Neurocomputing 74 (2011) 820–830

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Unsupervised feature extraction via kernel subspace techniques

A.R. Teixeira [a], A.M. Tomé [a,*], E.W. Lang [b]

[a] DETI/IEETA, Universidade de Aveiro, 3810-193 Aveiro, Portugal
[b] CIML/Biophysics, University of Regensburg, D-93040 Regensburg, Germany

ARTICLE INFO

Article history:
Received 20 February 2010
Received in revised form
12 October 2010
Accepted 16 November 2010
Communicated by S. Fiori
Available online 15 December 2010

Keywords:
Kernel PCA
Feature extraction and low-rank decompositions

ABSTRACT

This paper provides a new insight into unsupervised feature extraction techniques based on kernel subspace models. The data projected onto kernel subspace models are new data representations which might be better suited for classification. The kernel subspace models are always described exploiting the dual form for the basis vectors which requires that the training data must be available even during the test phase. By exploiting an incomplete Cholesky decomposition of the kernel matrix, a computationally less demanding implementation is proposed. Online benchmark data sets allow the evaluation of these feature extraction methods comparing the performance of two classifiers which both have as input either the raw data or the new representations.

1. Introduction

Unsupervised feature extraction methods try to generate representative features from raw data helping any classifier to learn a more robust solution and achieving a better generalization performance. Often original features are inappropriate, and even the number of features might be too large to conduct an efficient training of a classifier. Subspace techniques can be considered unsupervised feature generators which simultaneously provide dimension reduction and a more efficient data representation.

Principal component analysis (PCA) is a subspace technique which is widely used in many fields of research like face recognition [1] and related computer vision tasks [2]. It computes an eigenvalue decomposition of the data covariance matrix. The related eigenvectors form an orthogonal representation and identify directions of largest variance in the data. A new data representation is obtained by projecting the given data onto the new eigenvectors. These projections, called features, are thus uncorrelated and preserve as much energy of the raw data as corresponds to the sum over the ordered eigenvalues, i.e. the new data variances, depending on the dimension reduction envisaged. It is generally assumed that the smallest eigenvalues represent the variance of the noise in the data. Thus a representation of the data with reduced dimensionality ignores the eigenvectors with related smallest eigenvalues and enables dimension reduction and denoising simultaneously. The PCA model thus assumes that the original data can be represented as linear combinations of these features. However, this linear representation is a limitation if it is to model highly complex data.

Kernel PCA methods provide suitable non-linear extensions extracting non-linear principal components. KPCA is based on computing a standard linear PCA in a feature space, into which input data $\mathbf{x}$ is mapped via some nonlinear function $\phi(\mathbf{x})$. To this end a canonical dot product is considered in feature space and replaced by using a kernel function $k(\mathbf{x},\mathbf{y}) = \Phi^T(\mathbf{x})\Phi(\mathbf{y})$ which can be evaluated in input space. This so-called *kernel trick* can be carried out for any algorithm that can be formulated with dot products in feature space [3,4]. In a classification task, the new feature space representation provided by the non-linear kernel method most probably becomes linearly separable [5]. Several computer vision applications incorporate these techniques [6–8]. But KPCA can become quite cumbersome if not impossible with large data sets. It is well known that such techniques require $O(N^3)$ operations and $O(N^2)$ storage capacity, where N is the number of data points [9]. Also numerical instabilities arise if the kernel matrix is ill-conditioned. Such numerical instabilities can be avoided by resorting to multi-dimensional scaling techniques to reconstruct the feature vectors in interim space as proposed in [10].

Greedy KPCA offers an alternative in such cases. It builds upon the observation that in case of a full rank kernel matrix $\mathbf{K}$ one can approximate it by a low rank positive semi-definite matrix $\tilde{\mathbf{K}}$ via random sampling [11]. Further reduction of the computational complexity can be achieved by resorting to an incomplete Cholesky decomposition [12] combined with a symmetric pivoting scheme to guarantee numerical stability in case of smooth and gradually decaying eigenvalue spectra. It has been shown that a Cholesky factorization of the kernel matrix is numerically more stable and robust than approaches based on the Sherman–Morrision–Woodbury

* Corresponding author.
E-mail addresses: ateixeira@ua.pt (A.R. Teixeira), ana@ua.pt, ana@ieeta.pt (A.M. Tomé), elmar.lang@biologie.uni-regensburg.de (E.W. Lang).

0925-2312/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2010.11.011

formula [13] which often run into numerical instabilities as the inverse of a matrix needs to be updated. In fact it has been reported that a Cholesky factorization avoids numerical instabilities even when the initial data matrix is not well conditioned [14]. Furthermore, the quality of greedy approaches can be controlled by applying an appropriate stopping criterion. In [11], the trace of the approximation error of the kernel matrix $\mathrm{tr}(\mathbf{K}-\tilde{\mathbf{K}})$ was considered an upper bound and the algorithm was stopped when a certain tolerance ε was achieved. Similarly, the quality can be traced by iteratively optimizing the mapping error as discussed in [10]. The latter optimization showed excellent performance, however, it might occasionally suffer from multiple local optima. Alternatively, one can compute a low rank approximation to the kernel matrix and then invoke a Nyström extension of the eigenvectors to obtain a solution for the higher dimensional problem. The Nyström method is generally used for numerical solutions of eigenproblems [15]. The reduced-rank approximation of the kernel matrix can be obtained in either of the ways mentioned above and numerical instabilities due to ill-conditioning can be avoided by adding a jitter term $\mathbf{K}+\sigma\mathbf{I}$, $\sigma>0$ [16]. But the Nyström method can also be employed as an alternative to estimate a low rank approximation of the kernel matrix as has been discussed in [16]. It has been shown also that this Nyström approximation is equivalent to a variational calculus based on the Rayleigh principle [17].

In this work KPCA and greedy KPCA subspace models to perform feature extraction are discussed: all proposed models and the related formalism are consistently described in matrix notation. Basis vector matrices are represented in dual coordinates like it is often considered in support vector machines [18] to exploit the dot product properties of kernels. The dual representation arises from the fact that the new coordinates can be solely expressed by the training data set. The greedy KPCA is based on an incomplete Cholesky decomposition [19] of the kernel matrix using a symmetric pivoting scheme. Hence, the incomplete Cholesky decomposition of the kernel matrix, when formed with the training data set, is exploited to reduce the complexity of several kernel models [20]. This method allows to compute the Nyström approximation of the eigenvectors of the kernel matrix corresponding to the largest eigenvalues. The symmetric pivoting scheme of the algorithm is used to select a subset of the training data set to obtain the dual form of the greedy KPCA model. Another issue to be discussed is the impact that centering of the data has on the models. The proposal is to perform the centering and simultaneously maintain the new representation of the training data set uncorrelated. Finally, the extracted features are used for classification.

Numerical simulations compare the suitability of these feature extraction procedures via the performance of two simple classifiers: the nearest neighbor (NN) classifier and linear discriminant function (RL). As features either kernel features or principal component features were employed as well as a direct classification of the raw data. To further evaluate the impact of feature selection via non-linear projective subspace techniques, classification results are compared with the best results published in [21]. The greedy KPCA, furthermore, is used with a large benchmark data set (USPS data set of handwritten digits) which cannot be evaluated with standard KPCA using large training data sets.

2. Feature extraction and classification

Classification algorithms assume that the objects to be classified are represented by points in a multidimensional space. However before executing a training algorithm to estimate the parameters of the classifier, a transformation of the original vectorial data is advantageous extracting appropriate features which ease the classification task. The classification scheme, including subspace methods as a preprocessing step, is illustrated in Fig. 1. The system is composed of two parts: the subspace model and the classification. The present study will mainly focus on the feature generation step and discusses several appropriate subspace models. Any such subspace model is described by a matrix $\mathbf{U}$ whose columns form the basis vectors. The latter are used to generate the features to be used for classification by projecting the input data ($\mathbf{x}_n$) onto them. Hence, the projections ($\mathbf{z}_n$) constitute the new representation of the data. These projections thus results either as a simple linear combination of the input data, like with PCA projections, or they represent a superposition of non-linear components of the data, like with KPCA projections. In a subsequent classification task, the projections then form the input to any suitable classifier. While training the classifier, the basis vectors are estimated from a labeled training data set, and the projections are used to adapt the parameters of the classifier. Finally, the performance of the classifier can be evaluated employing projections of new data, taken from a test data set, onto those basis vectors computed from the training data set.

2.1. The subspace model for feature generation

The basis vector matrix $\mathbf{U}$ describes the subspace model, and the number of columns represents the dimension of the new representation. Within a PCA model, the largest dimension of the input space coincides with the dimension of the data vectors. The subspace model $\mathbf{U}$ is obtained from an eigenvalue decomposition of the covariance matrix $\mathbf{C}\propto\mathbf{U}\mathbf{D}\mathbf{U}^T$ of the data. In practice, the method is applied primarily to yield new representations with reduced dimensionality. Within a kernel PCA model, however, the nonlinear mapping function ϕ usually results in an increase of the dimension of the mapped data vectors. Fortunately, the mapping is never performed explicitly. Thus, kernel subspace techniques represent projective methods in a feature space created by a nonlinear transformation of the input data. The necessary steps will be summarized in the following employing a concise matrix notation and the representation of the subspace model in its dual form.

Consider that the mapped training data set is given by

$$\mathbf{\Phi}=[\phi(\mathbf{x}_1),\phi(\mathbf{x}_2)\ldots\phi(\mathbf{x}_N)]$$

Then the new representation of any training data point $\mathbf{x}_n=[x_{1n},x_{2n},\ldots,x_{Dn}]^T$ is obtained by computing the dot product between each mapped data vector $\phi(\mathbf{x}_n)$ and the basis vector matrix

$$\mathbf{z}_n=\mathbf{U}^T\phi(\mathbf{x}_n) \tag{1}$$

In order to avoid an explicit mapping of the data, the basis vector matrix $\mathbf{U}$ must be written in its dual form [3,22]:

$$\mathbf{U}=\mathbf{\Phi}\mathbf{V}\mathbf{D}^{-1/2} \tag{2}$$

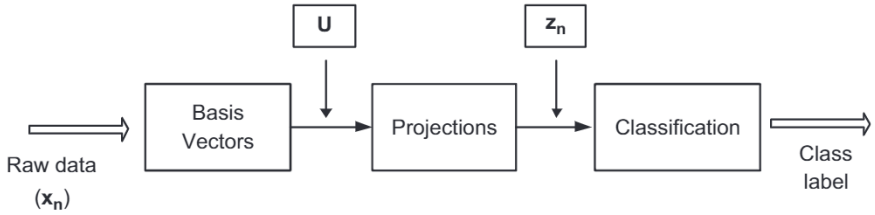

Fig. 1. Computing features ($\mathbf{z}_k$) using subspace techniques.

which expresses the fact that the eigenvectors can be written as a linear combination of the mapped training data vectors. Here $\mathbf{V}$ is a matrix of eigenvectors, and $\mathbf{D}$ is a diagonal matrix of corresponding eigenvalues, both resulting from the eigenvalue decomposition of the related kernel matrix $\mathbf{K}=\Phi^T\Phi$. Usually the columns of $\mathbf{V}$ are ordered in accord with the corresponding eigenvalues which decrease along the diagonal of matrix $\mathbf{D}$, i.e.

$$\lambda_1>\lambda_2>\cdots>\lambda_L>\cdots>\lambda_R>\cdots>\lambda_N$$

Therefore the basis vector matrix $\mathbf{U}$ has dimension $F\times L$, where F is the dimension of the feature space $\mathcal{F}$, determined by the non-linear mapping, and L is the number of selected eigenvectors (and corresponding eigenvalues) of the kernel matrix. Notice that if the kernel matrix is computed without mapping the data, its elements are given by $(k_{ij}=\mathbf{x}_i^T\mathbf{x}_j)$ and the number of nonzero eigenvalues is at most $\min(D,N)$, where D is the dimension of the input vector $\mathbf{x}_n=[x_{1n},\ldots,x_{Dn}]^T$.

It can be proven easily that the columns of matrix $\mathbf{U}$ also correspond to the eigenvectors of the related non-normalized correlation matrix $\mathbf{S}=\Phi\Phi^T$ [23]. However, in kernel methods this correlation matrix is never computed to avoid an explicit mapping of the data. Rather, kernel methods rely on the so-called kernel trick [3,4] which provides an efficient implementation of dot products via their related elements of a suitable kernel matrix. Thus dot products in feature space are evaluated via kernel functions, like radial basis functions (RBF), using the data in input space. For instance, considering a pair of data vectors, the dot product of the mapped data is defined via

$$\phi^T(\mathbf{x}_i)\phi(\mathbf{x}_j):=k(\mathbf{x}_i,\mathbf{x}_j)=\exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{3}$$

where the width parameter σ is to be assigned according to the range of values of the data set. In that way it is very easy to compute the $N\times N$ kernel matrix of dot products $\mathbf{K}$, where each entry (i,j) is the result of the dot product between a pair (i,j) of examples of the training set.

In a classification task, the training set is used to learn the basis vector matrix ($\mathbf{U}$) and its projections $\mathbf{Z}$, and the corresponding labels are used to train the classifier. Using an RBF kernel (Eq. (3)), the number of nonzero eigenvalues depends on the size of the training set but also on the value assigned to σ. Assuming only $L\le N$ eigenvectors of the kernel matrix are chosen, the reduced $N\times L$ eigenvector matrix $\mathbf{V}$ and the corresponding $L\times L$ diagonal eigenvalue matrix $\mathbf{D}$ lead to a new, reduced representation of each training data point with dimension L. Hence, the training set can be represented by an $L\times N$-dimensional matrix $\mathbf{Z}$ of projections onto the reduced set of L basis vectors $\mathbf{U}$ as

$$\mathbf{Z}=\mathbf{U}^T\mathbf{\Phi}=\mathbf{D}^{-1/2}\mathbf{V}^T\mathbf{\Phi}^T\mathbf{\Phi}=\mathbf{D}^{-1/2}\mathbf{V}^T\mathbf{K} \tag{4}$$

By replacing the kernel matrix by its eigenvalue decomposition $\mathbf{K}=\mathbf{V}_N\mathbf{D}_N\mathbf{V}_N^T$, where the matrix of eigenvectors and the matrix of eigenvalues both have dimension $N\times N$, the previous equation can be simplified. As $\mathbf{V}^T\mathbf{V}_N=[\mathbf{I}\quad\mathbf{0}]$ where $\mathbf{I}$ is the $L\times L$ identity matrix and $\mathbf{0}$ is a $L\times(N-L)$ matrix of zeros, the $L\times N$-dimensional matrix of projections in feature space read

$$\mathbf{Z}=\mathbf{D}^{1/2}\mathbf{V}^T \tag{5}$$

It can be shown easily that the new representations of the training data set are uncorrelated, i.e. $\mathbf{ZZ}^T$ is a diagonal matrix. And also notice that a low-rank approximation $\tilde{\mathbf{K}}=\mathbf{VDV}^T$ for the kernel matrix can be obtained by computing $\mathbf{Z}^T\mathbf{Z}$. This low-rank approximation results from the L leading eigenvalues and related eigenvectors.

After learning the subspace model, the test data set is also projected onto the basis vectors $\mathbf{U}$, i.e. $\mathbf{z}_{test}=\mathbf{U}^T\boldsymbol{\phi}(\mathbf{x_{test}})$, thus forming the new representation of the test data. These projections then build the input to the classifier and the corresponding outputs, i.e. the classified test data, are used to evaluate the performance of both, the feature extraction method and the classification method.

However, the following drawbacks of the proposed procedure during training and testing have to be pointed out:

- Kernel methods, employing dual coordinates to represent the model (Eq. (2)) necessitate the storage of the training set even during the test phase to compute the projections of any new test point $\phi(\mathbf{y})$ onto the basis vectors of the model.
- A large kernel matrix $\mathbf{K}$ can render its eigenvalue decomposition unfeasible in practical applications where often the manipulation of large data sets is required.

These issues have been addressed in several kernel based algorithms and different solutions, and strategies have been presented in literature. Some approaches try to select a subset of data within the training set [24–29], others rely on low-rank approximations of the kernel matrices [19,30,7,16,31]. Most of these approximations are based on a similar criterion to optimize the solutions.

2.2. Low-rank approximations to the subspace model

In large training data sets, the computation of the corresponding kernel matrix $\mathbf{K}$ becomes prohibitive. Consequently, its eigenvalue decomposition is often impractical to achieve in real data applications. However, in most of the cases, the kernel matrix is low rank and its eigenspectrum gradually decays to zero. Therefore only the $R<N$ most significant eigenvalues and corresponding eigenvectors need to be computed to yield a good approximation of the kernel matrix.

2.2.1. The Nyström method based on a random subset selection

Few papers [7,16,32] discuss the application of the Nyström method to compute a low rank approximation $\tilde{\mathbf{K}}=\mathbf{V}_R\mathbf{D}_\mathbf{R}\mathbf{V}_R^T$ of the kernel matrix $\mathbf{K}$ where only the R largest eigenvalues $\mathbf{D}_R=diag(\lambda_1,\ldots,\lambda_r,\ldots,\lambda_R)$ and corresponding eigenvectors, forming an $N\times R$ matrix $\mathbf{V}_R$, are computed. The Nyström method is based on the fact that the kernel matrix can be re-written in block notation as follows:

$$\mathbf{K}=\begin{bmatrix}\mathbf{K}_R & \mathbf{K}_{RS}\\ \mathbf{K}_{RS}^T & \mathbf{K}_S\end{bmatrix} \tag{6}$$

Considering that the full matrix has dimension $N\times N$, the upper left block matrix $\mathbf{K}_R$ has dimension $R\times R$, the upper right block matrix $\mathbf{K}_{RS}$ has dimension $R\times S$ and the lower right block matrix $\mathbf{K}_S$ has dimension $S\times S$ where $S=N-R$. Thus, the mapped training data set of dimension N is divided into two subsets of size R and S, respectively. The matrix $\mathbf{K}_R$ represents the kernel matrix within subset $\mathbf{\Phi}_R$ (with R vectors), $\mathbf{K}_{RS}$ is the kernel matrix comprising subsets $\mathbf{\Phi}_R$ and $\mathbf{\Phi}_S$ and $\mathbf{K}_S$ is the kernel matrix of the subset $\mathbf{\Phi}_S$.

The Nyström extensions for the $N\times R$ eigenvector matrix $\mathbf{V}_R$ can be obtained via

$$\mathbf{V}_R^T=\mathbf{H}^T[\mathbf{K}_R\ \ \mathbf{K}_{RS}] \tag{7}$$

where the $R\times R$-dimensional matrix $\mathbf{H}$ is computed using eigenvalue decompositions of certain $R\times R$ matrices computed from the original kernel matrix or from the data set. Different approaches to build such reduced size matrices are discussed in the literature:

- In [16], the data (or its related kernel matrix) is split into subsets (or the matrix into blocks) by randomly selecting the subset $\mathbf{\Phi}_R$ (or the rows/columns) to compute a low rank approximation to the kernel matrix. Alternatively, in [32] the data is clustered into

R clusters, and the centroids are used to form the upper left block $\mathbf{K}_R$ of the kernel matrix $\mathbf{K}$. In both cases the eigenvalue decomposition of $\mathbf{K}_R = \mathbf{B}_R \mathbf{\Gamma}_R \mathbf{B}_R^T$ leads to

$$\mathbf{H} = \mathbf{B}_R \mathbf{\Gamma}_R^{-1} \tag{8}$$

therefore substituting $\mathbf{H}$ yields *non-orthogonal* eigenvectors, i.e., $\mathbf{V}_R^T \mathbf{V}_R \neq \mathbf{I}$ [23].

- An alternative approach, leading to *orthogonal* eigenvectors instead, is proposed in [7]. The proposal adds a second step to the proposal of [16]. The data is transformed using the eigendecomposition of $\mathbf{K}_R$, and an $R \times R$ matrix is computed with the transformed data. The eigendecompositions of the $R \times R$ matrices of both steps are used to compute $\mathbf{H}$.

In the next section we will propose an equivalent solution which substitutes the random selection to form $\mathbf{K}_R$ and the data transformation to compute the second $R \times R$ matrix. We discuss the properties of this approach and compute the corresponding matrix $\mathbf{H}$.

2.2.2. Nyström method based on an incomplete Cholesky decomposition

Instead of randomly choosing a subset of the training set, an incomplete Cholesky decomposition of the kernel matrix $\mathbf{K}$ can also be used as an alternative to form the block matrix $\mathbf{K}_R$ in Eq. (7) [31]. In the following we discuss this decomposition and its connection with the Nyström approximation.

The incomplete Cholesky decomposition of the kernel matrix $\mathbf{K}$, using a symmetric pivoting scheme to obtain an $R \times N$ matrix $\mathbf{C}$, reads

$$\mathbf{C} = [\mathbf{L} \;\; \mathbf{L}^{-T} \mathbf{K}_{RS}] \tag{9}$$

The matrix $\mathbf{L}$ represents a triangular matrix corresponding to the Cholesky factorization of $\mathbf{K}_R = \mathbf{L}^T \mathbf{L}$. Notice that the matrix $\mathbf{L}$ arises naturally from the incomplete Cholesky decomposition [20], while the concomitant symmetric pivoting scheme leads to the selection of the subset $\mathbf{\Phi}_R$ of the training set.

Considering the $R \times N$ matrix $\mathbf{C}$, the eigenvalue decomposition of its related $R \times R$ correlation matrix $\mathbf{Q}$ can be computed via

$$\mathbf{Q} = \mathbf{C}\mathbf{C}^T = \mathbf{E}_R \mathbf{D}_R \mathbf{E}_R^T \tag{10}$$

The $R \times R$-dimensional matrix $\mathbf{E}_R$ contains in its columns R eigenvectors and the R diagonal entries of $\mathbf{D}_R$ represent the related eigenvalues. The latter are identical to the leading R eigenvalues of the related kernel matrix $\tilde{\mathbf{K}} = \mathbf{C}^T \mathbf{C} = \mathbf{V}_R \mathbf{D}_R \mathbf{V}_R^T$. The result of this eigenvalue decomposition together with the Cholesky decomposition of $\mathbf{K}_R$ lead to

$$\mathbf{H} = \mathbf{L}^{-1} \mathbf{E}_R \mathbf{D}_R^{-1/2} \tag{11}$$

Substituting this result into Eq. (7), the $N \times R$ eigenvector matrix $\mathbf{V}_R$ of the kernel matrix $\tilde{\mathbf{K}}$ can be written as

$$\mathbf{V}_R = \begin{bmatrix} \mathbf{K}_R \\ \mathbf{K}_{RS}^T \end{bmatrix} \mathbf{L}^{-1} \mathbf{E}_R \mathbf{D}_R^{-1/2} \tag{12}$$

The eigenvectors, i.e. the columns of the $N \times R$ matrix $\mathbf{V}_R$, are *orthogonal* thus yielding $\mathbf{V}_R^T \mathbf{V}_R = \mathbf{I}$. Using the spectral theorem $\mathbf{V}_R \mathbf{D}_R \mathbf{V}_R^T$ to approximate the kernel matrix, it can be proven that the last block of the kernel matrix (Eq. (18)) is approximated by $\tilde{\mathbf{K}}_S = \mathbf{K}_{RS}^T \mathbf{K}_R^{-1} \mathbf{K}_{RS}$. This approximation holds also if $\mathbf{H}$ is computed as described by Eq. (8), but in that case the values of the eigenvalues have to be properly scaled (see [16] for more details).

2.2.3. Basis vectors and Cholesky decomposition

In Eq. (12), the eigenvalues should be arranged in decreasing order, and the related eigenvectors, forming the columns of matrix $\mathbf{E}_R$, should be ordered accordingly. As before, $L \leq R$ eigenvalues and their related eigenvectors, forming the $R \times L$ eigenvector matrix $\mathbf{E}$, can be selected to project the data onto only L directions. The projections of the training data set, in feature space then become

$$\mathbf{Z} = \mathbf{U}^T \mathbf{\Phi} = \mathbf{D}^{1/2} \mathbf{V}^T = \mathbf{E}^T \mathbf{L}^{-T} \mathbf{\Phi}_R^T [\mathbf{\Phi}_R \;\; \mathbf{\Phi}_S] \tag{13}$$

Consequently, the basis vector matrix $\mathbf{U}$ can be written as

$$\mathbf{U} = \mathbf{\Phi}_R \mathbf{L}^{-1} \mathbf{E} = \mathbf{\Phi}_R \mathbf{A} \tag{14}$$

Hence, the dual form of the basis vector matrix of the subspace model is written only employing a subset of the training data set. This substantially reduces the storage requirements and the computational load during testing the classifier. Also note that the L basis vectors form an *orthogonal* basis in feature space, i.e., $\mathbf{U}^T \mathbf{U} = \mathbf{I}$.

As an alternative to Eq. (13), the projections of the training data set can be written as

$$\mathbf{Z} = \mathbf{U}^T \mathbf{\Phi} = \mathbf{E}^T \mathbf{C} \tag{15}$$

Then it can be proven that the projections are *uncorrelated* even when the dimension of the data is reduced to $L \leq R$, i.e, when the number of columns of $\mathbf{E}_R$ is $L \leq R$. But the projections $\mathbf{Z}$ can be used to compute the low-rank approximation of the kernel matrix only when the dimension of the new representation is R. To see this notice that $\mathbf{E}_R \mathbf{E}_R^T = \mathbf{I}$ only holds when the eigenvector matrix has dimension $R \times R$.

In summary, the incomplete Cholesky decomposition with a symmetric pivoting scheme provides the means to compute matrix $\mathbf{C}$ and *simultaneously* identify the elements of subset $\mathbf{\Phi}_R$. Furthermore, in [14,20] an implementation of the incomplete Cholesky decomposition of a kernel matrix (using the RBF kernel function), without explicitly computing the kernel matrix, is introduced. In the Appendix, an adaptation of this implementation to compute the basis vector matrix as described by Eq. (14) is presented.

2.3. Centering the data

All previous deductions were conducted assuming that the data is centered. In input space, centering can be considered a preprocessing step which must be accomplished before computing the correlation matrix, and before projecting any new data vector onto its related eigenvectors. But in kernel methods, centering has to be integrated in the projection step. In the following, this centering procedure will be discussed in case of a complete and an incomplete training set, respectively.

2.3.1. KPCA with a complete training set

In feature space, centering the mapped data is a more elaborate procedure performed mostly during computing the projections. To facilitate the exposition, let us consider a vector $\mathbf{m} = [m_1, \ldots, m_N]^T$ with N elements, all of which equal to $1/N$, and a matrix $\mathbf{M}$ with N identical column vectors $\mathbf{m}$. Then centering the training data set and projecting a new data point $\phi(\mathbf{y})$ onto the subspace directions, the following operations need to be integrated into the dot product:

$$(\mathbf{\Phi} - \mathbf{\Phi}\mathbf{M})^T (\phi(\mathbf{y}) - \mathbf{\Phi}\mathbf{m}) \tag{16}$$

The first term removes the mean to the training data set, while the second subtracts the mean of the training set from new point $\phi(\mathbf{y})$. The manipulation of the previous equation leads to the following vectors:

- $\mathbf{k}_1 = \mathbf{\Phi}^T \phi(\mathbf{y})$ is a vector of dot products between the point and the training set.
- $\mathbf{k}_2 = \mathbf{M}^T \mathbf{\Phi}^T \phi(\mathbf{y})$ is a vector with identical values in all entries. It is the mean of the dot products between the new point and the training set,

- $\mathbf{k}_3 = \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{m}$ is a vector containing the mean values of the N rows of the kernel matrix $\mathbf{K}$, and finally
- $\mathbf{k}_4 = \mathbf{M}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{m}$ is a vector with all entries equal to $(1/N^2)$ $\sum_i\sum_j k(i,j)$, where $k(i,j)$ is the (i,j)-th entry of the kernel matrix.

According to Eq. (4), the feature space projections of the input data point $\mathbf{y}$ then read

$$\mathbf{z}_y = \mathbf{D}^{-1/2}\mathbf{V}^T(\mathbf{k}_1 - \mathbf{k}_2 - \mathbf{k}_3 + \mathbf{k}_4) = \mathbf{D}^{-1/2}\mathbf{V}^T(\mathbf{I} - \mathbf{M}^T)(\mathbf{k}_1 - \mathbf{k}_3) \tag{17}$$

The summands related with $\mathbf{k}_3$ and $\mathbf{k}_4$ only depend on the training set and are present in every data point projected into the subspace $\mathbf{U}$. Hence, they can be stored in advance and constitute a bias term that is present in every projection.

It can be shown easily that projecting the complete training set $\mathbf{\Phi}$ to obtain $\mathbf{Z}$, the terms $\mathbf{k}_2$, $\mathbf{k}_3$ and $\mathbf{k}_4$ arise from the centered kernel matrix

$$\mathbf{K}_c = (\mathbf{I} - \mathbf{M})\mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{I} - \mathbf{M}) \tag{18}$$

where $\mathbf{I}$ is an $N \times N$ identity matrix. Then, to accomplish *uncorrelated* projections of the training data set, in Eq. (4) the kernel matric $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$ should be replaced by $\mathbf{K}_c$ as given by Eq. (18), and the matrices $\mathbf{V}$ and $\mathbf{D}$, determining the subspace model, should be obtained from the eigenvalue decomposition of $\mathbf{K}_c$. Finally notice that with an RBF kernel the dot products in feature space are always less than unity (see Eq. (3)) and that the contribution of the terms $\mathbf{k}_3$ and $\mathbf{k}_4$ in Eq. (17) depend on the parameter σ of the RBF kernel.

2.3.2. KPCA with a reduced training set

As described in the Appendix, to avoid a computationally costly direct evaluation of the kernel matrix, an incomplete Cholesky decomposition can be performed having as input the complete training set. The outcome is a dimension reduced $R \times N$ matrix $\mathbf{C}$ with $R \leq N$. A centered version of a low rank approximation of its related kernel matrix can then be obtained via

$$\tilde{\mathbf{K}}_c = (\mathbf{I} - \mathbf{M})\mathbf{C}^T\mathbf{C}(\mathbf{I} - \mathbf{M}) \tag{19}$$

where the mean $\mathbf{Cm}$ is subtracted from every column of $\mathbf{C}$. In that case, the eigenvectors $\mathbf{E}_R$ must be computed from the correlation matrix $\mathbf{Q}_c$ computed after centering the matrix $\mathbf{C}$. Then, the term $\mathbf{b} = \mathbf{E}^T\mathbf{Cm}$ must be subtracted from every data point projected into the subspace $\mathbf{U}$ (see Eq. (14)) according to

$$\mathbf{z}_y = \mathbf{U}^T\phi(\mathbf{y}) - \mathbf{b} \tag{20}$$

In the Appendix, an implementation of the greedy KPCA algorithm, as presented in this work, is given in MATLAB, and the centering of the data is added as an option to be selected by the user.

3. Numerical simulations

The different subspace feature extraction methods discussed above are evaluated by comparing the performance of extracted features in a classification task employing two different simple classifiers. The goal is to compare the quality of the generated features, not the classification ability of the classifiers. The generalization errors of the latter allow to quantify the quality of the features extracted by projecting the data onto different subspace models. We carried out experiments on artificial and real world data sets available from public data repositories. We consider two groups of data: one consisting of 13 benchmarks and the other of the well-known USPS data set. To treat all generated features on an equal footing, two simple classifiers were used for classification: a nearest neighbor (NN) classifier and a linear discriminant function (RL). With a NN classifier, each element of the test set is classified according to its nearest neighbor in the training set. In case of linear discriminant functions, the weight vectors are learned within a training session by employing a training data set and by using the mean-squared-error criterion [33]. Each element in the test set is assigned to the class whose discriminant function has the largest value.

Table 1
Data set description: D is dimension of the input vector $\mathbf{x}_n$ and N is the size of training set. Classification of raw data with a nearest neighbor (NN) classifier and a linear discriminant function (RL).

	Best [21]	D	N	NN	RL	t-test
***B. Cancer* (*BC*)**	25.9 ± 4.6	9	200	32.5 ± 4.8	26.9 ± 2.7	$\oplus$
***Diabetis* (*Di*)**	23.5 ± 1.7	8	468	30.1 ± 2.0	23.4 ± 1.7	$\oplus$
***German* (*Gr*)**	23.6 ± 2.1	20	700	29.4 ± 2.4	24.3 ± 2.9	$\oplus$
***Heart* (*Hr*)**	16.0 ± 3.3	13	170	23.2 ± 3.7	15.8 ± 3.1	$\oplus$
***F. Solar* (*FS*)**	32.4 ± 1.8	9	666	39.0 ± 4.9	33.5 ± 1.5	$\oplus$
***Thyroid* (*Ty*)**	4.4 ± 2.2	5	140	4.3 ± 2.2	14.7 ± 3.1	$\ominus$
***Titanic* (*Ti*)**	22.4 ± 1.0	3	150	33.0 ± 11.0	22.6 ± 1.0	$\oplus$
***Twonorm* (*Tn*)**	2.7 ± 0.2	20	400	6.6 ± 0.7	2.6 ± 0.17	$\ominus$
***Image* (*Im*)**	2.7 ± 0.7	18	1010	3.3 ± 5.4	16.5 ± 0.9	$\ominus$
***Ringnorm* (*Rg*)**	1.6 ± 0.1	20	400	35.1 ± 1.3	24.7 ± 0.7	$\ominus$
***Splice* (*Sp*)**	9.5 ± 0.7	60	1000	28.8 ± 1.5	16.2 ± 0.6	$\ominus$
***Waveform* (*Wv*)**	9.8 ± 0.8	21	400	15.8 ± 0.6	14.8 ± 0.2	$\ominus$
***Banana* (*Ba*)**	10.7 ± 0.4	2	400	13.6 ± 7.0	46.9 ± 7.0	$\ominus$

3.1. Subspace model parameters

To establish a subspace model and compute its basis vectors, thereby using as kernel an RBF function to evaluate the dot products, it is needed to assign a value to the width parameter σ of the kernel. This parameter is often a variable of the experimental studies [34], or it is optimized applying a cross-validation strategy including training and test data sets [24]. The value of σ determines the sparsity of the kernel matrix, because the entries are in the range]0,1]. Furthermore, the trace of the kernel matrix, corresponding to the sum of its eigenvalues, is always equal to $\mathrm{tr}(\mathbf{K}) = N$ irrespective of the value of σ. For large σ, the first eigenvalue generally dominates the eigenvalue spectrum, while for smaller values a more gradual decrease of the eigenvalues results. The choice of sigma also interferes with the size R of the subset of basis vectors forming the subspace model of a greedy KPCA. In case of a gradual decrease of the eigenvalues, the incomplete Cholesky decomposition will automatically choose the complete training set, i.e. $R = N$. Experimentally we conclude that by choosing σ as the average of the distances of the training data to their mean $\mathbf{x}_{mean}$

$$\sigma = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{x}_{mean}\| \tag{21}$$

a smooth eigenspectrum decay is achieved.

3.2. Benchmark data sets

A collection of benchmark data sets can be found on the following web site[1]: Table 1 resumes the information of the 13 data sets. All data sets have 100 random partitions of pairs of training and test sets, except *Splice* and *Image* which have 20 partitions. These benchmark data sets will be used to extract features by applying the projective subspace techniques discussed above. Feeding these features into the two classifiers just mentioned and comparing the classification results enables us to quantify the suitability of the features extracted with the different subspace models. Especially feature degradation by greedy approximations may be quantified this way. Several other algorithms

[1] Accessible at http://ida.first.fraunhofer.de/projects/bench.

[21,24,29,31] have been applied already to each partition data set. All of them address binary classification problems and use various kinds of features. The second column of Table 1 summarizes the average and the standard deviation of the best generalization errors published in [21]. These data provide sort of a reference with which we compare our results. On the one hand this proves that our results are generally appropriate and competitive, and on the other hand it allows us to evaluate the usefulness of the features generated with the different projective subspace techniques.

Using these training and test sets, algorithms and methods have been compared pairwise, and the difference between error percentages, achieved by any pair of algorithms/methods, has been evaluated statistically. A student t-test was performed, applying a 95% significance level, by considering the two hypotheses

$$H0 : \mu_1 = \mu_2 \quad \text{and} \quad H1 : \mu_1 \neq \mu_2$$

where μ_1 and μ_2 represent the average test errors achieved by two methods, respectively. This statistical test has been carried out in an attempt to reject $\ominus$ or accept $\oplus$ the null hypothesis ($H0$) with the significance level 95%.

3.2.1. Raw data sets

First, both classifiers were applied to the raw data sets to provide sort of a base level where the features correspond to the observations themselves. The results of the classification lead us to organize the data sets into two groups (see table). In group 1, comprising the first eight data sets, at least one of the classifiers achieves an error rate comparable to the ones published in [21]. This means that for these data sets the observations themselves represent features informative enough to achieve a near optimal classification. In group 2, encompassing the remaining five data sets, the performance was far from the results presented in [21] indicating that the observations themselves represent suboptimal features which were not sufficiently informative. In group 1, the RL classifier achieves the better performance, with the exception of the *thyroid* data set. In the second group, the best results are split between the two classifiers. A student t-test is used to compare the best result, achieved with either an NN or an RL classifier, with the ones published. The null hypothesis $H0$ is rejected in the second group but is accepted in the first group, except for two data sets (see last column of Table 1).

3.2.2. Transformed data sets

Next, both classifiers were applied to the features generated from the data sets via the projective subspace techniques described. In feature space, the number of basis vectors which can be computed is at most equal to the number of training examples (N). Both versions of KPCA are used to compute the subspace model: KPCA depends on the full training set while greedy KPCA only depends on a subset with $R < N$ elements of the training set. Both subspace models are computed using centered versions of the kernel matrices as proposed before. Thus Eq. (17) is used to project a data point onto the KPCA model, and Eq. (20) is used to project the data onto a lower-dimensional greedy KPCA model. In both cases, the number of projections was varied from $L=1$ up $L=\min(R)$ in the training sets of each data set. The average error rate is computed using all test sets for each L and the minimum average error is reported.

In Table 2, average error rates and standard deviations are shown for both classifiers in case of a KPCA model to extract appropriate features. In addition, column I1 shows the result of the student t-test between the results of [21] and either the NN or the RL classifier. The null hypothesis H0 is accepted in all but the German data set indicating that the average error rate is in accord with the best results obtained in [21]. As only the simplest classifiers have been applied this hints to the excellent quality of the features extracted. It is also obvious that in group 1 there is no significant improvement of the classification result when using non-linear features of the data sets. In fact it can be seen that with the exception of the thyroid data set and the RL classifier as well as the Twonorm data set and the NN classifier, hardly any improvement could be reached compared to employing the bare observations as features. Only in the two cases mentioned, the transformation of the data in group 1 via a KPCA subspace model could improve the features extracted and render the classification results consistent internally and with the best classification result reported in the literature. The situation is completely different with the data sets from group 2, where the data transformation via a KPCA subspace model considerably improved the quality of the generated features and rendered them informative enough to compare favourably with the best classification results reported in the literature.

Table 2
KPCA: Error rate (%) using KPCA for feature extraction and a nearest neighbor (NN) as well as a linear discriminant (RL) classifier. **t-test**: results of a student t-test: Best results reported by [21] versus KPCA (column I1) and KPCA versus KPCA$_D$ (column I2), where $\oplus$ accepts $H0$ and $\ominus$ rejects $H0$. D is the dimension of the input data vector and L is the dimension in feature space.

	N	D	KPCA				t-test	
			L	NN	L	RL	I1	I2
BC	200	9	7	32.5 ± 4.8	21	25.2 ± 4.5	$\oplus$	$\oplus$
Di	468	8	17	25.3 ± 1.8	10	23.2 ± 1.6	$\oplus$	$\oplus$
Gr	700	20	12	30.0 ± 2.5	12	23.3 ± 2.1	$\ominus$	$\oplus$
Hr	170	13	8	22.7 ± 3.4	12	15.8 ± 3.0	$\oplus$	$\oplus$
FS	666	9	55	32.2 ± 0.5	25	32.1 ± 0.6	$\oplus$	$\oplus$
Ty	140	5	6	4.0 ± 2.2	15	5.8 ± 2.4	$\oplus$	$\oplus$
Ti	150	3	9	32.3 ± 1.1	10	22.3 ± 1.0	$\oplus$	$\oplus$
Tn	400	20	1	3.4 ± 0.4	1	2.3 ± 0.1	$\oplus$	$\oplus$
Im	1010	18	23	2.8 ± 0.6	75	7.9 ± 1.3	$\oplus$	$\ominus$
Rg	400	20	40	3.5 ± 0.4	25	1.6 ± 0.1	$\oplus$	$\ominus$
Sp	1000	60	600	7.5 ± 2.6	720	4.3 ± 2.1	$\oplus$	$\ominus$
Wv	400	21	29	9.7 ± 0.7	2	12.0 ± 0.8	$\oplus$	$\ominus$
Ba	400	2	5	13.6 ± 0.4	34	10.7 ± 0.4	$\oplus$	$\ominus$

In most cases, the minimum error rate is achieved using a number ($L > D$) of features higher than the dimension (D) of the raw data. One of the exceptions is the *twonorm* where $L=1$ and where a PCA model yields a similar performance if the data is projected onto only the leading eigenvector [31]. An additional t-test was performed to study how the number of projections affects the results. The error rates just discussed were compared to error rates achieved when the number of projections was $L=D$. Column I2 of Table 2 reports the t-test results: the null hypothesis $H0$ is rejected in group 2 and accepted in group 1. Furthermore, notice that the RL classifier yields, with the exception of the *waveform* set, the best classification only if the number of features is larger than the dimension of the raw data, i.e. when $L > D$.

Table 3 shows the classification results obtained by employing greedy KPCA to extract the features and to compute the projections of the data onto the feature space coordinates. The column R shows the range of values for the size of the subset Φ_R in the training sets. The average error rates obtained are similar to the ones computed with KPCA and corroborate that the computationally much less demanding greedy approximation extracts features of comparable quality and information content. The null hypothesis is now accepted for every data set (see column I1).

The number of projections, i.e. features, used with both classifiers, however, was not always identical as the greedy KPCA algorithm was implemented using the same threshold for the approximation error (see Appendix) irrespective of the data set. But considering that greedy KPCA uses an approximation of the kernel

matrix, some variation has to be expected. In fact, the average of the relative subset sizes (R/N) of the subspace models fluctuates strongly between the different data sets as shown in Fig. 2. Small subset sizes mostly result from low dimensional data sets like the banana, titanic and thyroid data sets. Empirically, a small subset size correlates with a steep rise of the cumulative sum of eigenvalues of the kernel matrix, meaning that these eigenvalues decrease quickly. Thus the normalized cumulative sum of the eigenvalues, if the latter were arranged in decreasing order, (see (22)) can be used to illustrate the decrease characteristics of the eigenvalues of the kernel matrix:

$$S(p) = \frac{\sum_{i=1}^{p} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \quad \text{with } \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \tag{22}$$

Fig. 2 shows the normalized cumulative sum $S(p)$ of the data sets of group 2. It can be seen that an abrupt increase on $S(p)$ corresponds to a small value for R, and if $S(p)$ converges slowly to the maximum value, then R has a higher value. The different decay profiles thus directly translate into different values for R/N. Therefore in group 2 kernel features lead to an improvement in the performance of both classifiers, either using KPCA or greedy KPCA algorithms to compute the basis vectors. And for both groups of data the performance of the classifier (either NN or RL) based on the features extracted in various ways is never worse than with the raw features, rather it is often much improved.

Table 3
Error rate (%) using greedy KPCA. Results of t-test: greedy versus KPCA, where $\oplus$ accepts $H0$ and $\ominus$ rejects $H0$. The entries of column R are the range of values for the subset size in the training sets.

	N	Greedy KPCA					t-test
		R	L	NN	L	RL	I1
BC	200	[89,100]	7	32.5 ± 4.8	22	25.2 ± 4.5	$\oplus$
Di	468	[141,155]	61	30.2 ± 1.9	10	23.1 ± 1.6	$\oplus$
Gr	700	[412,428]	13	29.1 ± 2.4	12	23.4 ± 2.3	$\oplus$
Hr	170	[116,124]	48	22.79 ± 2.9	11	15.8 ± 3.1	$\oplus$
FS	666	[65,83]	65	36.8 ± 0.7	48	33.8 ± 0.6	$\oplus$
Ty	140	[26,34]	6	3.9 ± 2.2	25	5.3 ± 2.3	$\oplus$
Ti	150	[6,9]	6	31.2 ± 1.4	6	21.8 ± 1.0	$\oplus$
Tn	400	[320,325]	1	3.5 ± 0.6	1	2.3 ± 0.1	$\oplus$
Im	1010	[105,123]	21	2.9 ± 0.7	80	8.1 ± 1.2	$\oplus$
Rg	400	[264,275]	45	3.8 ± 0.4	31	1.7 ± 0.1	$\oplus$
Sp	1000	[871,889]	620	7.7 ± 2.6	764	4.4 ± 2.1	$\oplus$
Wv	400	[285,299]	30	9.8 ± 0.3	2	12.0 ± 0.7	$\oplus$
Ba	400	[13,16]	13	13.6 ± 0.7	5	10.8 ± 1.8	$\oplus$

3.3. USPS data set

The USPS data set is a benchmark data set used in many works. It is accessible at www.kernel-machines.org and is divided into a training data set with 7921 images and a test data set with 2007 images. The training data set is very large, so not every algorithm could be applied to the complete training set. Hence, two different training data sets were formed to comprise 10% and 100% of the available data, respectively. Each image consists of a handwritten digit comprising 16×16 pixels. Concatenating these pixels into a row vector, the resulting data vector in input space $\mathbf{x}_n$ had dimension $D = 256$.

3.3.1. The raw data set

The best classification results obtained on this data set in literature report an error rate in the range 0.04–0.05 [4]. In a first attempt we considered the complete training set as features to be employed for training an NN or a RL classifier. Both classifiers were then used to classify the test data set. With an NN classifier an error rate equal to 0.056 was achieved, while a RL classifier yielded an error rate equal to 0.131. Note that especially with an NN classifier the raw data as features provide already enough information to achieve a near optimal classification. Generally, these error rates provide a baseline for comparison with results obtained employing more complex feature generation strategies.

3.3.2. PCA feature extraction

First, the basis vector model was computed by principal component analysis (PCA) employing both the complete and the reduced-size training sets. For that, the 256×256-dimensional covariance matrices of the training data sets were computed, and the basis vectors of he subspace model corresponded to their eigenvectors. Arranging the latter according to their related eigenvalues in descending order reveals that the data from both

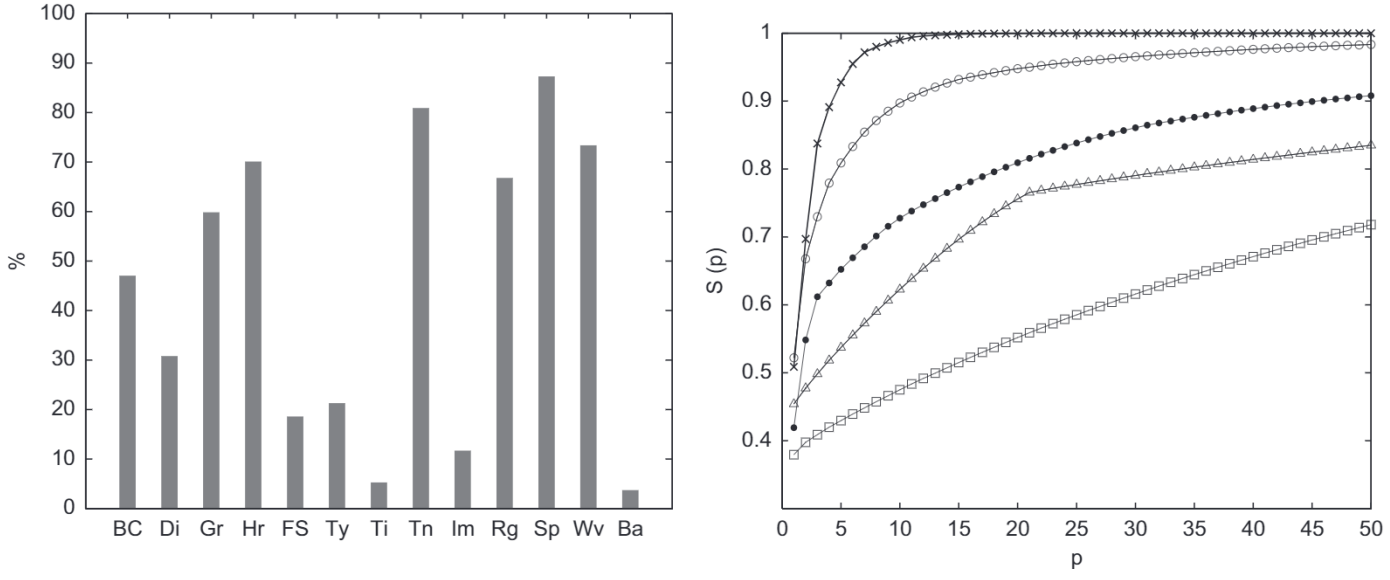

Fig. 2. Relative size of the subsets and cumulative sum of normalized eigenvalues $S(p)$. Left: ratio of the average size of subset (R) vs the size of the full training set (N) (%). Right: $S(p)$ of the data sets Im (°), Rg (Δ), Sp (□), Wv (•) and Ba (×).

training sets predominantly spread in only 50 of the 256 directions of the input space. The NN and RL classifiers were trained employing different numbers of extracted features as inputs $\mathbf{z}_k$. These numbers in each case corresponded to the number L of vectors taken to form the basis vector matrix.

In the PCA model, L varies in the range $L=1$–100. Fig. 5 illustrates the classification error of the NN classifier upon varying the number of projections in input space which represent the features used for classification. The best performance yields an error rate of 0.052 and is achieved with the complete training data set using 50 projections roughly. With the reduced training data set the error rate is around 0.1. This result has to be expected as the covariance matrix exhibits approximately 50 significant eigenvalues, with the remaining eigenvalues being very close to zero. Increasing the number of PCA projections to $L>50$ causes a slight increase of the error rate to 0.057. Fig. 6 shows the corresponding results for the RL classifier. Using PCA projections as features, the error rate of this classifier amounts to 0.14. This classifier yields similar error rates with both training sets indicating that the reduced feature set already contains enough information to achieve a good classification. But considering the best classification rates reported in the literature, these features turn out to be suboptimal still.

Table 4
Size (R) of subset $\mathbf{\Phi}_R$ for different values of σ using training sets with different sizes (N).

	N					
	100%			10%		
σ	5	8	10	5	8	10
R	1807	241	91	335	132	63

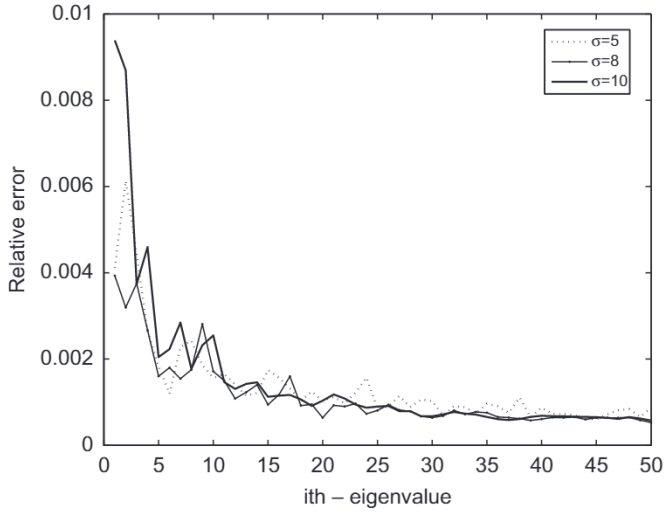

Fig. 3. Relative error between the largest eigenvalues of the kernel matrices deduced from KPCA and greedy KPCA, respectively, $N=729$.

3.3.3. *Kernel feature extraction*

As mentioned already, the training data set is too large for KPCA to be applied to the complete training data set. So results could be obtained only with the reduced training data set or by applying the greedy KPCA algorithm. The latter subspace model was computed without centering the data, thus eliminating the bias term of Eq. (20) and without centering $\mathbf{C}$. Table 4 presents the size (R) of subset Φ_R for the complete and the reduced training set, respectively, and for different values of the width parameter σ of the RBF kernel. Note that $\sigma=5$ corresponds to the value computed by Eq. (21).

For the reduced training data set ($N=729$), the KPCA subspace model, without centering, can be computed as well. Then the resulting eigenvalues λ_i can be compared to the eigenvalues $\tilde{\lambda}_i$ of the greedy KPCA model. Fig. 3 shows the relative error defined as

$$er(i)=\frac{|\lambda_i-\tilde{\lambda}_i|}{\lambda_i}=\left|1-\frac{\tilde{\lambda}_i}{\lambda_i}\right| \tag{23}$$

This relative error falls into the same range for all σ values of the RBF kernel, and it is larger for the leading eigenvalues. Fig. 4 shows the difference between the cumulative sums of eigenvalues $S(p)$ for different parameters σ of the RBF kernel when the greedy KPCA subspace model is employed to extract proper features from either the reduced training data set, $N=729$, or the complete training data set, $N=7921$. The relative value of the first eigenvalue ($S(1)$) is similar for both training sets but by comparing $S(p)$ of both training sets, the same absolute level of $S(p)$, corresponding to an identical amount of explained variance of the data, is achieved for different values of p. This comparison thus explains the values of the different subset sizes R of Table 4.

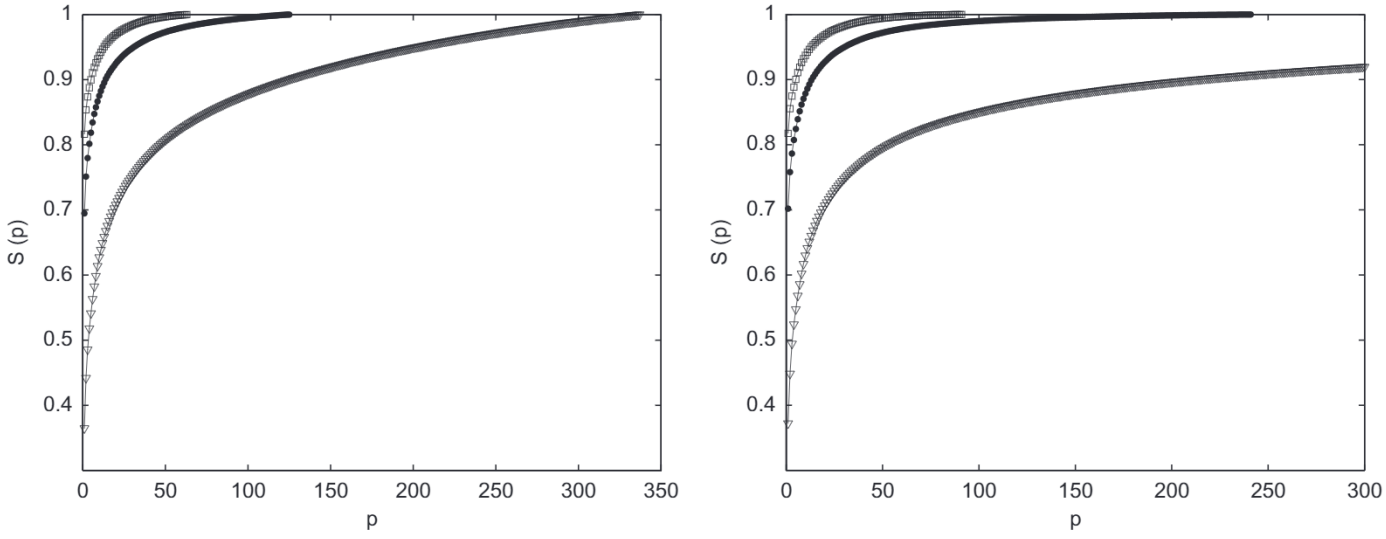

Fig. 4. Normalized cumulative sum of eigenvalues for the reduced training set, left: $N=729$, and the complete training set, right: $N=7291$. The different curves correspond to *top*: $\sigma=10$, *middle*: $\sigma=8$ and *down*: $\sigma=5$.

In the greedy KPCA model, L varies in the range $L=1$ to $L=R$. Fig. 5 illustrates the classification rates achieved with the NN classifier upon varying the number of features extracted from the training data sets. If the features, i.e. the subspace projections of the data, were extracted from the complete training data set, the smallest error rates, amounting to 0.05, were achieved. Features extracted from the reduced training data set resulted in an error rate of 0.1 roughly. Hence, using either linear features (PCA) or kernel features (greedy KPCA), a similar error rate 0.05 is achieved with the NN classifier. Employing $L > 50$ projections, the error rate increases slightly except when the features were computed applying a width $\sigma = 5$ of the RBF kernel.

Fig. 6 shows the classification results if the same features were classified with the RL classifier. Here the error rate does not have a so clear dependence on the size of the training set. Using as input features deduced from greedy KPCA subspace projections in feature space results in an error rate for classification of 0.09. It becomes roughly independent of the size of the kernel for a number of features beyond $L > 100$. Note that this is about twice the number of features than in case of linear features extracted with PCA. But compared to the classification results obtained with linear features (PCA), the nonlinear features extracted with greedy KPCA yield a much lower error rate which is roughly one third of the PCA error rate. The results presented in [4], employing a training data set of 3000 examples, show a similar tendency: the linear SVM classifier performs better using nonlinear features computed with KPCA instead of the linear features computed with PCA. In [4] 2048 projections in KPCA feature space were calculated, yielding an improvement in error rate to 0.046 if a polynomial kernel is used. This small decrease in classification error indicates that the choice of the classifier is less critical than the feature generation procedure.

In projective kernel subspace techniques, the number (R) of basis vectors has the highest value when $\sigma = 5$, but the performance of the linear discriminant classifier RL does not change with increasing L after $L=100$. On the other hand, for other values of σ, the best performance is achieved using projections onto all the available basis vectors. All figures demonstrate that the RBF kernel with $\sigma = 5$ shows the best performance. However, when $\sigma = 8$, in Fig. 6, a similar performance is visible when L reaches close to R. And notice that this represents the best tradeoff between performance and storage requirements during the testing phase as the subspace model is described using $R=241$ training examples.

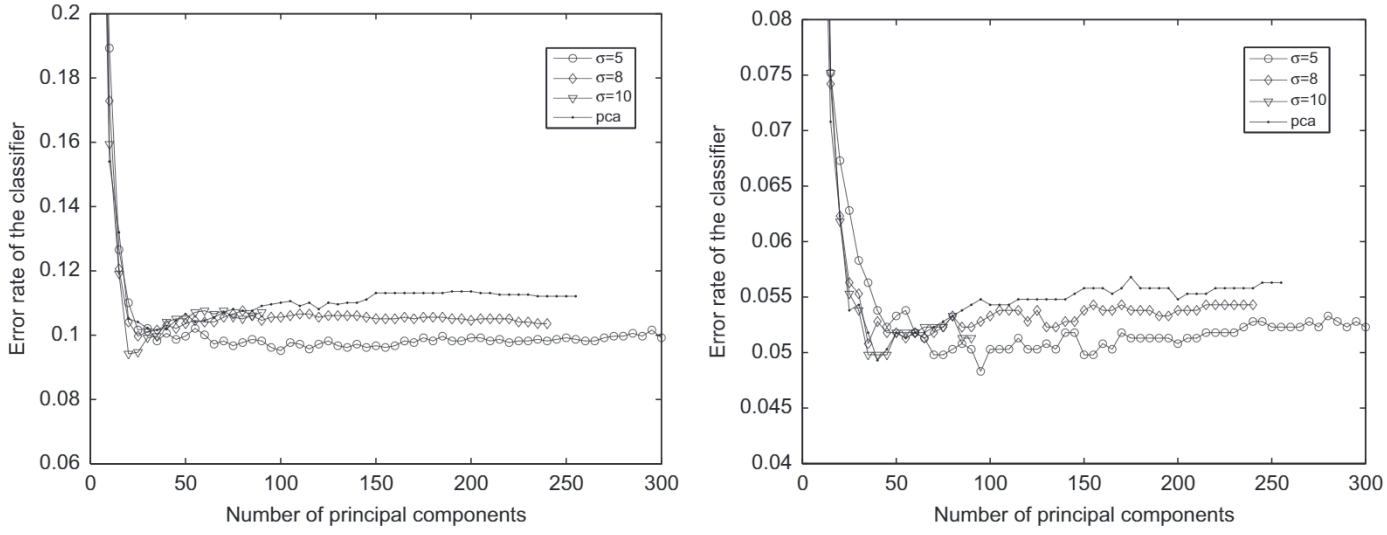

Fig. 5. Performance of NN using projections in input space (PCA) and in feature space. Training set with: 729 (left) or 7291 (right) images.

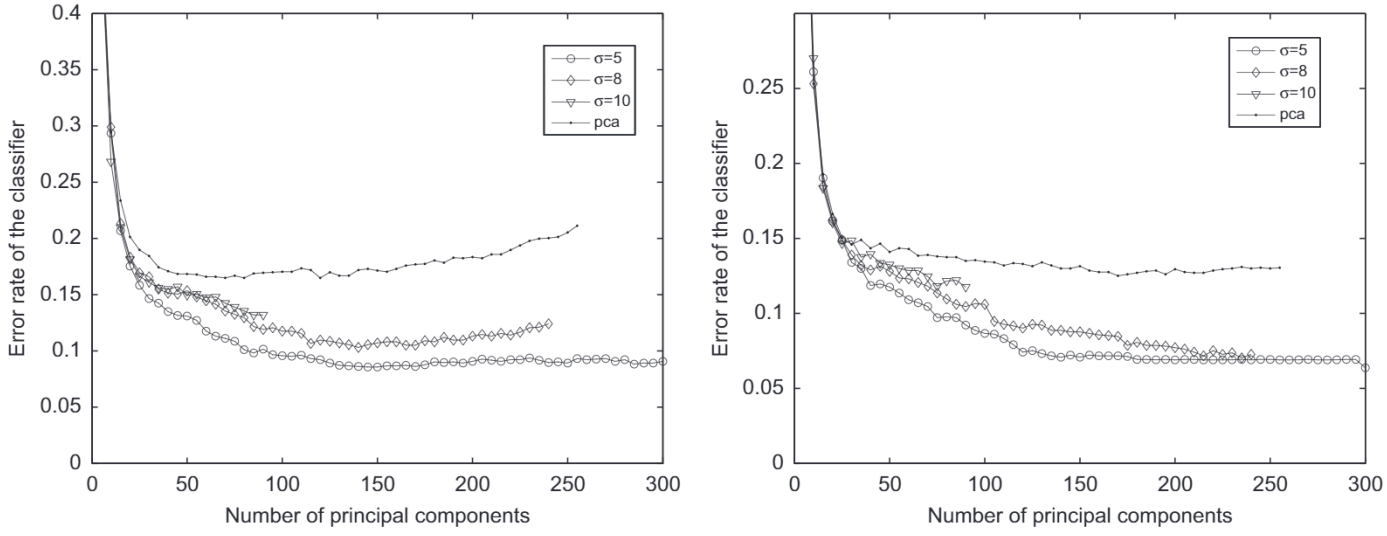

Fig. 6. Performance of the RL using projections in input space (PCA) and in feature space. Training set with: 729 (left) or 7291 (right) images.

4. Concluding remarks

In this work kernel projective subspace techniques are concisely described using an algebraic approach which relies on the dual form of the basis vector model. The latter expresses the basis vectors in terms of the training data and consequently every data manipulation is always casted into the dot products as it has to be in kernel methods. This approach highlights the importance of the kernel matrix to estimate the subspace model. Furthermore, low-rank approximations of the kernel matrices, based on Nyström extensions, can be easily integrated in such a formalism. Furthermore, it was proven that the incomplete Cholesky decomposition with symmetric pivoting provides the means to achieve a computationally less demanding solution, namely the greedy KPCA. It has been shown also how an algorithm can be constructed, for an RBF kernel, to perform an incomplete Cholesky factorization and simultaneously identify the subset of data forming the dual basis vector model.

Experiments show that these projective subspace techniques, cast into a greedy KPCA approach, achieve a performance which is similar to a full KPCA analysis. It is demonstrated also that the greedy KPCA model performs well on a very large data set like USPS data set.

Although it is often assumed that extracting non-linear features always results in an increase of performance in classification, our results do not give this indication for all data sets. The reason is mostly related with the data characteristics, i.e. its information content relevant for classification. In some benchmark data sets the raw data taken as features already contained enough information to yield a good classification and any more sophisticated feature extraction procedures, as the ones discussed in this study, could not improve the performance of the classifiers any further. In other benchmark data sets as well as the USPS data set sophisticated feature extraction procedures substantially improved the classification results. Feature extraction also strongly depends on the parameters of the subspace model, especially on the parameter σ of the RBF kernel functions. Simulations showed that it is possible to estimate an appropriate value of this parameter thus achieving a good tradeoff between the eigenvalue decay and the number of non-zero eigenvalues of the kernel matrix and still resulting in a good classification performance.

Acknowledgments

A.R. Teixeira received a PhD Scholarship (SFRH/BD/28404/2006) supported by the Portuguese Foundation for Science and Technology (FCT). The work has been supported also by grants of DAAD.

Appendix: Implementation of the Cholesky decomposition

A very efficient implementation for the incomplete Cholesky decomposition algorithm exists (accessible in [30]) having as input: the training data set $\mathbf{X}$, σ of the RBF kernel and a threshold to control approximation error of the decomposition. As described in [20], the matrix $\mathbf{C}$ is formed iteratively, starting with one row up to R when the error is less than the threshold. The error ε is approximated as $\varepsilon \approx \mathrm{tr}(\mathbf{K}_s - \mathbf{K}_{rs}^T \mathbf{K}_r^{-1} \mathbf{K}_{rs})$. A similar approximation error was defined in other algorithms [28,35,27] to choose the subset of the training set. Using an RBF function, the trace is obtained as $\mathrm{tr}(\mathbf{K}) = N$ where N denotes the size of the data set, then the threshold can be computed relative to the size. In the simulations a threshold equal to $0.01N$ was considered. The outputs of the algorithm are the index of the pivoting scheme and the matrix $\mathbf{C}$. The former allow to identify the subset $\mathbf{\Phi}_R$ which will contribute to form R orthogonal basis vectors (see Eq. (14)). The main steps of the algorithm are given in Algorithm 1.

Algorithm 1. Greedy KPCA: KPCA using Cholesky decomposition.

```
% Given training data set X with size N
% With c = 1 the model is computed with centered data
otherwise not
% The model is computed with L basis vectors
% Define Parameters: σ (rbf), ε(er)
% Perform a Cholesky Decomposition [30]
% Inputs: X, σ(rbf), ε(er) = 0.01*N
% Outputs: R × N matrix and vector with N elements: C, P
% Compute the basis vectors
% centering the data? c = 1 (yes)
if c==1
  mu=sum(C,2)/N;
else
  mu=0;
end
C1=C-mu * ones(1,N);
Q=C1 * C1';
% eigenvalues on decreasing order
[E, D, dummy]=svd(Q);
% L must be less or equal to R
A=inv(C(1:R,1:R))* E(:,1:L);
Xr=X(:,P(1:R));
% The bias term of centering
b=E(:,1:L)'* mu;
% Output parameters: R × L matrix A, L × 1 vector b
% Training subset with R elements Xr
```

References

[1] M.-H. Yang, D.J. Kriegman, N. Ahuja, Detecting faces in images: a survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (1) (2002) 34–58.
[2] B. Moghaddam, Principal manifolds and probabilistic subspace for visual recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (6) (2002) 780–788.
[3] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based algorithms, IEEE Transactions on Neural Networks 12 (2) (2001) 181–202.
[4] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (1998) 1299–1319.
[5] B. Schölkopf, S. Mika, C.J. Barges, P. Knirsch, K.-R. Müller, G. Ratsch, A.J. Smola, Input space versus feature space in kernel-based methods, IEEE Transactions on Neural Networks 10 (5) (1999) 1000–1016.
[6] J. Li, X. Li, D. Tao, Kpca for semantic object extraction in images, Pattern Recognition 41 (2008) 3244–3250.
[7] C. Fowlkes, S. Belongie, F. Chung, J. Malik, Spectral grouping using the Nyström method, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2) (2004) 214–225.
[8] J. Yang, X. Gao, D. Zhang, J.-y. Yang, Kernel ICA: an alternative formulation and its application to face recognition, Pattern Recognition 38 (2005) 1784–1787.
[9] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C. The Art of Scientific Computation, Cambridge University Press, 1994.
[10] P. Cheng, W. Li, Ph. Ogunbona, Greedy approximation of kernel {PCA} by minimizing the mapping error, Digital Image Computing: Techniques and Applications 0 (2009) 303–308, doi:ieeecomputersociety.org/10.1109/DICTA.2009.57.
[11] A.J. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in: International Conference on Machine Learning, Morgan Kaufmann, San-Francisco, USA, 2000, pp. 911–918.
[12] G.H. Golub, C.F. van Loan, Matrix Computations, The John Hopkins University Press, Baltimore, 1996.
[13] I.C. Choi, C.L. Monma, D.F. Shanno, Further development of primal-dual interior point methods, ORSA Journal on Computing 2 (4) (1990) 304–311.
[14] S. Fine, K. Scheinberg, Efficient SVM training using low-rank kernel representations, Journal of Machine Learning Research 2 (2001) 243–264.
[15] C.H.T. Baker, The Numerical Treatment of Integral Equations, Clarendon Press, Oxford, UK, 1977.

[16] C.K. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: Advances in Neural Information Processing Systems, NIPS 13, MIT Press, 2001, pp. 682–688.
[17] C.K. Williams, On the extension of eigenvectors to new datapoints, in: Note, available at 〈http://homepages.inf.ed.ac.uk/ckiw/postscript/nyseig.pdf〉, 2006, pp. 1–3.
[18] B. Schölkopf, A. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, MIT Press, Cambridge, MA, 2002.
[19] H. Wang, Z. Hu, Y. Zhao, An efficient algorithm for generalized discriminant analysis using incomplete Cholesky decomposition, Pattern Recognition Letters 28 (2007) 254–259.
[20] F.R. Bach, M.I. Jordan, Kernel independent component analysis, Journal of Machine Learning Research 3 (2002) 1–48.
[21] G. Rätsch, T. Onoda, K.R. Müller, Soft margins for adaboost, Machine Learning 42 (3) (2001) 287–320.
[22] A.R. Teixeira, A.M. Tomé, E.W. Lang, Greedy KPCA in biomedical signal processing, in: International Conference on Artificial Neural Networks—ICANN 07, Lecture Notes in Computer Science, vol. 4669, Porto, Portugal, 2007, pp. 486–495.
[23] A.R. Teixeira, A.M. Tomé, E.W. Lang, Exploiting low-rank approximations of kernel matrices in denoising applications, in: IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2007), Thessaloniki, Greece, 2007.
[24] Y. Xu, D. Zhang, F. Song, J.-Y. Yang, Z. Jing, M. Li, A method for speeding up feature extraction based on KPCA, Neurocomputing 70 (4–6) (2007) 1056–1061.
[25] S.-W. Kim, B.J. Oommen, On using prototype reduction schemes to optimize kernel-based nonlinear subspace methods, Pattern Recognition 37 (2004) 227–339.
[26] D. Achlioptas, F. McSherry, B. Schölkopf, Sampling techniques for kernel methods, in: Advances in Neural Information Processing Systems, MIT Press, 2002, pp. 335–342.
[27] G. Baudat, F. Anouar, Feature vector selection and projection using kernels, Neurocomputing 55 (2003) 21–38.
[28] V. Franc, V. Hlaváč, Greedy algorithm for a training set reduction in the kernel methods, in: 10th International Conference on Computer Analysis of Images and Patterns, Springer, Groningen, Holland, 2003, pp. 426–433.
[29] G.C. Cawley, N.L. Talbot, Efficient leave-one-out cross-validation of kernel Fisher discriminant classifiers, Pattern Recognition 36 (2003) 2585–2592.
[30] F.R. Bach, Kernel independent component analysis, 2003, URL 〈http://www.di.ens.fr/fbach/kernel-ica/index.htm〉.
[31] A.R. Teixeira, A.M. Tomé, E.W. Lang, Feature extraction using linear and non-linear subspace techniques, in: C. Alippi, M. Polycarpou (Eds.), Artificial Neural Networks—ICANN 2009, vol. II, Springer-Verlag, Cyprus, 2009, pp. 115–124.
[32] K. Zhang, J.T. Kwok, Density-weighted Nyström method for computing large kernel eigensystems, Neural Computation 21 (2009) 121–146.
[33] R. Duda, P. Hart, D.G. Stork, Pattern Classification, John Wiley & Sons, 2001.
[34] S. Mika, B. Schölkopf, A. Smola, K.R. Müller, M. Scholz, G. Rätsch, Kernel PCA and de-noising in feature spaces, in: Advances in Neural Information Processing 11, MIT Press, 1999, pp. 532–536.
[35] G.C. Cawley, N.L.C. Talbot, Efficient formation of a basis in a kernel induced feature space, in: M. Verleysen (Ed.), European Symposium on Artificial Neural Networks, d-side, Bruges, Belgium, 2002, pp. 1–6.

A.R. Teixeira is PhD student of Electrical Engineering at the University of Aveiro in Signal Processing group of IEETA. Her research interests include biomedical digital signal processing and principal and independent component analysis.

A.M. Tomé is an Associate Professor of electrical engineering with the DETI/IEETA of the University of Aveiro. Her research interests include digital and statistical signal processing, independent component analysis, and blind source separation, as well as classification and pattern recognition applications.

Elmar W. Lang is an Adjunct Professor of biophysics at the University of Regensburg, where he is heading the Neuro- and Bioinformatics Group. Currently, he serves as an Associate Editor of Neurocomputing and Neural Information Processing Letters and Reviews. His current research interests include biomedical signal and image processing, independent component analysis and blind source separation, neural networks for classification and pattern recognition, and stochastic process limits in queueing applications.