\mathbf{W}-$step: For fixed $\mathbf{U}$, minimize (1) with respect to $\mathbf{W}$, subject to $\mathbf{W}^\top\mathbf{W} = \mathbf{I}$.
   The solution of this problem is given by the eigenvalue decomposition
   $$\mathbf{C}_{\mathbf{X}'\mathbf{Y}}^\top \mathbf{C}_{\mathbf{X}'\mathbf{Y}} \mathbf{W} = \mathbf{W}\mathbf{\Lambda}, \tag{2}$$
   where $\mathbf{C}_{\mathbf{X}'\mathbf{Y}} = \mathbf{U}^\top \mathbf{C}_{\mathbf{XY}}$. Note that the dimension of the matrix that needs to be analyzed is m, which makes up a costly efficient step in the common case of $m < n$.
2) $\mathbf{U}-$step: For fixed $\mathbf{W}$, minimize (1) with respect to $\mathbf{U}$ only.
   We refer the reader to [45] and [46] for good summaries on optimization methods that can be used to solve this non-negative least squares (NNLS) problem. In the experiments section, we will use the MATLAB implementation of block principal pivoting algorithm provided by [46][2]. In case of adding also an ℓ_1-regularization term to (1) (i.e., $\lambda||\mathbf{U}||_1$), the Monotone Incremental Forward Stagewise Regression (MIFSR) algorithm proposed in [47] with the modifications introduced in [25] can be used.

The NOPLS method consists of the iterative application of the $\mathbf{W}$ and $\mathbf{U}-$steps until some convergence criterion is reached. Preliminary experiments have showed that initialization of the algorithm is not critical, and we simply initialize $\mathbf{U}$ for the first iteration as the identity matrix. As a stopping mechanism, we use $\text{Tr}\{\mathbf{\Lambda}^{(k)} - \mathbf{\Lambda}^{(k-1)}\} \leq \delta$, where the superscripts denote the iteration index and δ is a small constant. In plain words, the algorithm stops when the difference between the eigenvalues of the $\mathbf{W}-$step of two consecutive iterations is smaller than a prefixed constant δ.

B. NOPLS with Procrustes approach (P-NOPLS)

Our second proposed method consists of the modification of the $\mathbf{W}-$step of NOPLS applying the solution to the orthogonal Procrustes problem. This approach was used by [27] and [3] to derive sparse solutions of PCA and OPLS respectively. As we explained above, when fixing the projection matrix $\mathbf{U}$ the $\mathbf{W}-$step of the algorithm becomes:

$$\begin{array}{ll} \min\limits_{\mathbf{W}} & \|\mathbf{Y} - \mathbf{W}\mathbf{X}'\|_F^2 \\ \text{s.t.:} & \mathbf{W}^\top\mathbf{W} = \mathbf{I}. \end{array} \tag{3}$$

In [44], this problem is called an “Orthogonal Procrustes problem” and its solution is defined as

$$\mathbf{W}_{\text{PROCRUSTES}} = \mathbf{Q}\mathbf{P}^\top, \tag{4}$$

given the singular value decomposition $\mathbf{C}_{\mathbf{X}'\mathbf{Y}} = \mathbf{PDQ}^\top$, where $\mathbf{D}$ is a diagonal matrix containing the singular values, and $\mathbf{P}$ and $\mathbf{Q}$ contain the left and right singular vectors, respectively.

Since the solution of (2) gives $\mathbf{W}_{\text{NOPLS}} = \mathbf{Q}$, we can see that P-NOPLS results just in a rotated version of this matrix during the $\mathbf{W}-$step. However, we note that:

- The rotation process affects the relevance and ordering of the extracted features. For NOPLS we can affirm that the features/filter banks are sorted according to their relevance, i.e., first filter captures the maximum possible information with just one filter with respect to criterion (1), and so on. The rotation process prevents us from stating this same claim for P-NOPLS.
- In [29], we showed that the Procrustes solution is very sensitive to initialization and that for some initializations the algorithm may fail to converge.

The two arguments above justify our preference for NOPLS over the P-NOPLS solution. Nevertheless, P-NOPLS has also been included in the experiments for the sake of comparison and completeness.

C. Sequential implementation of NOPLS using deflation (defNOPLS)

In this section, we describe a sequential algorithm that implements the non-negative OPLS scheme introduced in Subsection II-A. This sequential algorithm consists of the following two steps: 1) Extraction of the projection vector $\boldsymbol{u}_j$ which represents the frequency response of the next filter to be included in the bank, and 2) application of a deflation procedure to eliminate the influence of the j^{th} eigenvector by annihilating the associated eigenvalue. These steps are repeated for $j = 1, \ldots, n_f$ until the desired number of filters or features is reached.

The design of the next filter consists of the extraction of a pair of vectors $\{\boldsymbol{u}_j, \boldsymbol{w}_j\}$ which are optimal with respect to (1). This can be done by iterating the $\mathbf{W}$ and $\mathbf{U}-$steps we have described for the NOPLS algorithm. Since in this case we are solving a unidimensional problem at each step, the solution to the $\mathbf{W}-$step simplifies to

$$\boldsymbol{w}_j = \frac{\mathbf{C}_{\mathrm{x}'\mathbf{Y}}^\top}{\|\mathbf{C}_{\mathrm{x}'\mathbf{Y}}\|_2}, \tag{5}$$

where $\mathbf{C}_{\mathrm{x}'\mathbf{Y}} = \boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}}$.

Throughout this paper, we will deflate the cross-covariance matrix by means of the Schur Complement deflation, which is one of the deflation methods proposed by [48], and can

[1]Note that $\mathbf{U}$ is not a projection operator in a rigorous mathematical sense, since it maps data from $\Re^n$ to $\Re^{n_f}$, and therefore does not satisfy the idempotent property of projection operators. However, the columns of $\mathbf{U}$ span the subspace of $\Re^n$ where the data are projected, and it is in this sense that we refer to $\mathbf{U}$ and $\boldsymbol{u}_i$ as projection matrix and vector respectively, and to $\mathbf{X}'$ as projected data. This nomenclature has been widely used in the machine learning field, particularly in works dealing with feature extraction methods.

[2]Code is available at http://www.cc.gatech.edu/~hpark/software/nmf bpas.zip

be applied to any of a number of constrained MVA methods or eigendecomposition-based problems, including this non-negative constrained OPLS solution. This deflation procedure avoids the reappearance of $\boldsymbol{u}_j$ as a component of future "pseudo-eigenvectors"[3]. Since, in our case, we have to deal with a supervised problem, the Schur Complement deflation can be written as

$$\mathbf{C}_{\mathbf{XY}} \leftarrow \mathbf{C}_{\mathbf{XY}} \left[\mathbf{I} - \frac{\mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j \boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}}}{\boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} \mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j} \right], \quad (6)$$

where $\leftarrow$ represents an update of the left matrix, and it renders u_j left orthogonal to $\mathbf{C}_{\mathbf{XY}}$ only, since after deflation it holds that

$$\boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} \left[\mathbf{I} - \frac{\mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j \boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}}}{\boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} \mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j} \right] = \boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} - \frac{\boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} \mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j \boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}}}{\boldsymbol{u}_j^\top \mathbf{C}_{\mathbf{XY}} \mathbf{C}_{\mathbf{XY}}^\top \boldsymbol{u}_j} = \mathbf{0}$$

Table I provides the pseudocode for the sequential algorithm that we have just described. Note that, in the table, subscript j is used to index the projection vectors (i.e., $j = 1, \ldots, n_f$), whereas superscript k indexes the iterative application of $\mathbf{W}-$ and $\mathbf{U}-$steps that are needed to converge to each projection vector. Different convergence criteria can be used at Step 2.2.3 of the algorithm. In the experimental section we will monitor the cosine distance

$$d_{\cos}\left(\boldsymbol{u}_j^{(k)}, \boldsymbol{u}_j^{(k-1)}\right) = \frac{\boldsymbol{u}_j^{(k)\top} \boldsymbol{u}_j^{(k-1)}}{\|\boldsymbol{u}_j^{(k)}\|_2 \|\boldsymbol{u}_j^{(k-1)}\|_2}, \quad (7)$$

and use as a stopping criterion $d_{\cos}\left(\boldsymbol{u}_j^{(k)}, \boldsymbol{u}_j^{(k-1)}\right) > 1 - \delta$, where δ is a tolerance parameter. Other possibilities would consist in monitoring the cosine distance between the regression coefficient vectors, or the eigenvalue of the $\mathbf{W}-$step.

TABLE I
PSEUDOCODE FOR THE SEQUENTIAL NOPLS USING DEFLATION.

```
1.- Inputs: centered matrices $\mathbf{X}$ and $\mathbf{Y}$, $n_f$.
2.- For $j = 1, \ldots, n_f$
    2.1.- Initialize $\boldsymbol{u}_j^{(1)} = \delta_j$ ‡.
    2.2.- For $k = 1, 2, \ldots$
        2.2.1.- Update $\boldsymbol{w}_j^{(k)}$ using (5).
        2.2.2.- Update $\boldsymbol{u}_j^{(k)}$ by solving the unidimensional version
                of the NNLS problem (1) subject to $\boldsymbol{u}_j^{(k)} \geq 0$.
        2.2.3.- If convergence criterion is reached, provide output
                values with $\{\boldsymbol{u}_j, \boldsymbol{w}_j\}$, otherwise back to 2.2.
    2.3.- Deflate cross-covariance matrix using (6).
3.- Output: $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{n_f}]$.
```

‡ δ_j is defined as a vector with its jth component equal to 1, and all other components equal to 0.

[3]In [48], the "pseudo-eigenvector" term is used to differentiate the true eigenvector obtained from the original objective function (i.e. without any constraint on the eigenvector) and the solution obtained when constraints are applied.

D. NMF-like OPLS (NMF-OPLS)

In this subsection, we solve problem (1) using an NMF approach, particularly we recur to the Multiplicative Updating (MU) rule proposed by [49], which is maybe the most popular NMF algorithm. Moreover, the loss function of the Projected-NMF algorithm proposed in [50] and some relationships among several versions [51] can be useful to see the similarities between NMF and the supervised version we propose here.

Unlike previous methods, the NMF methods require non-negative values both for $\mathbf{X}$ and $\mathbf{Y}$ (i.e., $\mathbf{X} \geq \mathbf{0}$ and $\mathbf{Y} \geq \mathbf{0}$) and, thus, the additional constraint $\mathbf{W} \geq \mathbf{0}$ needs to be considered. Since certain data will take negative values due to the centering operation, we will use the original data set $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$, which is non-negative.

The loss function to be minimized is also given by (1), although in this case the constraint $\mathbf{W} \geq \mathbf{0}$ will be added:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{W}} \quad & \|\bar{\mathbf{Y}} - \mathbf{W}\mathbf{U}^\top \bar{\mathbf{X}}\|_F^2 \\ \text{s.t.:} \quad & \mathbf{U} \geq \mathbf{0}, \ \mathbf{W} \geq \mathbf{0}. \end{aligned} \quad (8)$$

In order to ease the derivation of the current proposal, we rewrite the objective function of (1) in terms of the trace operator ($||\mathbf{A}||_F^2 = \mathrm{Tr}\{\mathbf{A}\mathbf{A}^\top\}$):

$$\begin{aligned} \mathcal{L}(\mathbf{W}, \mathbf{U}) = \mathrm{Tr}\{\mathbf{C}_{\bar{\mathbf{Y}}\bar{\mathbf{Y}}}\} - 2\,\mathrm{Tr}\{\mathbf{W}^\top \mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}^\top \mathbf{U}\} \\ + \mathrm{Tr}\{\mathbf{U}^\top \mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}} \mathbf{U} \mathbf{W}^\top \mathbf{W}\}. \end{aligned} \quad (9)$$

As a summary of the MU rule, let us suppose that the gradient of (9) with respect to $\mathbf{U}$ or $\mathbf{W}$ can be decomposed as

$$\partial\mathcal{L} = \partial\mathcal{L}^+ - \partial\mathcal{L}^-,$$

where $\partial\mathcal{L}^+ \geq 0$ and $\partial\mathcal{L}^- \geq 0$. Then, the element-wise updating rule follows as [51]:

$$\Psi \leftarrow \Psi \circ \frac{\partial\mathcal{L}^-}{\partial\mathcal{L}^+}, \quad (10)$$

where $\circ$ denotes the Hadamard (element-wise) product, $\frac{\mathbf{A}}{\mathbf{B}}$ represents the element-wise division, i.e., $\left[\frac{\mathbf{A}}{\mathbf{B}}\right]_{ij} = \frac{A_{ij}}{B_{ij}}$ (for the i^{th} row and the j^{th} column), and Ψ is the matrix that needs to be updated. Note that this update keeps the non-negativity of the solution Ψ at every step.

To apply the MU rule in our case, we obtain first derivatives of (9) with respect to $\mathbf{U}$

$$\frac{\partial\mathcal{L}(\mathbf{U}, \mathbf{W})}{\partial\mathbf{U}} = -2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}\mathbf{W} + 2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}\mathbf{W}^\top\mathbf{W},$$

that, considering that all involved matrices are non-negative, allows us to identify

$$\partial\mathcal{L}_{\mathbf{U}}^+ = 2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}\mathbf{W}^\top\mathbf{W}, \qquad \partial\mathcal{L}_{\mathbf{U}}^- = 2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}\mathbf{W}.$$

Similarly, first derivatives of (9) with respect to $\mathbf{W}$ are

$$\frac{\partial\mathcal{L}(\mathbf{U}, \mathbf{W})}{\partial\mathbf{W}} = -2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}^\top\mathbf{U} + 2\mathbf{W}\mathbf{U}^\top\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U},$$

so that we can identify

$$\partial\mathcal{L}_{\mathbf{W}}^+ = 2\mathbf{W}\mathbf{U}^\top\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}, \qquad \partial\mathcal{L}_{\mathbf{W}}^- = 2\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}^\top\mathbf{U}.$$

Therefore, following equation (10), the MU rules for updating $\mathbf{U}$ and $\mathbf{W}$, which constitute the core of the NMF-OPLS method, are given by

$$\mathbf{W} \leftarrow \mathbf{W} \circ \frac{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}^{\top}\mathbf{U}}{\mathbf{W}\mathbf{U}^{\top}\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}}, \qquad \mathbf{U} \leftarrow \mathbf{U} \circ \frac{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}\mathbf{W}}{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}\mathbf{W}^{\top}\mathbf{W}}.$$

As it is expected in NMF algorithms, preliminary experiments showed us that the algorithm initialization is critical. The nonnegative decomposition approximation method[4] (NNDSVD) proposed by [52] provides a good starting point for NMF algorithms, so we use it over the $\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}$ matrix, since we deal with a supervised scheme. Therefore, we initialize the $\mathbf{U}$ and $\mathbf{W}$ matrices as the left and right-decomposition matrices respectively by the NNDSVD approach: $\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}} \sim \mathbf{U}\mathbf{W}^{\top}$.

Non-negative constraints usually force a large number of zeros in the solution matrix, often causing numerical problems that make the MU update get stuck earlier than desired. In [53], it was proved that a slight improvement is reached substituting zeros by a small constant (e.g., $\epsilon = 10^{-16}$). Thus, the improved MU rules are given by

$$\mathbf{W} \leftarrow \max\left(\epsilon, \mathbf{W} \circ \frac{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}^{\top}\mathbf{U}}{\mathbf{W}\mathbf{U}^{\top}\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}}\right), \quad (11)$$

$$\mathbf{U} \leftarrow \max\left(\epsilon, \mathbf{U} \circ \frac{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{Y}}}\mathbf{W}}{\mathbf{C}_{\bar{\mathbf{X}}\bar{\mathbf{X}}}\mathbf{U}\mathbf{W}^{\top}\mathbf{W}}\right). \quad (12)$$

Furthermore, in NMF algorithms it is usual to include a normalization step in each MU update iteration in order to ease the convergence. In our case, this step is also applied and the $\mathbf{U}$ and $\mathbf{W}$ matrices are normalized with its Frobenius norm respectively.

Table II provides the pseudocode for the NMF-OPLS algorithm that we have just described. Different convergence criteria can be used at Step 2.2.4 of the algorithm. In the experimental section we use $||\mathbf{U}^{(k)} - \mathbf{U}^{(k-1)}||_F \leq \delta$ as a stopping mechanism, where the superscripts denote the iteration index and δ is a small constant. Then, the algorithm is stopped when the solutions achieved in two consecutive iterations differ less than a small threshold.

TABLE II
NMF-OPLS PSEUDOCODE.

```
1.- Inputs: positive matrices $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$.
    2.1.- Initialize $\mathbf{W}^{(1)}$ and $\mathbf{U}^{(1)}$ with NNDSVD algorithm.
    2.2.- For $k = 1, 2, \ldots$
        2.2.1.- Update $\mathbf{W}^{(k)}$ using (11).
        2.2.2.- Update $\mathbf{U}^{(k)}$ using (12).
        2.2.3.- Normalize $\mathbf{W}^{(k)}$ and $\mathbf{U}^{(k)}$.
        2.2.4.- If convergence criterion is reached, go to 3.
3.- Outputs: $\mathbf{U}$, $\mathbf{W}$.
```

A few considerations are in order with respect to the algorithm we have just described. First, the main advantage of the MU rule is its simplicity and ease of implementation; however, slow convergence is frequently observed [46]. Second, the application of NMF-OPLS requires that the number of filters in the bank (n_f) is selected a priori, and sequential implementations are suboptimal since the positive constraint prevents the deflation procedure from completely removing the contribution of the previous projections. Finally, unlike NOPLS, the NMF-based implementation does not guarantee the filters of the bank (i.e., the columns of $\mathbf{U}$) to be sorted according to their relevance.

[4]We have used the NNDSVDa version of the algorithm in [52], as well as the implementation provided by its authors.

E. Positive Constrained OPLS (POPLS)

For completeness, this subsection describes the algorithm proposed in [26] to solve (1). In this case, matrix $\mathbf{W}$ is not explicitly calculated since the trick here is to express $\mathbf{W}$ in terms of $\mathbf{U}$ and to introduce it into the functional in (1) to obtain an optimization problem involving $\mathbf{U}$ only.

The optimal regression matrix can be calculated minimizing (1) with respect to $\mathbf{W}$ only, being the solution $\mathbf{W} = \mathbf{C}_{\mathbf{XY}}^{\top}\mathbf{U}\left(\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XX}}\mathbf{U}\right)^{-1}$. Introducing this result into (1), we can rewrite the objective as a function of $\mathbf{U}$ only

$$\begin{aligned}\mathcal{L}(\mathbf{U}) &= \|\mathbf{Y} - \mathbf{C}_{\mathbf{XY}}^{\top}\mathbf{U}\left(\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XX}}\mathbf{U}\right)^{-1}\mathbf{U}^{\top}\mathbf{X}\|_F^2 \\ &= \mathrm{Tr}\{\mathbf{C}_{\mathbf{YY}}\} - \mathrm{Tr}\{\left(\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XX}}\mathbf{U}\right)^{-1}\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XY}}\mathbf{C}_{\mathbf{XY}}^{\top}\mathbf{U}\}.\end{aligned}$$

Thus, we arrive to the following optimization problem

$$\begin{array}{ll}\max_{\mathbf{U}} & \mathrm{Tr}\{\left(\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XX}}\mathbf{U}\right)^{-1}\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XY}}\mathbf{C}_{\mathbf{XY}}^{\top}\mathbf{U}\} \\ \text{s.t.:} & \mathbf{U} \geq 0,\ \mathbf{U}^{\top}\mathbf{U} = \mathbf{I},\end{array} \quad (13)$$

where the last constraint is added in order to obtain one of the infinite solutions of objective function in (13). Note that this constraint is different from that of the OPLS problem. However, this constraint was preferred in [26] since it can be directly incorporated into the hyperspherical representation of the projection vectors, where each $\boldsymbol{u}_j$ is represented by a radius r_j and $n-1$ angles $\theta_j^{(s)}$, $s = 1, \ldots, n-1$. This way, the optimization can be solved with respect to $\theta_j^{(s)}$ for $r_j = 1$, and constraints $0 \leq \theta_j^{(s)} \leq \frac{\pi}{2}$ guarantee the non-negativity of the solution. This approach was followed in [26] to solve the convergence problems of the *fmincon* matlab function with the original representation used in (13).

An inconvenience of this method is that the desired OPLS property $\mathbf{U}^{\top}\mathbf{C}_{\mathbf{XX}}\mathbf{U} = \mathbf{I}$ is not satisfied, implying that filters are not arranged according to their discriminative power. To correct this, in this paper we apply a sequential implementation using Schur Complement deflation. The resulting sequential POPLS algorithm is summarized in Table III.

TABLE III
PSEUDOCODE FOR POPLS WITH DEFLATION PROCEDURE.

```
1.- Inputs: centered matrices $\mathbf{X}$ and $\mathbf{Y}$.
2.- For $j = 1, \ldots, n_f$
    2.1.- Update $\boldsymbol{u}_j$ by solving the unidimensional version of (13), i.e.,
          $\max_{\boldsymbol{u}_j} \frac{\boldsymbol{u}_j^{\top}\mathbf{C}_{\mathbf{XY}}\mathbf{C}_{\mathbf{XY}}^{\top}\boldsymbol{u}_j}{\boldsymbol{u}_j^{\top}\mathbf{C}_{\mathbf{XX}}\boldsymbol{u}_j}$,
          subject to $\boldsymbol{u}_j \geq 0$ and $\|\boldsymbol{u}_j\|_2 = 1$.
    2.2.- Deflate cross-covariance matrix using (6).
3.- Outputs: $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{n_f}]$.
```

III. Texture Classification Application

To illustrate the advantages of OPLS methods for supervised filter design, we will consider texture recognition and music genre classification applications. For this reason, this section provides a brief summary of the necessary background on image processing required to understand the application of OPLS methods for texture recognition and the alternatives that are currently available in the literature. Similarly, Section IV will deal with the music genre recognition problem.

Figure 1 describes all the stages that are typically encountered in texture recognition. Following these stages, the image is firstly preprocessed and, then, it is transformed to frequency domain to facilitate the extraction of relevant features through filtering. Finally, a classifier is used to discriminate among the different textures.

A typical simple pre-process step applied in this field is to apply a two-dimensional Fast Fourier Transform to each image ($\boldsymbol{x}$, if it is vectorized) which is usually transformed into gray scale if the raw image is in color. This allows the next stage to extract features ($\boldsymbol{x}'$) in a frequency domain by means of a filter bank ($\mathbf{U}$) as

$$\boldsymbol{x}' = \mathbf{U}\boldsymbol{x}.$$

One of the most popular feature extraction techniques for texture classification is Gabor filtering. However, Gabor filters show a strong dependence on several parameters whose values may significantly affect the discriminative performances of the subsequent classifier. The effects of Gabor filter parameters on texture classification was comprehensively evaluated by [32].

One remarkable conclusion of [32] is that smoothing parameters γ and η are important parameters, whereas the number of frequencies and orientations has, in general, little effect on texture classification. This result contradicts the widely accepted belief that the parameters presenting a highest influence over the texture classification performance are related to the number of orientations (n_O), the number of frequencies (n_F), and the largest frequency of all filters.

As illustrated by the study in [32], the design of a GF bank can be very costly as a result of the validation process that is required to adjust the free parameters. Furthermore, the general shape of GFs is predefined a priori, and apart from their widespread use in texture classification, there are no guarantees that GFs are the most adequate selection for a particular task. In contrast to this, our proposed methods use available labels to build the bank of filters and do not assume any predefined shape for the frequency response of the filters. For this reason, we expect that they are able to extract more discriminative features for each particular supervised task.

To conclude the subsection, there are some practical considerations that need to be taken into account and imply differences between our schemes and the direct application of GF:

- Gabor filtering provides two features per filtered image: the mean (μ) and the standard deviation (σ) of the filtered image. In contrast, our methods produce only one feature per filtered image.
- In order to ease the operation of our algorithms, we decimate each frequency image by using the average energy over neighboring pixels of the image. This results in lower resolution of $\rho \times \rho$ and thus in a dimensionality reduction (of the vectorized frequency vector) to n variables (being $n = \rho^2$). This pre-processing step is illustrated in Figure 2. The first half of this scheme is also used as the GF pre-processing step.

IV. Music Genre Classification

In this section we review the applicability of the OPLS-based schemes presented in this paper for music recognition applications. Although we consider here the particular case of music genre classification, the approach could be straightforwardly extended to other music information retrieval tasks. As before, the goal of the automatic design of the filter bank is to obtain good recognition rates, while at the same time extracting interpretable features.

The whole music recognition application can be summarized into three well-differentiated blocks (see Figure 3): the audio pre-processing step, which transforms the raw data into actionable information for the subsequent step; a filter bank, which aims at reducing dimensionality of data and easing the working of the subsequent phase; and the classifier, which takes the final recognition decision.

The audio pre-processing step, which transforms raw audio signals into relevant information for the subsequent step, can be further subdivided into two stages (see Figure 4): short-time feature extraction, which consists of features extracted in periods ranging from 5-100 ms where music signals are typically stationary, see e.g. [54]; and temporal feature integration, which is the process of combining all the feature vectors of a time frame into a single feature vector in order to capture the relevant temporal information of the frame.

In the following, we detail these two stages:

1) **Short-time features**: Mel Frequency Cepstral Coefficients (MFCC) have been selected as the short-time feature representation because of its widespread use and great success in several fields of MIR (see, e.g., [39], [55]). The MFCCs are ranked in decreasing order of the richness of representation of the spectral envelope. Thus, the lower MFCCs contain information about the slow variations in the spectral envelope. In experiments, we use only the 6 first MFCCs (as proposed in [26]) and, in order to minimize aliasing in the MFCCs, we apply a frame-size of 30 ms and a hop-size of 7.5 ms.
2) **Temporal feature integration**: In order to capture the relevant temporal information in the frame, we first estimate the power spectrum of each MFCC using a periodogram as suggested in [39]. Then, we concatenate these six energy features into a new single feature vector. There exist many other temporal feature integration methods, see [56] for a good review on these techniques.

Once the raw data are converted into a non-negative representation (i.e. the periodograms of the MFCCs, $\mathbf{X}$), the following step relies on applying a filter bank, $\mathbf{U}$, in order to extract the desired non-negative features, $\mathbf{X}' = \mathbf{U}^\top\mathbf{X}$, which can be seen as the energy contained in certain frequency bands of each MFCC periodogram. Note that this filter bank $\mathbf{U}$ is

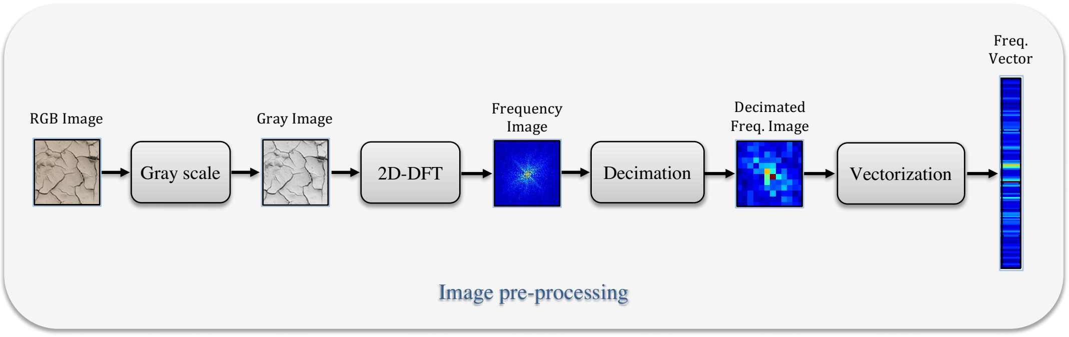

Fig. 2. Example of the pre-processing scheme applied to an image belonging to the earth class in the CGTextures dataset. The last two blocks are only included for our methods only.

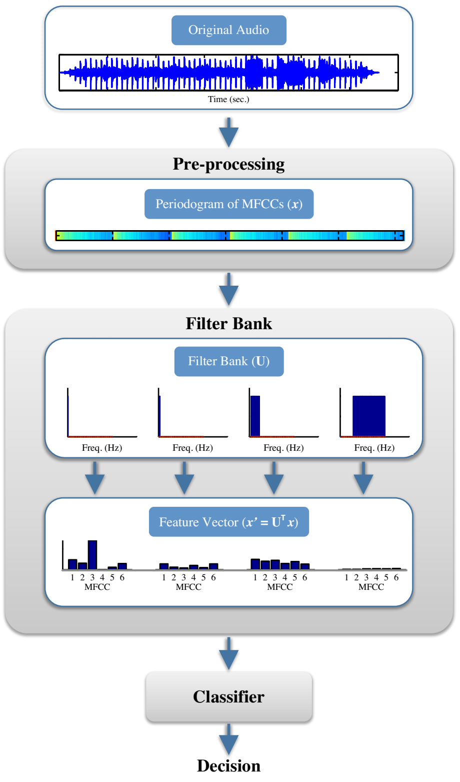

Fig. 3. Music genre classification scheme from the raw audio song to the decision. The audio clip is primarily processed to obtain a frequency representation, in this case a periodogram of the first 6 MFCCs. The periodograms are then passed through the filter bank, so that each extracted feature summarizes the energy contained in a certain frequency range. Finally, classification is carried out based on the extracted features.

inherently non-negative as well, since it is applied directly on the estimated power spectrum (periodogram), $\boldsymbol{x}_i' = \mathbf{U}^\top \boldsymbol{x}_i$, where $\boldsymbol{x}_i$ is the periodogram of the i^{th} MFCC coefficient, and $\boldsymbol{x}_i'$ is the corresponding i^{th} feature vector, which has as many components as the number of filters in the bank. These feature vectors will be introduced into the subsequent classifier.

In order to design the filter bank ($\mathbf{U}$), there are two different alternatives: using expert knowledge, which is the most typically used approach; and the supervised approaches

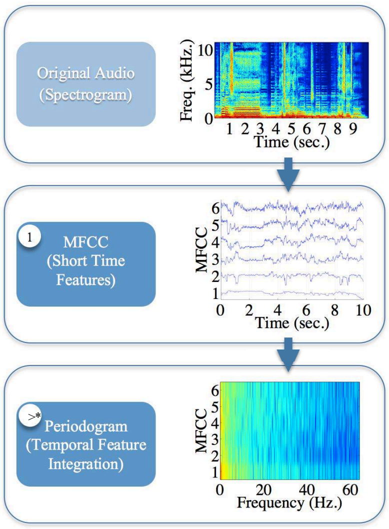
Original Audio (Spectrogram)
Freq. (kHz.)
Time (sec.)
1
MFCC (Short Time Features)
MFCC
Time (sec.)
Periodogram (Temporal Feature Integration)
MFCC
Frequency (Hz.)

Fig. 4. Pre-processing step scheme of a ten second excerpt of the song "Follow The Sun" by "Xavier Rudd".

that we propose in the paper, which make use of the label information allowing the *ad hoc* design of the filter banks for each recognition task. An example of the first alternative is the predefined Philips filter bank used in [39], where the authors suggest summarizing the power components in four frequency bands: 1) 0 Hz (DC component); 2) 0–2 Hz (beat rates); 3) 3–15 Hz (modulation energy, e.g., vibrato); 4) > 20 Hz (associated to the perceptual roughness). Therefore, for this particular filter bank, U is a matrix of size $D \times 4$, where $D = \frac{f_s}{2} + 1$ is the number of points of the periodogram and f_s is the length of the MFCC series used to calculate the periodogram (measured in number of samples). In this paper, we will use $f_s = 256$. In Subsection V-C, we compare our solutions with this fixed filter bank.

V. Experiments

In this section, we analyze the performance of the proposed set of supervised filter banks in two classification tasks: image texture recognition and music genre classification. In order to evaluate the advantage of our proposals, we analyze their

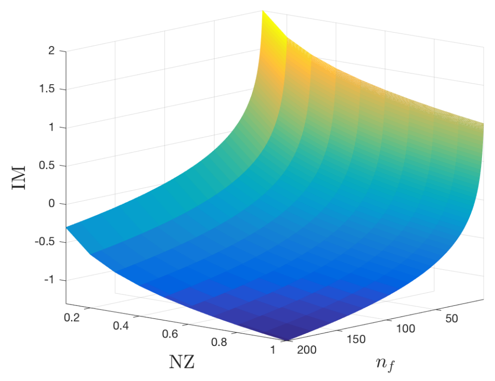

Fig. 5. IM curve in terms of n_f and NZ.

discriminative power and their interpretability capability in comparison to the well studied Gabor and Philips filter banks, which are designed *ad hoc* for the considered applications. Moreover, we compare the proposed methods against deep learning approaches, which are the state of the art in this problems, specially in visual applications as texture classification. Therefore, we can evaluate if the interpretable solutions obtained with our methods can be competitive in terms of performance.

In order to evaluate the interpretability of the solutions, we need to define some objective criterion. Focusing in these multimedia applications, we consider that the interpretability of the solutions is higher when 1) we use a small number of filters (n_f), and 2) the number of non-zero energy bands in each feature is small. We denote as NZ the rate of non-zero coefficients of the filters. However, give that these two interpretability criteria are of different nature and are subject to a tradeoff (a similar performance can be obtained with fewer but less sparse features), it is difficult to combine them into a single measure of interpretability. Moreover, the preference of either criterion will depend on the user or the application. Due to these reasons, in this paper we will inform these two terms separately, but we also propose and use a combined interpretability measure:

$$\text{IM} = -\log(\text{NZ}) - \log(n_f/n_{ref}), \tag{14}$$

where n_{ref} is a reference number of filters in order to compared with this reference. As an example, if an algorithm use n_{ref} filters, second part of (14) is equal to zero. Therefore, with NZ $= 1$ (i.e. all coefficients are non-zero), a number of filters smaller than n_{ref} obtains a positive IM (good interpretability), while using more filters than n_{ref} produces negative IM (poor interpretability). In this paper, we use the number of classes to be classified (m) as n_{ref} ($n_{ref} = m$). Note that IM increases linearly as NZ approaches one logarithmically. Figure 5 displays the IM curve in terms of n_f and NZ with $n_{ref} = 10$.

A. Experiment 1: Texture Classification

This subsection considers two different image texture classification tasks: classification based on a predefined set of categories, which is a more realistic scenario for texture classification, and the original image classification task, which is also a typical task used in the literature.

TABLE IV
DESCRIPTION OF THE MAIN CHARACTERISTICS OF THE IMAGE DATA SETS USED FOR TEXTURE CLASSIFICATION.

	No. images (train/test)	Size	No. classes
CGTextures	3840/1568	120×120	10
Brodatz [57]	1332/444	120×120	111

The first task considers a real scenario for texture classification, where each image belongs to a specific class of textures[5] among 10 different categories: bark, earth, gravel, plywood, snow, brick, grass, ivy, sky, and water. In order to provide more patterns to the database, each image is divided in a set of 16 sub-images. The second task considers the Brodatz dataset [57] which has been widely used in the texture classification literature. In this experiment, each image is also divided in a set of 16 sub-images and the goal of the classification task consists of assigning each sub-image to the original image. Table IV summarizes the main characteristics of these datasets. Figure 6 displays a 5 images per class excerpt of the CGTextures dataset, where each class consists of very different pictures, making this a difficult texture classification task.

For the following experiments, we have divided each image of side-size $L = 480$ pixels in 16 sub-images, and for our methods we have converted each sub-image into a frequency image of 12×12 pixels (i.e., $\rho = 12$) decimating the original frequency image by a factor of 10.

In case of Gabor Filtering, we also cross-validated (CV) the parameters η and γ, setting their values to $\eta = 0.5$ and $\gamma = 0.5$ for both datasets. The rest of parameters have been fixed according to [32]: $n_F = 4$, $n_O = 6$, and $F_r = \sqrt{2}$. Furthermore, the number of filters in the bank has been cross-validated for each method under study.

Thereupon, we will study the discriminative power and interpretability of the proposed supervised filter designs in comparison to the *ad hoc* well designed Gabor filter banks [32]. After designing each filter bank, we will train a linear Support Vector Machine (C-SVM) using the projected input data ($\mathbf{X}' = \mathbf{U}\mathbf{X}$) in order to evaluate the overall accuracy (OA) of every method; the optimal value of parameter C of the SVM has been cross-validated for each method under study. Since the aim of this paper is to obtain a subset of interpretable features useful for these classification tasks, we will focus on extracting the optimum quantity of energy for each one of the frequency bands making up the image. The interpretability capability will be analyzed by measuring the number of frequencies used by each filter bank and displaying the resulting projected data.

[5]Textures were downloaded from http://www.cgtextures.com/ in 2009 and the dataset created and used in this paper can be downloaded from http://www.tsc.uc3m.es/~smunoz/CGTextures.zip. Due to the origin of the textures, we will refer to this dataset as CGTextures.

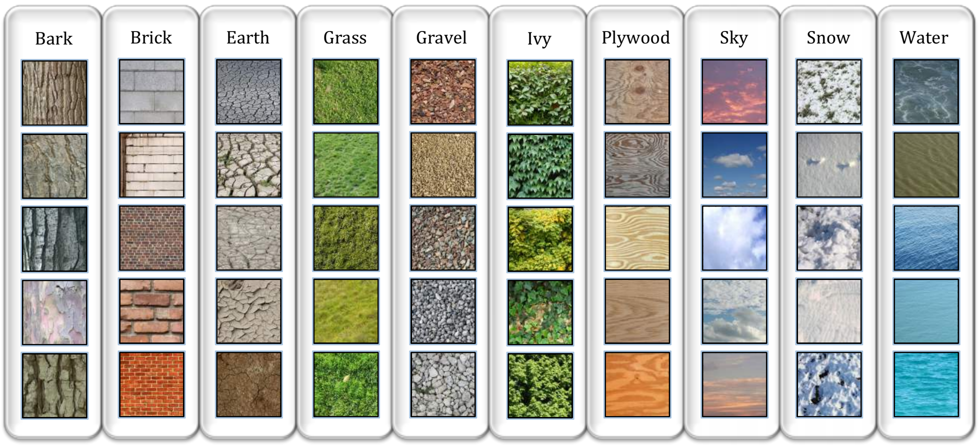

Fig. 6. A five images per class excerpt of the CGTextures dataset. In the preprocessing step, each of these images of size 480×480 pixels is divided in 16 subimages of size 120×120, which are the images used for the texture classification task.

TABLE V
PERFORMANCE COMPARISON AMONG OUR PROPOSALS AND SORTED GABOR FILTERS FOR THE CGTEXTURES DATASET.

Algorithm	OA(%)	n_f	$\#feat.$	NZ	IM
NOPLS	**79.91**	9	9	0.046	1.4
P-NOPLS	77.74	10	10	0.029	1.5
defNOPLS	77.81	9	9	0.041	1.4
NMF-OPLS	75.96	10	10	0.045	1.3
POPLS	74.49	10	10	0.031	1.5
OPLS	79.21	8	8	0.800	0.2
sorted GF	73.47	24	48	0.524	0.4

Texture Classification in the CGTextures dataset

Table V and Figure 7 compare the performances obtained by the proposed methods and by the Gabor Filter (GF) bank over the CGTextures dataset. In particular, Figure 7 displays the evolution of the overall accuracy (OA) with the number of filters in the bank, and Table V shows the OA of each method when n_f is selected using CV. To carry out a fair analysis, we have included results for the GF approach sorting out the filters according to MSE in the training set, i.e., for each n_f we selected the subset of filters achieving the best recognition capabilities.

As expected, the set of proposed supervised filter designs, and mainly the NOPLS algorithms, present an increased accuracy with respect to GF approaches; note that NOPLS outperforms the rest of the algorithms including OPLS. Moreover, the number of filters used by the supervised filter banks is less than half the number of filters selected for the GF bank. Furthermore, it is important to remark that, although all methods use a similar number of filters (n_f), the number of frequency bands selected by our methods is significantly smaller than by GF as it can be seen with the rate of non-zero coefficients of the filters (NZ) in Table V.

Besides, as we explained in Section III, our methods only extract one feature by each filter (see $\#feat.$ in Table V), while GF uses two features extracted by each filter: the mean of the filtered image (μ), and its standard deviation (σ). In order to compare the performance between the GF with one or two features by filter and our best method, we display a comparison of the evolution of the OA with the number of considered filters in Figure 7 (b).

In summary, we can state that the proposed methods are more discriminative, more selective, and more sparse than GF. In order to analyze the interpretability of each method under study in a qualitative way, Figure 8 shows the 10 first filters ($\boldsymbol{u}$) of the filter bank provided by each method, as well as an example of the filtered images ($\boldsymbol{x}_F = \boldsymbol{x} * \boldsymbol{u}$, being $*$ the convolution operation) from an image of the class *grass*. As we can see, the supervised filters are more acute and selective than those of the GF bank, being a mixture of band-pass filters in horizontal, vertical, and oblique directions. It is interesting to note the similarity among the filters in the banks of NOPLS, defNOPLS, and even the first filters of POPLS, which is indicative of NOPLS doing better than P-NOPLS. With respect to the GF bank, the worse accuracy of the classification system relying on these features indicate that this set of filters failed to extract discriminative features for the task at hand, making it clear the convenience of designing the filter bank in a supervised manner.

Texture Classification in the Brodatz dataset

In this subsection, we evaluate the different methods under study over the Brodatz texture classification scenario. In this case, each subimage has to be assigned to its original image correctly and, thus, the number of classes to be labeled is the same as the original number of images available in the database. As in the previous subsection, we compare our proposals with the GF bank, although in this case we use the GF design proposed in [32] where the GF bank is designed *ad hoc* for this particular task.

Following the same experimental procedure than in the previous experiment, Table VI and Figure 9 include a comparison of the different methods under study. Again, to get a more fair comparison, the filters in the GF bank have been sorted according to criterion (1) measured over the training dataset. To further discuss the differences between

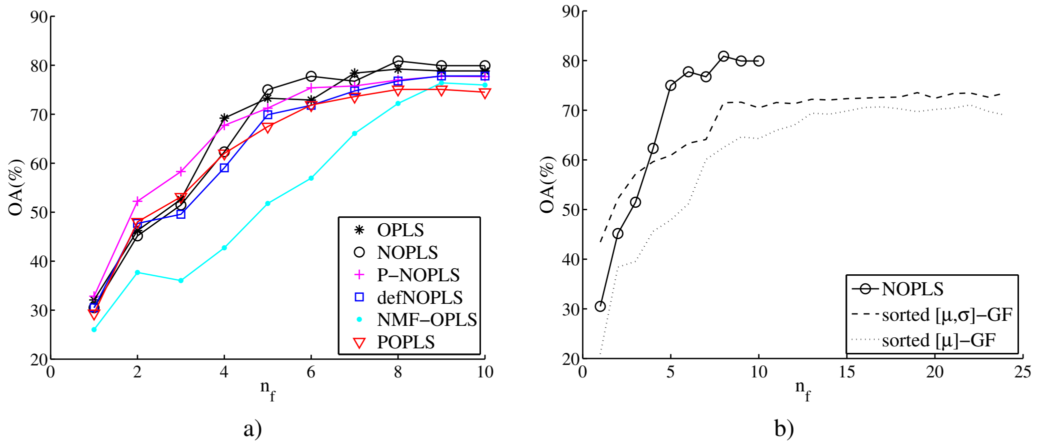

Fig. 7. Performance comparison among (a) our methods and (b) best-performing NOPLS method and Gabor Filter bank using mean and standard deviation ($[\mu, \sigma]$) or the mean only ($[\mu]$) of each filtered image.

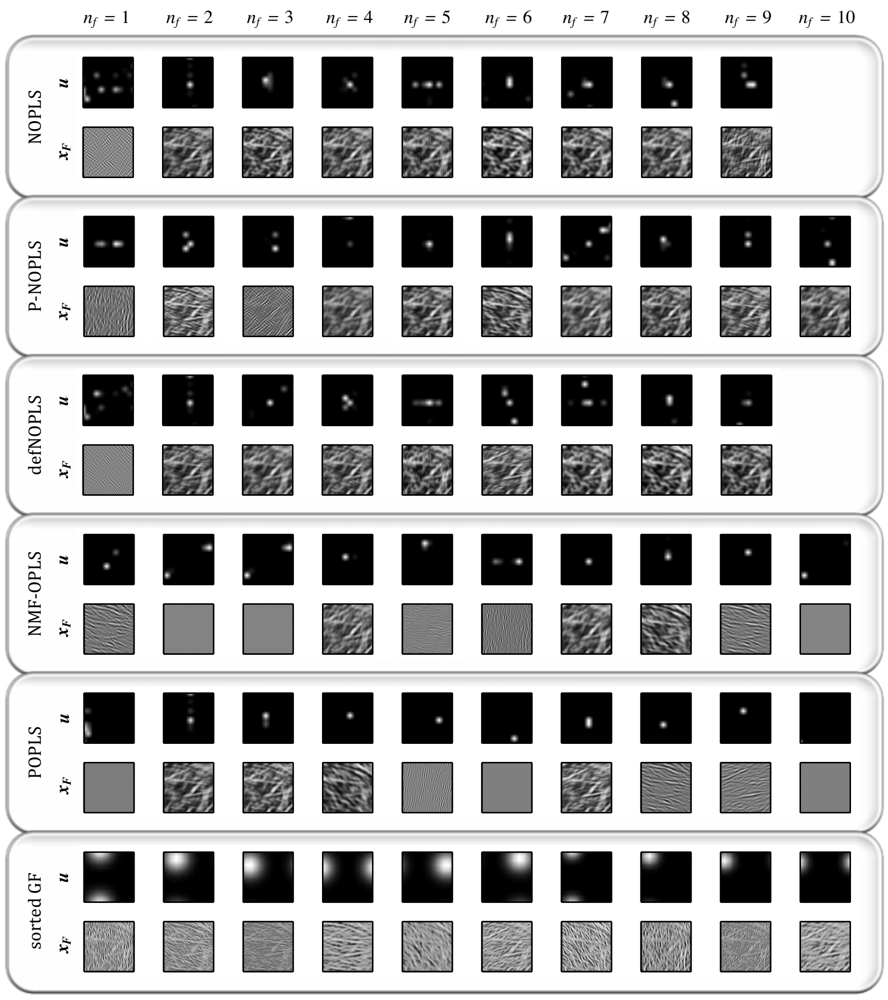

Fig. 8. Representation of the frequency response ($\boldsymbol{u}$) of the 10 first filters used by each method in the texture classification task. The corresponding filtered images ($\boldsymbol{x}_F$) for an example of class *grass* have also been represented for the different methods and filters.

the proposed methods, in this subsection we will analyze the training times required to build the filter banks. All simulations were run using Matlab 8 on a Macbook Pro with 8 GB RAM Memory and a 2.9 GHz Dual-core Intel Core i7 CPU.

TABLE VI
PERFORMANCE COMPARISON AMONG OUR PROPOSALS AND SORTED GFS.

Algorithm	OA(%)	n_f	$\#feat.$	NZ	IM	Time (sec.)
NOPLS	90.32	24	24	0.015	2.5	2
P-NOPLS	91.22	105	105	0.016	1.8	20
defNOPLS	**92.12**	53	53	0.0182	2.1	15
NMF-OPLS	90.99	63	63	0.009	2.3	20
POPLS	91.67	55	55	0.0059	2.5	44,000
OPLS	85.81	20	20	0.1802	1.5	<1
sorted GF	90.09	23	46	0.5202	1.0	-

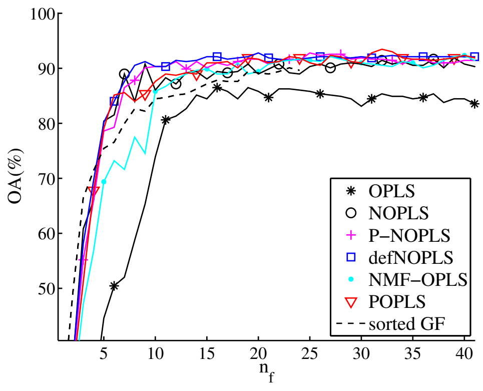

Fig. 9. Performance comparison among our proposals and GF for Brodatz dataset. These curves display the OA as a function of the number of filters used in the filter bank (n_f).

As we can see, all supervised methods are more discriminative than GF bank, even when there are few filters in the banks. Although P-NOPLS is slightly more discriminative than NOPLS, it takes much longer to train, and the number of filters is also larger. It is important to remark that NOPLS is the fastest algorithm (2.34 sec.) –excluding the OPLS algorithm since it does not include any constraint on the formulation– and requires half of the features than GF, whereas, in this case, the defNOPLS solution is the most discriminative and second fastest (14.84 sec.). P-NOPLS and NMF-OPLS need around 20 sec. and POPLS is dramatically slower with 12 h. and 12 min. Unlike previous problem, GF here uses less number of filters than supervised approaches, however the number of coefficients are similar (except for P-NOPLS) and the number of frequency bands of the images needed by our algorithms are drastically smaller than GF (see NZ and SR in Table VI). Comparing to OPLS results, we can see that the standard OPLS solution obtains the worst performance with any subset of filters. This fact points out as the non-negativity constraints not only provides interpretable solutions but also (in some cases) increased performance.

Note that, as we explained in Section II, P-NOPLS and NMF-OPLS do not sort the filters of the bank in terms of their importance. One of the consequences of this is that they require more filters than the other supervised methods; as an example, we see that P-NOPLS needs twice the number of filters than the rest of methods.

B. Experiment 2: Music Genre Classification

This second block of experiments aims at classifying music genre of a song from the periodogram of the 6 first MFCC coefficients extracted from each song. The dataset used here has been previously investigated in [26], [56], [58], and their results have revealed a great difficulty to successfully classify each song according to its musical genre (see [26], [58]). Moreover, the human evaluation study of [58] has found that the human definition of the genre for the audios in this dataset presents low consistency, resulting in a difficult dataset to apply genre classification task. Notwithstanding the above, it is interesting to study how supervised filter bank designs work in this setup.

The dataset consists of 1317 music snippets of 30s each, distributed evenly among the following 11 music genres: Alternative, Country, Easy Listening, Electronica, Jazz, Latin, Pop&Dance, Rap&Hiphop, R&B and Soul, Reggae and Rock. In case of Latin category, there are only 117 music samples. The music snippets are MP3 encoded music with a bitrate of 128 kbps or higher downsampled with a factor two to 22050 Hz. Note that this dataset has on average 1.83 songs per artist, which is other of the reasons making it so hard for genre classification.

For comparison purposes, we are going to consider the Philips Filter Bank proposed in [39] for a music genre classification task. As we explained at the end of Section IV, we will use periodograms of length D=129, so that the size of matrices U characterizing the Philips filter bank, as well as the supervised banks designed with any of our methods, will be $129 \times n_f$, with $n_f = 4$ for the Philips filter bank.

Due to the lack of a specific test subset, we apply a 10-fold cross-validation procedure in order to measure the classification accuracy of every method. In each fold, we design the optimal filters with nine partitions of the data, as described in Section IV, and subsequently we evaluate the performance on the remaining partition. Note that all samples of the same song cannot be divided into different partitions, i.e., partitions are defined in terms of songs instead of samples of the dataset.

A comparison among the supervised filter approaches and the Philips Filter Bank is displayed in Table VII. This table shows the OA (averaged over the 10 folds) when the 4 and 10 first filters in the bank are considered ($n_f = 4$ and $n_f = 10$, respectively). In the case of the Philips filter bank, results are only analyzed with 4 filters, since this is its maximum number of available filters. Table VII also includes the rate of non-zero coefficients of the filters (NZ) as well as the time required to design the different filter banks. To complete this analysis, Figure 10 displays the averaged OA as a function of the number of filters for all methods under study.

As we explained in Section II, two of our proposed methods (P-NOPLS and NMF-OPLS) lack the ability to sort the filters in the bank with respect to the importance of each filter. As a consequence of this shortcoming, when only a few filters are used the performance may be adversely affected, as is the case here, where these methods are even outperformed by the Philips filter bank when $n_f = 4$. Regarding the remaining

TABLE VII
OA (%) IN THE GENRE CLASSIFICATION TASK WHEN USING $n_f = 4$ AND $n_f = 10$ FILTERS. THE PERCENTAGE OF NON-ZERO COEFFICIENTS AND TRAINING TIME REQUIRED BY EACH METHOD ARE ALSO DISPLAYED.

Algorithm	OA ($n_f = 4$)	OA ($n_f = 10$)	NZ	IM ($n_f = 4$)	Time (sec.)
NOPLS	**35.69**	37.23	**0.029**	2.0	6.56
P-NOPLS	34.07	36.15	0.167	1.2	7.65
defNOPLS	35.23	36.77	0.039	1.8	15.40
NMF-OPLS	32.85	36.54	0.063	1.6	32.13
POPLS	34.85	37.31	0.138	1.3	2667.59
OPLS	30.08	**39.23**	1.000	0.4	**2.7**
Philips F. B.	34.15	-	0.038	1.9	-

supervised filters, it is not so clear which one presents the best performance: although POPLS has the best accuracy with $n_f = 10$, NOPLS obtains a similar performance, but with a lower percentage of non-zero coefficients. Even more, the accuracy of both defNOPLS and NOPLS are the best ones when few filters are used (see left subfigure of Figure 10), improving significantly the performance of the Philips filter bank. One of the advantages of defNOPLS with respect to NOPLS, is that the sequential implementation of the algorithm allows to automatically select n_f and stop the training process.

To conclude the section, Figure 11 shows the first 4 filters obtained on a single fold[6] for the first MFCC, so that we can analyze the information provided by each filter bank. Note that, similarly to the Philips Filter bank, NOPLS, defNOPLS, and POPLS pay attention to three well differentiated regions of the spectra, even though they are not in the same order: the lower modulation frequencies, which includes components at the beat-scale; the higher modulation frequencies, which are related to the perceptual roughness; and the modulation frequencies of instruments, which are the most important frequencies of the MFCCs periodograms. However, the supervised approaches are more flexible in the definition of the filters, and can adjust the cut-off frequencies and even shape the filter waveform to obtain the best possible performance in the genre classification task. The superior performance of the supervised techniques allows us to conclude the convenience of using the available target data not only for the training of the final classifier, but also in the design of the filters used in the feature extraction stage.

C. Experiment 3: Comparison with deep learning solutions

To complete this experimental analysis, in this section we compare the performance of the proposed methods with deep-learning approaches, since they are considered the state of the art in image recognition. With this purpose, we have selected these Deep Neural Networks (DNN):

- T-CNN-3 [59], which is a 3-layer convolutional neural networks (CNN), specifically designed for texture recognition problems,
- AlexNet [60], which is probably one the most popular state of the art DNN architectures,
- FV-CNN [61], that combines a CNN architecture with an internal representation based on the so-called Fisher vectors, and is the current state of the art solution in texture classification,

and we will compare their performance with our proposed methods in two texture classification databases: kth-tips-2b [7] [62] and DTD [8] [63], with 11 and 47 classes, respectively. In order to fairly compare all the algorithms, the same training and test data partitions have been used for all the methods and as well as the same experimental configuration as [59].

The results for all DNN methods have been directly taken from [59]. As we can see in Table VIII, our methods outperforms these state of the art techniques. Regarding the interpretability of the different solutions, we have reported the objective measure IM for our approaches. In spite of their good performance, the main criticism of DNNs is precisely that they can be seen as black boxes implementing a very large number of connections among computation nodes. Consequently, the number of filters is very large, which results in small IM values. To be more precise, and using the information available in [60], AlexNet is using $613,120$ for a final IM of approximately -4 or -5. For T-CNN-3 method, the number of units reported in [59] is $1,824$, and thus it obtains a final IM of approximately -2. IM of these DNN approaches is significantly smaller than for our methods. In summary, our methods show competitive performance in these problems, while providing more interpretable solutions.

For completeness, we should also mention that pretrained DNNs have also been used in [59,61], where the widely-used ImageNet database has been used as a pre-step to adjust the intermediate units of the networks. Note that these results cannot be directly compared to our methods, since for a fair comparison we should design specific procedures that allow our feature extractors to learn from additional databases such as ImageNet, which is out of the scope of this paper. Nevertheless, our methods still remain better than the results provided for DNN approaches in kth-tips-2b dataset, being 73.2%, 71.5% and 81.5% for T-CNN-3, AlexNet and FV-CNN, respectively. However, using ImageNet together with the DTD problem allows deep learning networks to very significantly improve their performance, achieving up to 75% for the current state of the art FV-CNN method, which is much better than any of the results in Table VIII for that database. We can conclude that probably DTD database does not provide much information for the identification of a good set of filters, and therefore using an external database to carry out this task is clearly beneficial.

Regarding interpretability, Figure 12 shows that just 10 sparse filters can be very useful for image processing experts to understand the filter design. In the case of DNN approaches, we have not considered showing their filters due to the high number of obtained filters, which are also not ordered by any criterion of relevance to the classification.

[6]We have checked that the differences between the filters obtained for each data fold are not very significant, so the presented conclusions can be easily generalized to the remaining folds.

[7]https://github.com/v-andrearczyk/caffe-TCNN
[8]https://www.robots.ox.ac.uk/~vgg/data/dtd/

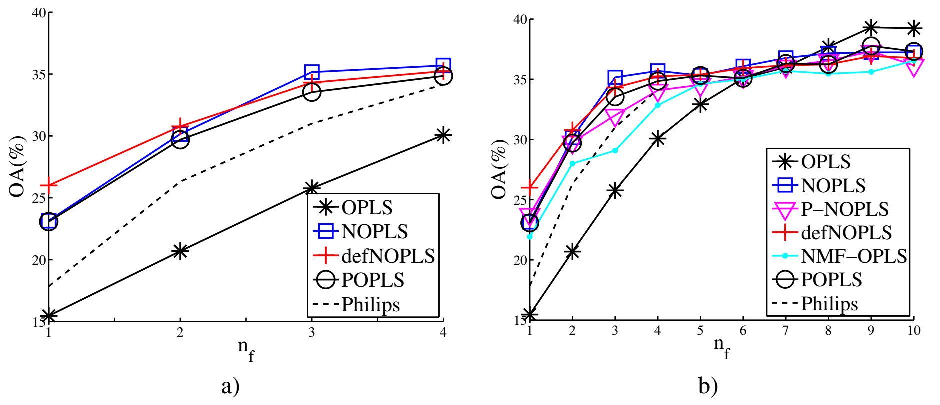

Fig. 10. Overall Accuracy (OA) comparison with (a) a detailed study among the best supervised filter banks and the Philips filter bank (only first 4 filters), and (b) a complete comparison among all methods with the full filter bank.

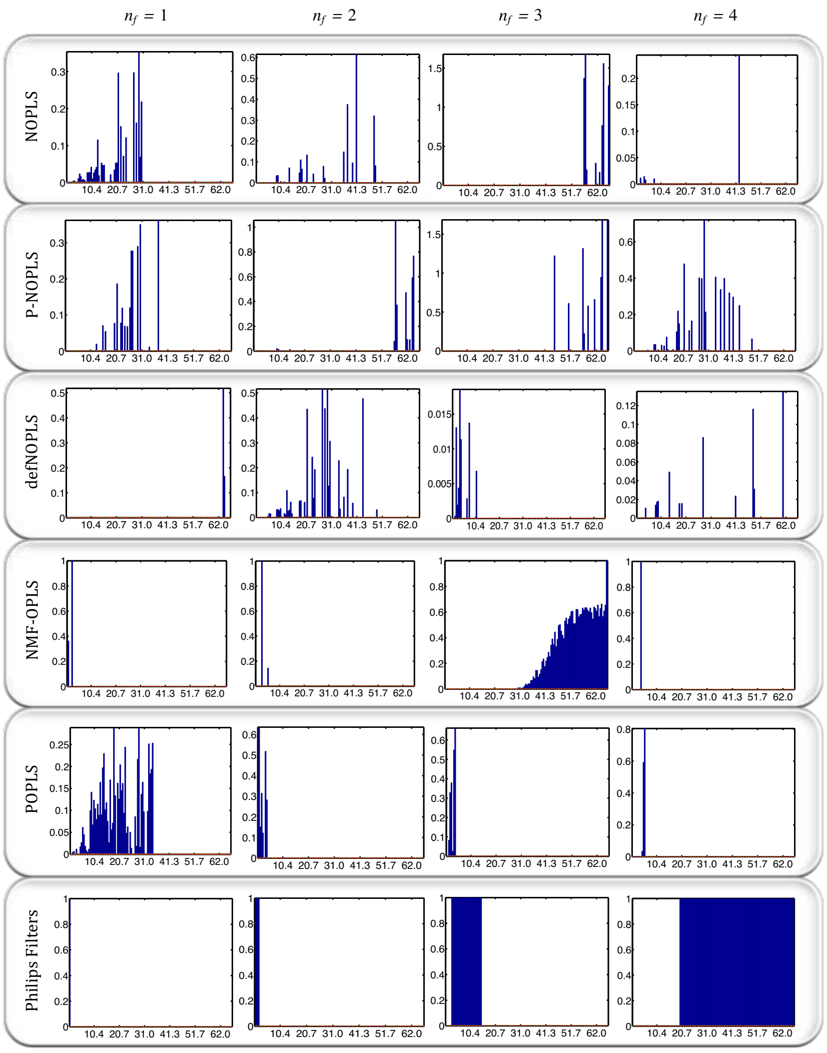

Fig. 11. Frequency response of the four first filters designed by each algorithm.

VI. Conclusions

In this paper, we have presented different methods for designing very versatile and interpretable filter banks for particular visual or audio classification tasks. All proposed methods are based on a supervised design with a common

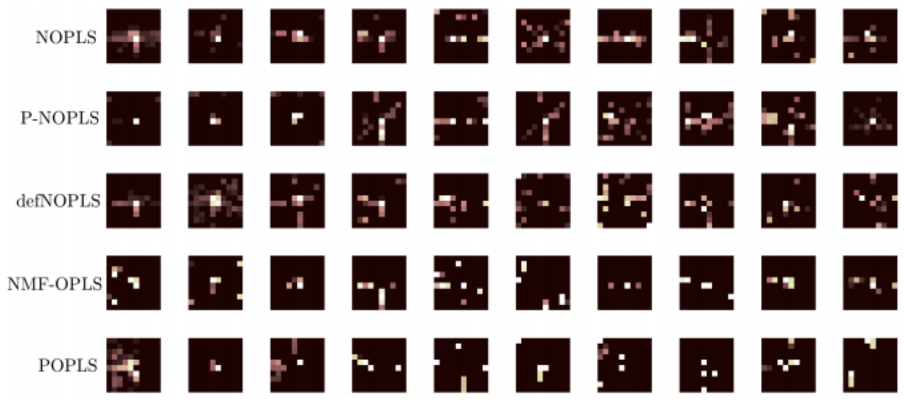

Fig. 12. Filter bank obtained as an average of the 4 filter banks from the 4 given splits by using NOPLS in kth-tips-2b dataset.

TABLE VIII
OA (%) AND IM COMPARISON WITH STATE OF THE ART DEEP LEARNING APPROACHES IN TEXTURE CLASSIFICATION TASKS.

Algorithm	kth-tips-2b		DTD	
	OA	IM	OA	IM
NOPLS	88.4	1.3	29.1	1.4
P-NOPLS	84.3	1.3	**31.8**	1.7
defNOPLS	**88.6**	1.3	31.1	1.5
NMF-OPLS	71.6	1.1	28.6	1.1
POPLS	79.1	1.5	31.3	1.9
T-CNN-3	48.7	-2.2	27.8	-1.6
AlexNet	47.6	-4.7	22.7	-4.1

objective function, and differ in the way they try to solve this non-convex problem. As an alternative to the POPLS algorithm proposed in [26], in this paper we propose several far less time-consuming methods which obtain similar or even better performance than POPLS. Moreover, our proposals outperform the *ad hoc* and well studied filter banks used in the state-of-art of visual and audio applications.

We have illustrated the versatility of our methods in our experiments section, where we have tackled two very different classification tasks: texture and music genre classification. The advantages of our approaches over other feature extraction methods are: 1) they provide elegant physical interpretations of the extracted features; 2) they are more discriminative with less number of filters; 3) they provide more interpretable and sparse solutions; 4) and they fit their filter banks to each particular task, unlike generic filter banks. Based on our findings, we can conclude that the block and deflated NOPLS algorithms seem to obtain the best results in terms of accuracy, sparsity and CPU requirements and, therefore, they should preferred over the other methods.

References

[1] K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate analysis*. Academic press, 1980.

[2] J. Arenas-García, K. Petersen, G. Camps-Valls, and L. K. Hansen, "Kernel multivariate analysis framework for supervised subspace learning: A tutorial on linear and kernel multivariate methods," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 16–29, 2013.

[3] M. A. J. van Gerven, Z. C. Chao, and T. Heskes, "On the decoding of intracranial data using sparse orthonormalized partial least squares," *Journal of Neural Engineering*, vol. 9, no. 2, pp. 26 017–26 027, 2012.

[4] L. K. Hansen, "Multivariate strategies in functional magnetic resonance imaging," *Brain and Language*, vol. 102, no. 2, pp. 186–191, 2007.

[5] J. Arenas-García and G. Camps-Valls, "Efficient kernel orthonormalized PLS for remote sensing applications," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, pp. 2872–2881, 2008.

[6] J. Arenas-García and K. B. Petersen, "Kernel multivariate analysis in remote sensing feature extraction," in *Kernel Methods for Remote Sensing Data Analysis*, G. Camps-Valls and L. Bruzzone, Eds. Wiley, 2009.

[7] M. Barker and W. Rayens, "Partial least squares for discrimination," *Journal of Chemometrics*, vol. 17, no. 3, pp. 166–173, 2003.

[8] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 6, pp. 559–572, 1901.

[9] H. Wold, "Non-linear estimation by iterative least squares procedures," in *Research Papers in Statistics*. Wiley, 1966, pp. 411–444.

[10] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, pp. 321–377, 1936.

[11] K. Worsley, J. Poline, K. Friston, and A. Evans., "Characterizing the response of pet and fMRI data using multivariate linear models (MLM)," *Neuroimage*, vol. 6, pp. 305–319, 1998.

[12] M. Borga, T. Landelius, and H. Knutsson, "A unified approach to PCA, PLS, MLR and CCA," Linköping University, SE-581 83 Linköping, Sweden, Report LiTH-ISY-R-1992, November 1997.

[13] G. C. Reinsel and R. P. Velu, *Multivariate reduced-rank regression: theory and applications*. Springer New York, 1998.

[14] S. Roweis and C. Brody, "Linear heteroencoders," Gatsby Computational Neuroscience Unit, Tech. Rep. 1999-002, 1999.

[15] J. Arenas-García, K. B. Petersen, and L. K. Hansen, "Sparse kernel orthonormalized PLS for feature extraction in large data sets," *Advances in Neural Information Processing Systems*, vol. 19, pp. 33–40, 2007.

[16] C. Dhanjal, S. R. Gunn, and J. Shawe-Taylor, "Efficient sparse kernel feature extraction based on partial least squares," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 31, no. 8, pp. 1347–1361, 2009.

[17] L. Sun, S. Ji, S. Yu, and J. Ye, "On the equivalence between canonical correlation analysis and orthonormalized partial least squares," in *Proc. 21st Intl. Joint Conf. on Artificial Intelligence (IJCAI-09)*, Pasadena, California, USA, July 2009, pp. 1230–1235.

[18] F. De la Torre, "A least-squares framework for component analysis," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 34, no. 6, pp. 1041–1055, 2012.

[19] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[20] D. Kounades-Bastian, L. Girin, X. Alameda-Pineda, S. Gannot, and R. Horaud, "A variational EM algorithm for the separation of time-varying convolutive audio mixtures," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 8, pp. 1408–1423, 2016.

[21] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. New Paltz, NY: IEEE, October 2003, pp. 177–180.

[22] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for spectral data analysis," *Linear algebra and its applications*, vol. 416, no. 1, pp. 29–47, 2006.

[23] E. Oja and M. Plumbley, "Blind separation of positive sources using non-negative PCA," in *Proc. 4th International Symposium on Independent Component Analysis and Blind Signal Separation*, Nara, Japan, April 2003.

[24] G. I. Allen, C. Peterson, M. Vannucci, and M. Maletić-Savatić, "Regularized partial least squares with an application to NMR spectroscopy," *Statistical Analysis and Data Mining*, vol. 6, no. 4, pp. 302–314, 2013.

[25] C. Sigg, B. Fischer, B. Ommer, V. Roth, and J. Buhmann, "Nonnegative CCA for audiovisual source separation," in *Proc. IEEE Intl. Workshop on Machine Learning for Signal Processing*, Thessaloniki, Greece, August 2007, pp. 253–258.

[26] J. Arenas-García, J. Larsen, L. K. Hansen, and A. Meng, "Optimal filtering of dynamics in short-time features for music organization," in *Proc. 7th Intl. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, October 2006, pp. 290–295.

[27] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of Computational and Graphical Statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[28] D. Hardoon and J. Shawe-Taylor, "Sparse canonical correlation analysis," *Machine Learning*, vol. 83, no. 3, pp. 331–353, 2011.

[29] S. Muñoz Romero, J. Arenas-García, and V. Gómez-Verdejo, "Iterative orthonormalized partial least squares with sparsity constraints," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. Vancouver, BC, Canada: IEEE, May 2013, pp. 3387–3391.

[30] M. R. Turner, "Texture discrimination by Gabor functions," *Biological Cybernetics*, vol. 55, no. 2-3, pp. 71–82, 1986.

[31] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, vol. 61, no. 2, pp. 103–113, 1989.

[32] F. Bianconi and A. Fernández, "Evaluation of the effects of Gabor filter parameters on texture classification," *Pattern Recognition*, vol. 40, no. 12, pp. 3325–3335, 2007.

[33] W. Li, K. Mao, H. Zhang, and T. Chai, "Designing compact Gabor filter banks for efficient texture feature extraction," in *Proc. 11th Intl. Conf. on Control Automation Robotics & Vision (ICARCV)*, Singapore, December 2010, pp. 1193–1197.

[34] J. Han and K.-K. Ma, "Rotation-invariant and scale-invariant Gabor features for texture image retrieval," *Image and Vision Computing*, vol. 25, no. 9, pp. 1474–1481, 2007.

[35] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 24, no. 7, pp. 971–987, 2002.

[36] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Trans. Multimedia*, vol. 13, no. 2, pp. 303–319, 2011.

[37] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content: a survey," *IEEE Signal Process. Mag.*, vol. 23, no. 2, pp. 133–141, 2006.

[38] B. L. Sturm, "On music genre classification via compressive sampling," in *Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME 2013)*, San Jose, USA, July 2013.

[39] M. F. McKinney and J. Breebaart, "Features for audio and music classification," in *Proc. Intl. Symposium on Music Information Retrieval (ISMIR)*, vol. 3, Baltimore, Maryland, USA, October 2003, pp. 151–158.

[40] C. Bishop, *Neural Networks for Pattern Recognition*. New York (NY): Oxford University Press, 1995.

[41] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[42] S. Muñoz Romero, J. Arenas-García, and V. Gómez-Verdejo, "Sparse and kernel opls feature extraction based on eigenvalue problem solving," *Pattern Recognition*, vol. 48, no. 5, pp. 1797 – 1811, 2015.

[43] S. Muñoz Romero, V. Gómez-Verdejo, and J. Arenas-García, "Regularized multivariate analysis framework for interpretable high-dimensional variable selection," *IEEE Computational Intelligence Magazine*, vol. 11, no. 4, pp. 24–35, Nov 2016.

[44] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966.

[45] M. H. Van Benthem and M. R. Keenan, "Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems," *Journal of chemometrics*, vol. 18, no. 10, pp. 441–450, 2004.

[46] J. Kim and H. Park, "Toward faster nonnegative matrix factorization: A new algorithm and comparisons," in *Proc. 8th IEEE Intl. Conf. on Data Mining (ICDM'08)*. Pisa, Italy: IEEE, December 2008, pp. 353–362.

[47] T. Hastie, J. Taylor, R. Tibshirani, and G. Walther, "Forward stagewise regression and the monotone lasso," *Electronic Journal of Statistics*, vol. 1, pp. 1–29, 2007.

[48] L. W. Mackey, "Deflation methods for sparse PCA," in *Advances in neural information processing systems*, vol. 21. The MIT Press, 2008, pp. 1017–1024.

[49] D. Seung and L. Lee, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556–562, 2001.

[50] Z. Yuan and E. Oja, "Projective nonnegative matrix factorization for image compression and feature extraction," in *Proc. 14th Scandinavian Conf. Image Analysis (SCIA 2005)*, Joensuu, Finland, June 2005, pp. 333–342.

[51] S. Choi, "Algorithms for orthogonal nonnegative matrix factorization," in *Proc. IEEE Intl. Joint Conf. on Neural Networks, IJCNN 2008*, Hong Kong, China, June 2008, pp. 1828–1832.

[52] C. Boutsidis and E. Gallopoulos, "SVD based initialization: A head start for nonnegative matrix factorization," *Journal of Pattern Recognition*, vol. 41, no. 4, pp. 1350–1362, 2008.

[53] N. Gillis and F. Glineur, "Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization," *Neural computation*, vol. 24, no. 4, pp. 1085–1105, 2012.

[54] J.-J. Aucouturier, F. Pachet, and M. Sandler, "The way it sounds": timbre models for analysis and retrieval of music signals," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1028–1035, December 2005.

[55] M. I. Mandel, G. E. Poliner, and D. P. Ellis, "Support vector machine active learning for music retrieval," *Multimedia systems*, vol. 12, no. 1, pp. 3–13, 2006.

[56] A. Meng, P. Ahrendt, J. Larsen, and L. K. Hansen, "Temporal feature integration for music genre classification," *IEEE Trans. Audio, Speech, and Lang. Process.*, vol. 15, no. 5, pp. 1654–1664, 2007.

[57] P. Brodatz, *Textures: a photographic album for artists and designers*. Dover New York, 1966, vol. 66.

[58] A. Meng and J. Shawe-Taylor, "An investigation of feature models for music genre classification using the support vector classifier," in *Proc. 6th Intl. Conf. on Music Information Retrieval (ISMIR)*, London, UK, September 2005, pp. 604–609.