Feature Extraction Using Linear and Non-linear Subspace Techniques

Ana R. Teixeira[1], Ana Maria Tomé[1], and E.W. Lang[2]

[1] DETI/IEETA, Universidade de Aveiro, 3810-193 Aveiro, Portugal
ana@ieeta.pt
[2] Institute of Biophysics, University of Regensburg 93040 Regensburg, Germany
elmar.lang@biologie.uni-regensburg.de

Abstract. This paper provides a new insight into unsupervised feature extraction techniques based on subspace models. In this work the subspace models are described exploiting the dual form of the basis vectors. In what concerns the kernel based model, a computationally less demanding model based on incomplete Cholesky decomposition is also introduced. An online benchmark data set allows the evaluation of the feature extraction methods comparing the performance of two classifiers having as input the raw data and the new representations.

1 Introduction

Finding better representations of a given set of data with more informative features is sometimes fundamental to improve the performance of a classifier. Often not all original features are appropriate, and even the number of features might be too large to conduct an efficient training. Subspace techniques can be applied as unsupervised feature generators simultaneously providing dimension reduction and more suitable representations.

Principal Component Analysis (PCA) is a subspace technique widely used in many fields like face recognition [1] and related computer vision tasks [2]. In this application a new representation of a given data set is formed by a linear combination of the original features whereby the data is projected onto orthogonal basis vectors. These projections represent new features which are non-correlated and even can be of smaller number. This model also implies that the original features are linear combinations of these projections. This assumption is a limitation if it is to model highly complex data. Kernel PCA methods are well suited in such cases to find the non-linear principal components. And in a classification task, the new representation provided by the non-linear kernel methods belongs to a new space (called feature space) where the data most probably become linearly separable [3]. The main characteristic of kernel methods is that the non-linear components (in input space) are computed via a transformation to a space of higher dimension. In this feature space, the main steps of the PCA are formulated using dot products. However, the non-linear mapping and the dot product are performed simultaneously using kernel functions [4],[3]. The parameters of these functions are the features in the input space, thus avoiding an

C. Alippi et al. (Eds.): ICANN 2009, Part II, LNCS 5769, pp. 115–124, 2009.

explicit mapping to the higher dimensional space. Kernel methods are computationally demanding whenever it is needed to manipulate huge training data sets. If the kernel (dot product) matrices are large, then their storage as well as their eigendecomposition might be prohibitive in any practical application because of memory limitations. Besides this, the basis vectors of the subspace have to be written in their dual form, i.e., as a linear combination of the training set. So, in a classification task, the training set has also to be available even during the testing phase of the classifier, i.e. , even when the parameters of the subspace model do not change.

In this work we show different strategies to perform feature extraction either in input or feature space. In input space the features are calculated by using the PCA decomposition. In feature space KPCA and greedy KPCA are applied. The latter is based on an incomplete Cholesky approach to compute the new representation of the training data set in feature space. Then, the dual form of the kernel subspace model is computed, formed by a subset of the training set which turns the model less demanding, during the application to new data. Another issue to be discussed is the influence of centering the data on the models. The proposal is to perform the centering and simultaneously maintain the new representation of the training data set non-correlated. The numerical simulations compare the performance of classifiers using kernel features, principal component features and a direct classification of the raw data using two classifiers: the nearest neighbor (NN) and linear discriminant function (RL). Furthermore, to evaluate the impact of the projective techniques, a comparative study with the best results published in [5] is presented and discussed.

2 Subspace and Classification

With subspace methods, denoising or classification is achieved by projecting the data onto basis vectors ($\mathbf{U}$), i. e. by computing products between the data vectors and basis vectors. The projections constitute the new representation of the data which can be a simple linear combination of the input data (PCA projections) or it can represent non-linear components of the data (KPCA projections). In a classification task the projections are then the input to the classifier. During the training phase the basis vectors are computed and the projections are used to adapt the parameters of classifiers. Afterwards, the performance of the classifier is evaluated with the projections of new data (test data) onto the basis vectors computed using the training set. These steps are the same either using PCA or KPCA, the differences are only concerned with the introduction of a kernel to replace the dot products.

2.1 Subspace Model

Using the dual form to describe the basis vector matrix $\mathbf{U}$, the basis vectors (columns of the matrix) are obtained as a linear combination of the training data set, either in input space or after a non-linear mapping. Considering that the mapped training data set is $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2) \ldots \phi(\mathbf{x}_N)]$, the basis vector matrix reads

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

In this form the matrices $\mathbf{V}$ and $\mathbf{D}$ are obtained by computing the eigendecomposition of the kernel matrix $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$. Usually the eigenvectors, i.e. the columns of $\mathbf{V}$, are ordered according to the value of the corresponding eigenvalues, the diagonal of matrix $\mathbf{D}$. Assuming that the eigenvalues are ordered in decreasing order,$\lambda_1 >, \lambda_2, \ldots > \lambda_L \ldots > \lambda_{last}$,the number (L) basis vectors can be chosen according to the percentage of variance of the data to be kept in the new representation. Afterwards the mapped training data set $\mathbf{\Phi}$ projected into the basis vector leads to new representation,

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

where each column (j) of $\mathbf{Z}$, of dimension L, is the representation of $\mathbf{x}_j$ in the feature space. The substitution of the kernel matrix by its eigendecomposition in previous equation leads to $\mathbf{Z} = \mathbf{D}^{1/2}\mathbf{V}^T$. Then, the new representation of the training data set is non-correlated, i.e. , $\mathbf{Z}\mathbf{Z}^T$ is a diagonal matrix. Also notice that a low-rank approximation for the kernel matrix $\mathbf{K}$ can be obtained by computing $\mathbf{Z}^T\mathbf{Z}$. If the dimension of $\mathbf{Z}$ is L, the approximation corresponds to the L leading eigenvalues and related eigenvectors. Furthermore, note that the Principal Component Analysis model and its corresponding projections can be obtained by substituting the mapped data set ($\mathbf{\Phi}$) by the raw data set $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]$ in the previous equations. Due to lack of space those descriptions are omitted and it can also be verified that the properties discussed above are also accomplished.

2.2 Basis in Input Space

The common approach to compute the matrix of basis vectors $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_L]$ is to perform the eigendecomposition of the covariance matrix (or the scatter matrix $\mathbf{S} = \mathbf{X}\mathbf{X}^T$). However, instead of computing $\mathbf{S}$, a matrix of dot products, the kernel matrix ($\mathbf{K} = \mathbf{X}^T\mathbf{X}$) can be an alternative. This strategy is often used when the dimension of the data D , as in case of face recognition applications, is larger than the size of the training set N to avoid the eigendecomposition of the scatter matrix. Taking the singular value decomposition (SVD) of the data, we can establish the relations between eigenvectors of both matrices and the non-zero eigenvalues of both matrices, that are identical [6].

2.3 Basis in Feature Space

In feature space, the dot products are evaluated by kernel functions using the data in input space. For example, with the radial basis function (RBF), the dot product between a pair of points is

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

where σ is a value to be assigned according to the range of values of the data set. Thus the matrix of dot products $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$ can be calculated easily this way. Each entry $k_{i,j}$ is the result of the dot product between a pair (i, j) of mapped examples of the training set. As referred before the drawbacks of these methods are

- the size of $\mathbf{K}$. The eigendecomposition of large matrices can be unfeasible in practical applications requiring the manipulation of large data sets.
- the dual form of the model (eqn. 1). This form in kernel methods needs that the training set must be stored, even during the test phase, to compute the projections of any new point $\phi(\mathbf{y})$ into the model. This is because the mapping is never explicitly computed but is simultaneously obtained with dot product (eqn.3)- the so called kernel trick.

In the next section a method is suggested to deal with both problems. The kernel matrix of the complete training set is not computed and the eigendecomposition is performed with matrices of smaller size ($R < N$). The description of the model is then also based on a subset of the training data set.

2.4 Basis Vectors and Cholesky Decomposition

The Cholesky decomposition is a decomposition of an $N \times N$ symmetric positive matrix into the product of a $N \times N$ triangular matrix by the transpose of the triangular matrix. The incomplete approach with symmetric pivoting leads to $R \times N$ matrix, $\mathbf{C}$, which allows to compute an approximation of the original matrix controlling the error of the approximation. In [7] and [8] an algorithm is proposed which allows the computation of the decomposition of the kernel matrix having as input the training data set $\mathbf{X}$. The outcomes of the algorithm are the indexes of the pivoting and the matrix is

$$\mathbf{C} = \begin{bmatrix} \mathbf{L} & \mathbf{L}^{-T}\mathbf{K}_{rs} \end{bmatrix} \tag{4}$$

The pivoting scheme leads to the division of the training set into two subsets which can identified also using a block notation $\mathbf{X} = [\mathbf{X}_r \ \mathbf{X}_s]$. Then, the approximation of the kernel matrix $\tilde{\mathbf{K}} = \mathbf{C}^T\mathbf{C}$ can be expressed with four blocks: the upper left block matrix $\mathbf{K}_r = \mathbf{L}^T\mathbf{L}$ has dimension $R \times R$, the upper right block matrix $\mathbf{K}_{rs}$ has dimension $R \times (N - R)$, the lower left block is $\mathbf{K}_{rs}^T$ and the lower right block matrix $\mathbf{K}_{rs}^T\mathbf{K}_r^{-1}\mathbf{K}_{rs}$ has dimension $S \times S$ where $S = N - R$. The block $\mathbf{K}_r$ is the kernel matrix of the subset $\mathbf{\Phi}_r \equiv \phi(\mathbf{X}_r)$ and $\mathbf{K}_{rs}$ corresponds to the kernel matrix between the subsets. It can be easily shown that the last block represents an approximation of the corresponding block of the original matrix which should be $\mathbf{K}_s = \mathbf{\Phi}_s^T\mathbf{\Phi}_s \equiv \phi^T(\mathbf{X}_s)\phi(\mathbf{X}_s)$ [9]. The minimization of $trace(\mathbf{K}_s - \mathbf{K}_{rs}^T\mathbf{K}^{-1}\mathbf{K}_{rs}) = trace(\mathbf{\Delta}_s)$ is used as criterion to the pivoting scheme of the incomplete Cholesky algorithm [8]. The goal is to iteratively construct $\mathbf{C}$ so that $trace(\mathbf{\Delta}_s) < \varepsilon$, is an user defined threshold. The pivoting is the choice of the index of the elements in $\mathbf{X}_s$ that is moved to $\mathbf{X}_r$ in each iteration [6],[9].

The matrix $\mathbf{C}$ can also be used to form an orthogonal representation of the training data set in the feature space

$$\mathbf{Z} = \mathbf{V}_q^T \mathbf{C} \tag{5}$$

where $\mathbf{V}_q$ is the eigenvector matrix of the matrix $\mathbf{Q} = \mathbf{C}\mathbf{C}^T$. Note that $L \leq R$ projections can be considered by choosing L eigenvectors that correspond to the largest eigenvalues. Furthermore note the new representation is always non-correlated, i.e. $\mathbf{Z}\mathbf{Z}^T$ is diagonal whatever the value of L $(< R)$. A simple manipulation of $\mathbf{Z}$ leads to the description of the subspace model in its dual form

$$\mathbf{Z} = \mathbf{V}_q^T \mathbf{C} = \mathbf{V}_q^T \mathbf{L}^{-T} \mathbf{\Phi}_r^T \left[\mathbf{\Phi}_r \ \mathbf{\Phi}_s \right] \tag{6}$$

Consequently, the basis vector matrix can be written as

$$\mathbf{U} = \mathbf{\Phi}_r \mathbf{L}^{-1} \mathbf{V}_q \tag{7}$$

So, the dual form of the basis vector matrix is written using only a subset of the training data set thus reducing the storage and computational requirements during testing phases of the classifier. It has to be noticed that the vectors form an orthogonal basis in the feature space, i.e., $\mathbf{U}^T\mathbf{U} = \mathbf{I}$. Several algorithms [10], [11], [12] lead to similar solutions by exploiting the idea that there are samples in the training set that can be expressed as a linear combination of others.

2.5 Centering the Data

All previous deductions were conducted assuming that the data is centered. In the input space this can considered a pre-processing step that must be accomplished before computing the scatter or kernel matrix and before projecting any new data vector. So computing the mean of the training set $\mathbf{x}_{mean}$, the mean must be subtracted from every data vector whether it belongs to the training set or not.

KPCA and a complete training set: In feature space centering the mapped data is a more elaborate procedure that must performed mostly during the computation of the projections. To facilitate the exposition, let's consider a vector $\mathbf{m}$ with N elements all of which equal $1/N$, and a matrix $\mathbf{M}$ filled with N column vectors $\mathbf{m}$. Therefore to project a new data point $\phi(\mathbf{y})$ and to take into account the centered training data set, the following operations need to be integrated into the dot product

$$(\mathbf{\Phi} - \mathbf{\Phi}\mathbf{M})^T (\phi(\mathbf{y}) - \mathbf{\Phi}\mathbf{m}) \tag{8}$$

The first term removes the mean to the training data set, the second subtracts the mean of the training set from the new data. Then the manipulation of the previous expression results into four terms that contribute to the L projections in the feature space of the input data point y as

$$\mathbf{z}_y = \mathbf{D}^{-1/2} \mathbf{V}^T (\mathbf{\Phi}^T \phi(\mathbf{y}) - \mathbf{M}^T \mathbf{\Phi}^T \phi(\mathbf{y}) - \mathbf{\Phi}^T \mathbf{\Phi}\mathbf{m} + \mathbf{M}^T \mathbf{\Phi}^T \mathbf{\Phi}\mathbf{m}) \tag{9}$$

The last two terms only depend on the training set and they are present in every data point projected into $\mathbf{U}$, so they can be stored in advance and constitute a bias term that is present in every projection. It can be easily shown that projecting the complete training set $\mathbf{\Phi}$ to obtain $\mathbf{Z}$, the last three terms within parenthesis arise from the centered kernel matrix $\mathbf{K}_c = (\mathbf{I} - \mathbf{M})\mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{I} - \mathbf{M})$, where $\mathbf{I}$ is an $N \times N$ identity matrix. Then, to accomplish non-correlated projections for the training data set the matrices $\mathbf{V}$ and $\mathbf{D}$ should be obtained from the eigendecomposition of $\mathbf{K}_c$. It should also be noticed that with an RBF kernel the dot products in feature space are always less than the unit (see eqn. 3). And in particular the contribution of the last two terms depends on the parameter σ of the kernel function.

KPCA and a reduced training set: The starting point of a Cholesky approach is the incomplete Cholesky decomposition of the full matrix. The projections can be written (see eqn.5), in order to turn the projections related to the centered data, the low rank approximation of the kernel matrix can be centered $\tilde{\mathbf{K}_c} = (\mathbf{I} - \mathbf{M})\mathbf{C}^T\mathbf{C}(\mathbf{I} - \mathbf{M})$ where the mean $\mathbf{Cm}$ is subtracted from every column of $\mathbf{C}$. In that case the eigenvectors $\mathbf{V}_q$ must be computed with $\mathbf{Q}$ after centering the matrix $\mathbf{C}$. Then, the term $\mathbf{b} = \mathbf{V}_q^T\mathbf{Cm}$ should also be subtracted from every data projected onto the model (see eqn.7).

$$\mathbf{z}_y = \mathbf{U}^T\phi(\mathbf{y}) - \mathbf{b} \tag{10}$$

3 Numerical Simulations

The effectiveness of the subspace feature extraction methods discussed above is evaluated by comparing the performance of the classifiers. For that we carried out experiments on thirteen artificial and real world data sets available on Gunnar Ratsch's web site (accessible at http://ida.first.fraunhofer.de/projects/bench). The data sets represent benchmarks and several algorithms [5], [13], [14], in wich it has been been applied to these data sets. In this work generalization error rates of the classifiers are presented using as input: the raw data, the PCA, the KPCA, and greedy KPCA projections.

Data sets. Table 1 resumes the information of the 13 data sets. All data sets have 100 random partitions of pairs training/test sets, except *Splice* and *Image* which have 20 partitions. On each partition data sets, different classification algorithms were used, the second column shows the average and the standard deviation of the generalization error published in [5]. Furthermore, it is possible to download from the web page the generalization errors for every partition of data.

Evaluation. The performance of classifiers is used to illustrate the influence of projecting the data into the different models. Two classifiers were considered: one-nearest neighbor (NN) and the linear discriminant function (RL). With an NN classifier, each element of the test set is classified according to the nearest

Table 1. Data set description. The results of t-test (95%): Best results of [5] versus raw data classification (column I1) and Raw data versus PCA projections (column I2), where ⊕ accepts $H0$ and ⊖ rejects $H0$.

		Data description			Projections PCA				t-test	
		D	N	**Bench [5]**	**L**	**NN**	**L**	**RL**	I1	I2
Group 1	***B. Cancer (BC)***	9	200	25.9 ± 4.6	9	32.5 ± 4.8	2	26.2 ± 2.3	⊕	⊕
	Diabetis (Di)	8	468	23.5 ± 1.7	8	30.1 ± 2.0	8	23.4 ± 1.7	⊕	⊕
	German (GR)	20	700	23.6 ± 2.1	20	29.4 ± 2.4	17	23.9 ± 2.1	⊕	⊕
	Heart (Hr)	13	170	16.0 ± 3.3	9	22.0 ± 3.1	10	15.9 ± 3.1	⊕	⊕
	F. Solar (FS)	9	400	32.4 ± 1.8	9	39.0 ± 4.9	6	32.9 ± 1.8	⊕	⊖
	Thyroid (Ty)	5	140	4.4 ± 2.2	3	3.9 ± 2.1	5	14.7 ± 3.2	⊖	⊕
	Titanic (Ti)	3	150	22.4 ± 1.0	1	33.0 ± 11.0	3	22.6 ± 1.0	⊕	⊕
	Twonorm (Tn)	20	400	2.7 ± 0.2	1	3.4 ± 0.5	1	2.3 ± 0.1	⊖	⊖
Group 2	***Image (Im)***	18	1010	2.7 ± 0.7	13	3.30 ± 0.5	18	16.5 ± 0.98	⊖	⊕
	Ringnorm (Rg)	20	400	1.6 ± 0.1	6	21.3 ± 1.2	19	24.6 ± 0.7	⊖	⊖
	Splice (Sp)	60	1000	9.5 ± 0.7	6	22.4 ± 1.4	60	16.31 ± 0.6	⊖	⊕
	Waveform(Wv)	21	400	9.8 ± 0.8	2	11.7 ± 0.7	2	12.6 ± 0.7	⊖	⊖
	Banana (Ba)	2	400	10.7 ± 0.4	2	13.6 ± 7.0	1	46.9 ± 7.0	⊖	⊕

neighbor in the training set. In case of linear discriminant functions, the weight vectors are computed within the training data set by using the mean-square-error criterion [15]. In spite of having two classes, the multi-class strategy is followed and each element in the test set is assigned to the class whose discriminant function has the largest value. The averages (and standard deviation) of the generalization error rates are presented. The difference between error percentages achieved by pairs of methods are also compared using t-test with 95% significance. The statistical test has been carried out in an attempt to reject ⊖ or accept ⊕ the null hypothesis ($H0$), i.e, the equality of mean performance of both methods.

3.1 Linear Features

Both classifiers were applied to the raw data. The results of the classification leads us to organize the data sets into two groups (see table 2). In the first group (group 1), at least one of the classifiers achieves an error rate comparable to the ones published in [5]. In the remaining five data sets (the group 2) the performance was far from the results presented in [5]. A t-test is used to compare the best result with the ones published, and the $H0$ is rejected in the group 2 and it is accepted in group 1 except for two data sets (see column I1 of table 1). The data is pre-processed using PCA and its L projections are used as input to the classifiers. The number of projections was varied from 1 to D (the number of features of raw data). In table 1, for each data set the average minimum error rate and the corresponding dimension (L) of the new data representation are shown. Globally we verified that the error rate of the linear discriminant classifier is similar to the one achieved with the raw data. Using a t-test to compare the best results of raw data versus PCA projections, the $H0$ hypothesis is accepted for most of the data sets, except in four data sets (see column I2 of table 1). However, also to be noticed is that in some data sets a considerable dimension reduction is reached. The most significant occurs in the *twonorm* data set where $D = 20$ and $L = 1$ for both classifiers and in both cases the performance improves when compared with the raw data version. In the case of the RL classifier the result is even better than the result published in [5]. In this case $H0$ hypothesis is rejected (see column I1 and I2 of table 1).

3.2 Non-linear Features

Both versions of KPCA are used to compute the model: the first depends on the full training set (KPCA) and the second depends on a subset with R elements (greedy KPCA). In both cases the number of projections varied from $L = 1$ up to R. Both models are computed using centered versions of the kernel matrices as proposed before. Using the RBF kernel function to evaluate the dot products, a value must be assigned to σ. This parameter is often a variable of the experimental studies [16] or it is optimized using a cross validation strategy using the training data [13]. In this study, the value of σ^2 is chosen as the average of euclidian squared distances of each training vector to the center of training set. Notice that the choice of sigma also interferes with the size R of the subset to be included in the basis vector model of greedy KPCA. If the decay of eigenvalues is too smooth the complete training set will be chosen in the incomplete Cholesky decomposition using a fixed threshold for the approximation error. The threshold $0.01N$ was considered, because with the RBF kernel the trace of the kernel matrix is always the size of training data set.

In table 2 the average error rates and standard deviation are shown. Column $I1$ shows the result of the t-test between the results of [5] and either the NN or the RL classifier . And we can see that the H0 hypothesis is accepted in all but the German data set. We can see that in group 1 there is no significant improvement in the performance of the classifiers using the non-linear features of the data sets. In most cases the minimum error rate is achieved using a number ($L > D$) of projections higher than the dimension (D) of the raw data. One of the exceptions is the *twonorm* where $L = 1$ as in input space. Another t-test was performed to see how the number of projections can affect the results. The error rates were compared to the error rates achieved when the number of projections is $L = D$, KPCA_D. The column $I2$ of table 2 shows the result and it can be verified that the $H0$ hypothesis is rejected in group 2 and accepted in group 1. Furthermore, notice that with the linear discriminant function the best performance is achieved with $L > D$ with the exception of the *waveform* set.

Table 2 also shows the performance of classifiers using the greedy KPCA to compute projections and we can verify that the results are similar to the ones computed with KPCA. The null hypothesis is accepted for every data set (see column I3). The number of projections used in both methods was not always the same, but considering that the second method is obtained using an approximation of the kernel matrix, some variations had to be expected. In what concerns

Table 2. Error rate (%) using KPCA and greedy KPCA. Results of t-test: Best versus KPCA (column I1), KPCA versus KPCA$_D$ (column I2) and greedy KPCA versus KPCA (column I3) where ⊕ accept $H0$ and ⊖ reject $H0$.

	KPCA				$H0$		**greedy KPCA**					$H0$
	L	**NN**	L	**RL**	I1	I2	R	L	**NN**	L	**RL**	I3
BC	7	32.5 ± 4.8	21	25.2 ± 4.5	⊕	⊕	90	7	32.5 ± 4.8	22	25.2 ± 4.5	⊕
Di	17	25.3 ± 1.8	10	23.2 ± 1.6	⊕	⊕	140	61	30.2 ± 1.9	10	23.1 ± 1.6	⊕
Gr	12	30.0 ± 2.5	12	23.3 ± 2.1	⊖	⊕	400	13	29.1 ± 2.4	12	23.4 ± 2.3	⊕
Hr	8	22.7 ± 3.4	12	15.8 ± 3.0	⊕	⊕	110	48	22.79 ± 2.9	11	15.8 ± 3.1	⊕
FS	55	32.2 ± 0.5	25	32.1 ± 0.6	⊕	⊕	74	70	35.3 ± 0.7	48	33.8 ± 0.6	⊕
Ty	6	4.0 ± 2.2	15	5.8 ± 2.4	⊕	⊕	25	6	3.9 ± 2.2	25	5.3 ± 2.3	⊕
Ti	9	32.3 ± 1.1	10	22.3 ± 1.0	⊕	⊕	10	10	31.1 ± 1.4	6	21.8 ± 1.0	⊕
Tn	1	3.4 ± 0.4	1	2.3 ± 0.1	⊕	⊕	285	1	3.5 ± 0.6	1	2.3 ± 0.1	⊕
Im	23	2.8 ± 0.6	75	7.9 ± 1.3	⊕	⊖	120	21	2.9 ± 0.7	80	8.1 ± 1.2	⊕
Rg	40	3.5 ± 0.4	25	1.6 ± 0.1	⊕	⊖	262	45	3.8 ± 0.4	31	1.7 ± 0.1	⊕
Sp	600	7.5 ± 2.6	720	4.3 ± 2.1	⊕	⊖	874	620	7.7 ± 2.6	764	4.4 ± 2.1	⊕
Wv	29	9.7 ± 0.7	2	12.0 ± 0.8	⊕	⊖	258	30	9.8 ± 0.3	2	12.0 ± 0.7	⊕
Ba	5	13.6 ± 0.4	34	10.7 ± 0.4	⊕	⊖	15	15	13.6 ± 0.7	5	10.8 ± 1.8	⊕

the approximation, we see that, using the same threshold, the relative decrease (N/R) of the number of examples to describe the model is very heterogenous, it ranges from 1.1 to 26.6, but in 7 data sets is higher than 2.

4 Concluding Remarks

In this work we introduce projective subspace techniques and cast them in a concise presentation by using the dual form for the models. Besides that an algorithm to compute the KPCA model using a subset of the training data set using a greedy approach is also presented. We further consider the centering problem and adapt the model description to remove the mean of the data. We verify that these techniques have a different impact on the performance of the classifiers. The reason is mostly related to the data characteristics. We showed that for some data sets the performance achieved using raw data is similar to the results published [5]. In these data sets a dimension reduction by PCA yields similar results. It can also be verified that for these data sets, using KPCA projections, the generalization errors rate remains roughly constant. The other group are nonlinear data sets. The performance on the high-dimensional feature space clearly improves, and is comparable to the one described in [5]. Another aspect to point out is that with KPCA projections the linear discriminant function classifier performs better than the nearest neighbor one, in 10 of the 13 data sets. Confirming that having decision functions based on an hyperplane is possible but the dimension has to increase like in the case of *banana* data sets. The numerical simulations corroborate that the greedy approach to KPCA does not harm the performance.

Acknowledgment

A.R. Teixeira received a PhD Scholarship (SFRH/BD/28404/2006) supported by the Portuguese Foundation for Science and Technology (FCT).

References

1. Yang, M.-H., Kriegman, D.J., Ahuja, N.: Detecting faces in images: a survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(1), 34–58 (2002)
2. Moghaddam, B.: Principal manifolds and probabilistic subspace for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(6), 780–788 (2002)
3. Schölkopf, B., Mika, S., Barges, C.J., Knirsch, P., Müller, K.-R., Ratsch, G., Smola, A.J.: Input space versus feature space in kernel-based methods. IEEE Transactions on Neural Networks 10(5), 1000–1016 (1999)
4. Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B.: An introduction to kernel-based algorithms. IEEE Transactions on Neural Networks 12(2), 181–202 (2001)
5. Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for adaboost. Machine Learning 42(3), 287–320 (2001)
6. Teixeira, A.R., Tomé, A.M., Lang, E.W.: 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)
7. Bach, F.R.: Kernel independent component analysis (2003), `http://www.di.ens.fr/~fbach/kernel-ica/index.htm`
8. Fine, S., Scheinberg, K.: Efficient SVM training using low-rank kernel representations. Journal of Machine Learning Research 2, 243–264 (2001)
9. Teixeira, A.R., Tomé, A.M., Lang, E.W.: Feature extraction using low-rank approximations of the kernel matrix. In: Campilho, A., Kamel, M.S. (eds.) ICIAR 2008. LNCS, vol. 5112, pp. 404–412. Springer, Heidelberg (2008)
10. Franc, V., Hlaváč, V.: Greedy algorithm for a training set reduction in the kernel methods. In: 10th International Conference on Computer Analysis of Images and Patterns, pp. 426–433. Springer, Holland (2003)
11. Cawley, G.C., Talbot, N.L.C.: Efficient formation of a basis in a kernel induced feature space. In: Verleysen, M. (ed.) European Symposium on Artificial Neural Networks, pp. 1–6. d-side, Belgium (2002)
12. Baudat, G., Anouar, F.: Feature vector selection and projection using kernels. Neurocomputing 55, 21–38 (2003)
13. Xu, Y., Zhang, D., Song, F., Yang, J.-Y., Jing, Z., Li, M.: A method for speeding up feature extraction based on kpca. Neurocomputing 70(4-6), 1056–1061 (2007)
14. Cawley, G.C., Talbot, N.L.: Efficient leave-one-out cross-validation of kernel Fisher discriminant classifiers. Pattern Recognition 36, 2585–2592 (2003)
15. Duda, R., Hart, P., Stork, D.G.: Pattern Classification. John Wiley & Sons, Chichester (2001)
16. Mika, S., Schölkopf, B., Smola, A., Müller, K.R., Scholz, M., Rätsch, G.: Kernel pca and de-noising in feature spaces. In: Advances in Neural Information Processing 11, pp. 536–542. MIT Press, Cambridge (1999)