Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

NEUROCOMPUTING

Neurocomputing 55 (2003) 135–150

www.elsevier.com/locate/neucom

# On problem-oriented kernel refining☆

E. Parrado-Hernández*, J. Arenas-García, I. Mora-Jiménez, A. Navia-Vázquez

*Department of Signal Theory and Communications, University Carlos III de Madrid, Avda Universidad 30, Leganés, Madrid 28911, Spain*

Received 8 March 2002; accepted 8 January 2003

**Abstract**

Much attention has been recently devoted to those machine learning procedures known as kernel methods, the Support Vector Machines being an instance of them. Their performance heavily depends on the particular 'distance measurement' between patterns, function also known as 'kernel', which represents a dot product in a projection space. Although some attempts are being made to 'a priori' decide which kernel function is more suitable for a problem, no definite solution for this task has been found yet, since choosing the best kernel very often reduces to a selection among different possibilities by a cross-validation process. In this paper, we propose a method for solving classification problems relying on the *ad hoc* determination of a kernel for every problem at hand, i.e., a problem-oriented kernel design method. We iteratively obtain a semiparametric projecting function of the input data into a space which has an appropriately low dimension to avoid both overfitting and complexity explosion of the resulting machine, but being powerful enough to solve the classification problems with good accuracy. The performance of the proposed method is illustrated using standard databases, and we further discuss its suitability for developing problem-oriented feature extraction procedures.


*Keywords:* Kernel design; Support vector machine; Compact architectures; Growing

## 1. Introduction

Kernel methods (KM) have become a powerful technique within the machine learning community, due to the possibility of building non-linear versions of classical linear

☆ This work has been partially supported by Spain CAM Grant CAM-07T/0046/2000.
* Corresponding author. Tel.: +34-91-624-8759; fax: +34-91-624-8749.
*E-mail address:* emipar@tsc.uc3m.es (E. Parrado-Hernández).

0925-2312/03/$ - see front matter 
doi:10.1016/S0925-2312(03)00432-6

algorithms in an easy and elegant way [15]. The main idea underlying KM consists in projecting the input space onto a higher-dimension space and applying the corresponding linear algorithm in that space. Support Vector Machines (SVM) [17] stand as the most popular members of KM. They implement linear classifiers in the projected space that yield nonlinear decision boundaries in the input space. SVM-based classifiers are able to determine maximal margin boundaries which are an implementation of structural risk minimization, what leads to excellent generalization capabilities. Examples of other kernel-based algorithms are Nonlinear PCA [16], and SVM Regressors [18], among several others.

An open issue in KM literature focuses on the design of kernel functions that provide the best conditions for a particular problem to be solved. Common kernels in the literature are the linear, sigmoidal, polynomial, or Gaussian ones. The latter are considered to behave well in general, since they imply a projection onto an infinite-dimension space, where any linear algorithm can then be successfully applied (for instance, any classification problem can be solved with minimal error, since enough degrees of freedom are always available); another issue is that of correct generalization, though, which is guaranteed by the hyperparameter selection and the capacity control imposed by the SVM learning method. Special attention should be paid to those kernels specifically designed to deal with a particular non-numerical-input signal, such as those proposed for text classification [9] or in genomic processing [4]. We will not cover these issues here, since the avenues opened in this case are too numerous to be approached in a general way. We will assume, therefore, that only numerical inputs are available, and propose a general method for Problem-Oriented Kernel (POKER) design in this case.

The commonly used kernel functions (Gaussian and polynomial) tend to project data onto very high-dimension spaces, such that a tendency to overfitting exists. SVMs tend to limit this effect by controlling the capacity of the machine through the maximization of the margin and hyperparameter selection. In any case, a large number of support vectors (SVs) is usually found (every SV represents an axis in the projection space to represent new data), which does not benefit the generalization nor the complexity control of the machine. If we are mainly interested in solving a decision problem, we may ask the projection function to build the minimal representation of the input patterns in the projection space, by focusing from the beginning on the relevant data to solve the problem. Since the boundary is not known beforehand, an incremental procedure is mandatory. In this sense, the aim of this paper is to explore a method for constructing a POKER that better fits the characteristics of the problem at hand, and that also provides a more compact solution for the final classifier.

The outline of the paper is as follows: Section 2 introduces the formulation of the training algorithm for support vector classifiers (SVCs) with semiparametric kernels and Section 3 presents an iterative scheme to construct those POKERs leading to compact machines. Experimental results in several benchmark problems (both synthetic and real) are shown in Section 4 and finally, Section 5 collects the main conclusions of this paper and identifies lines for future research.

## 2. Building SVCs with semiparametric kernels

We will consider here a binary classification problem in which we have a training dataset $\{\vec{x}_1,\ldots,\vec{x}_l\}$ belonging to an input space $\mathscr{R}^n$ and the corresponding targets $y_i \in \{-1,+1\}, i=1,\ldots,l$. Nonlinear SVMs project input data onto a higher-dimension space $\mathscr{F}$, by means of a function $\vec{\phi}(\vec{x})$ and solve the problem in $\mathscr{F}$ with a linear classifier

$$\hat{y} = \text{sign}(f(\vec{x})) = \text{sign}\,(\vec{w}^{\mathrm{T}}\vec{\phi}(\vec{x}) + b), \tag{1}$$

where $\vec{w}$ is itself a linear combination of certain $\vec{\phi}(\vec{x}_i)$ (those with $\alpha_i > 0$ are called SV in SVM terminology):

$$\vec{w} = \sum_{i=1}^{l} \alpha_i y_i \vec{\phi}(\vec{x}_i). \tag{2}$$

Since the equations to solve the linear classifier only involve inner products in $\mathscr{F}$, there is no need to explicitly compute $\vec{\phi}(\vec{x})$, providing we can calculate the inner product $\langle\vec{\phi}(\vec{x}_i),\vec{\phi}(\vec{x}_j)\rangle$ directly from $\vec{x}_i$ and $\vec{x}_j$ [3]. Mercer's theorem [17] identifies certain functions or kernels associated to those projections, whose evaluation on a pair of vectors $\vec{x}_i, \vec{x}_j$ equals the inner product in $\mathscr{F}$. This way, the kernel function is defined as

$$\mathrm{k}(\vec{x}_i,\vec{x}_j) = \langle\vec{\phi}(\vec{x}_i),\vec{\phi}(\vec{x}_j)\rangle. \tag{3}$$

By means of this kernel trick we can even consider infinite dimension spaces $\mathscr{F}$ as long as we know the corresponding kernel function (this is the case of the widely used Gaussian kernel). In consequence, suitable projection design has been converted into suitable kernel design. Under SVM formulation (see by instance [3,15,17]) the classifier is expressed according to

$$\hat{y} = \text{sign}(f(\vec{x})) = \text{sign}\left(\sum_{i=1}^{NSV} \alpha_i y_i \mathrm{k}(\vec{x},\vec{x}_i) + b\right). \tag{4}$$

The above summatory incorporates only the kernel terms belonging to some of the training patterns (the SVs), but usually this number ($NSV$) is very large (in some problems even 20–40% of the database). Another inconvenience of this approximation is that we cannot analyze vectors directly in $\mathscr{F}$, what could be interesting in other tasks different from that of classification (such as feature selection). Some works [9,10,12,15] have tried to get more compact solutions (i.e., with less kernel evaluations per pattern) by assuming the following expansion of the weight vector:

$$\vec{w} = \sum_{i=1}^{R} \beta_i \vec{\phi}(\vec{x}_i). \tag{5}$$

Doing so, the classifier evaluation involves the calculation of just $R$ kernel functions for each pattern (usually $R \ll NSV$).

$$f(\vec{x}) = \sum_{i=1}^{R} \beta_i \mathrm{k}(\vec{x},\vec{x}_i) + b. \tag{6}$$

In [11,12], the algorithm for Growing Support Vector Classifiers (GSVC) is introduced as a method to iteratively construct a semiparametric approximation to a nonlinear SVC using the standard SVM formulation. As it will be explained in Section 3, GSVC combines a heuristic to select the columns that are added to the reduced kernel matrix with the algorithm WLS-SVC (Weighted Least Squares SVC) [10] to determine the coefficients of the classifier built at each iteration. Other works, such as [9,15], present methods that first approximate the kernel matrix by a reduced number of columns in a sense that some matrix norm is minimized and then solve a simpler optimization problem. The work in [7] proposes to first consider a subset of the columns of the kernel matrix at random, and then, regularize by minimizing the norm of the vector formed with the subset of the Lagrange multipliers (dual variables) corresponding to the selected columns, besides some other modifications in the original SVM formulation.

However, the above-mentioned methods (with the exception of [7]) maintain an infinite-dimensional $\mathscr{F}$. In particular, GSVC's projection and kernel functions have not changed at all and, by using this formulation, the goal is to find the $\beta_i$ that best approximates the exact SVM solution in $\mathscr{F}$; in fact, these coefficients can be exactly calculated by first solving the complete SVM problem and then solving a pseudoinverse problem (as shown in [15]). Since the projection space has not changed in this new approach, this method still makes opportunities difficult for a further data analysis in $\mathscr{F}$.

In this paper, we follow a different approach and focus on designing a mapping function $\vec{\phi}(\vec{x})$ of finite dimension which allows both to control machine complexity and to operate in the projection space, while still achieving a good solution for the classification problem in terms of maximal margin. To do so, we propose to reinterpret Eq. (6) by introducing the following projection function that maps data onto a different space $\mathscr{F}_{\mathrm{PO}}$ (problem-oriented, PO): [1]

$$\vec{\phi}_{\mathrm{PO}}(\vec{x}) = [\mathrm{k}(\vec{x},\vec{x}_1),\ldots,\mathrm{k}(\vec{x},\vec{x}_R)]^{\mathrm{T}},$$

$$\mathrm{k}_{\mathrm{PO}}(\vec{x},\vec{z}) = \sum_{i=1}^{R} \mathrm{k}(\vec{x},\vec{x}_i)\mathrm{k}(\vec{z},\vec{x}_i), \tag{7}$$

where $\vec{\phi}_{\mathrm{PO}}(\vec{x})$ is a projection function that incorporates some information about the projection of data in space $\mathscr{F}$ (in fact, $\mathscr{F}_{\mathrm{PO}}$ as defined in Eq. (7) is a subspace of $\mathscr{F}$), but it also opens the possibility of grouping together information obtained from different spaces $\mathscr{F}_k$ with different kernel functions (multikernel (MK) mapping function):

$$\vec{\phi}_{\mathrm{MK}}(\vec{x}) = [\mathrm{k}'_1(\vec{x},\vec{x}_1),\ldots,\mathrm{k}'_R(\vec{x},\vec{x}_R)]^{\mathrm{T}},$$

$$\mathrm{k}_{\mathrm{MK}}(\vec{x},\vec{z}) = \sum_{i=1}^{R} \mathrm{k}'_i(\vec{x},\vec{x}_i)\mathrm{k}'_i(\vec{z},\vec{x}_i). \tag{8}$$

[1] Note that the components of $\vec{\phi}_{\mathrm{PO}}(\vec{x})$ are calculated with the kernel function corresponding to $\mathscr{F}$.

The above definition of $\vec{\phi}_{\mathrm{PO}}(\vec{x})$ and $\mathrm{k}_{\mathrm{PO}}(\vec{x},\vec{y})$ enables us to obtain the exact SVM solution in $\mathscr{F}_{\mathrm{PO}}$ (instead of the approximate one in $\mathscr{F}$ given by Eq. (6)). Moreover, since $\mathscr{F}_{\mathrm{PO}}$ is a finite (and relatively low) dimension space, we can also analyze patterns projected in $\mathscr{F}_{\mathrm{PO}}$, what would open up a promising line for feature selection algorithms from SVM classifiers as we discuss in Section 4.

The remaining of this section is devoted to reviewing the formulation of the parametric kernel SVC so that it can be solved in a computationally efficient fashion with WLS-SVC [10], that also enables the application of growing schemes for the construction of a PO architecture (see Section 3).

In what follows, we review the formulation of the parametric kernel SVC so that the exact linear SVC in $\mathscr{F}_{\mathrm{PO}}$ is solved with WLS-SVC [10]. Afterwards, in Section 3, we describe a growing scheme for the construction of the POKER. Before, we briefly comment that other algorithms apart from WLS-SVC could be applied to determine the linear SVC in the projected space $\mathscr{F}_{\mathrm{PO}}$. We have opted for using WLS-SVC because it is computationally efficient, especially when combined with incremental algorithms for the construction of the architecture of the classifier (such as the one used in this work). This is due to the fact that WLS-SVC can take advantage of the classifier built in the previous iteration to construct the classifier for the present iteration. Furthermore, WLS-SVC does not involve any change or modification in the formulation of the minimization problem of the standard SVM.

Let us now consider a linear classification function in $\mathscr{F}_{\mathrm{PO}}$ [2]

$$\hat{y} = \mathrm{sign}(f(\vec{x})) = \mathrm{sign}(\vec{w}^{\mathrm{T}}\vec{\phi}_{\mathrm{PO}}(\vec{x}) + b), \tag{9}$$

where $\vec{w}$ and $b$ result from the minimization of

$$L_p = \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{l}\xi_i \tag{10}$$

subject to constraints

$$\left.\begin{aligned} y_i\vec{w}^{\mathrm{T}}(\vec{\phi}_{\mathrm{PO}}(\vec{x}_i) + b) &\geqslant 1 - \xi_i \\ \xi_i &\geqslant 0 \end{aligned}\right\} \quad \forall i = 1,\ldots,l \tag{11}$$

with respect to $\vec{w}$, $b$ and $\xi_i$. After the incorporation of constraints into the functional with Lagrangian multipliers $\alpha_i \geqslant 0$ and $\eta_i \geqslant 0$, respectively, and by introducing the variables

$$e_i = y_i - (\vec{\phi}_{\mathrm{PO}}(\vec{x}_i)\vec{w} + b), \tag{12}$$

$$a_i = \frac{2\alpha_i}{1 - y_i\vec{w}^{\mathrm{T}}(\vec{\phi}_{\mathrm{PO}}(\vec{x}_i) + b)} = \frac{2\alpha_i y_i}{e_i}, \tag{13}$$

[2] Note that in this case all vectors $\vec{\phi}_{\mathrm{PO}}(\vec{x}_i)$ and $\vec{w}$ can be explicitly calculated.

we can rewrite (10) as

$$L_p = \frac{1}{2}\|\vec{w}\|^2 + \frac{1}{2}\sum_{i=1}^{l} a_i e_i^2 + \sum_{i=1}^{l}(C - \alpha_i - \eta_i)\xi_i \tag{14}$$

which, apart from the last term that vanishes at the minimum, is a WLS functional. Taking derivatives with respect to $\vec{w}$, and $b$, and forcing them to be 0 at the solution, we get the following matrix expressions:

$$\begin{aligned} &\vec{w} + \mathbf{\Phi}\mathbf{D}_a\mathbf{\Phi}^{\mathrm{T}}\vec{w} + \mathbf{\Phi}\vec{a}b = \mathbf{\Phi}\mathbf{D}_a\vec{y}, \\ &\vec{a}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}\vec{w} + \vec{a}^{\mathrm{T}}b = \vec{a}^{\mathrm{T}}y, \end{aligned} \tag{15}$$

where

$$\mathbf{\Phi} = [\vec{\phi}_{\mathrm{PO}}(\vec{x}_1), \ldots, \vec{\phi}_{\mathrm{PO}}(\vec{x}_l)], \quad \vec{a} = [a_1, \ldots, a_l]^{\mathrm{T}}, \quad (\mathbf{D}_a)_{ij} = a_{ij}\delta(i-j).$$

We can now group both equations into a single system of linear equations

$$\begin{bmatrix} (\mathbf{\Phi}\mathbf{D}_a\mathbf{\Phi}^{\mathrm{T}} + \mathbf{I}) & \mathbf{\Phi}\vec{a} \\ \vec{a}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}} & \vec{a}^{\mathrm{T}}\vec{1} \end{bmatrix} \begin{bmatrix} \vec{w} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}\mathbf{D}_a\vec{y} \\ \vec{a}^{\mathrm{T}}\vec{y} \end{bmatrix}. \tag{16}$$

Note that (16) is an equation in $\vec{w}$, $b$ and $\vec{a}$ that cannot be solved simultaneously for all these variables. Thus, we use an iterative scheme consisting of two steps; after randomly initializing $\vec{w}$ and $b$:

- First, we calculate the corresponding $a_i$ values[3] by using (13).
- Then, we consider the $a_i$ values as constants and use (16) to update $\vec{w}$ and $b$.

This procedure continues until the minimum of (14) is reached. This type of iterated WLS scheme has asymptotic convergence as shown in [13]. The proof is based on the "Convergence of Block Coordinate Descent" theorem [1].

## 3. Growing scheme for designing a POKER

In this section, we present an iterative scheme for constructing a POKER that incorporates the knowledge about the decision boundary that has been learned by the classifier in the previous iterations. This growing algorithm, named GSVC, has already been proposed in [11,12] to construct compact semiparametric approximations to SVCs. The main idea underlying GSVC is to start with a very reduced machine (a few centroids for each class) and to iteratively add new centroids in order to achieve better

[3] Computation of $\alpha_i$ is straightforward, please refer to [10] for details.

representations of the decision boundary. At each iteration, the partial classification boundary is determined through the solution of a WLS-SVC problem [10]. Therefore, the classifier is initialized with $M/2$ (typically $M = 2$ or 4) centroids from each class, picked up at random from the input data ($\vec{c}_1 \cdots \vec{c}_M$). Then, the initial projection is computed as

$$\vec{\phi}_{\mathrm{PO}}(\vec{x}) = [\mathrm{k}(\vec{x}, \vec{c}_1), \ldots, \mathrm{k}(\vec{x}, \vec{c}_M)]^{\mathrm{T}} \tag{17}$$

and after the application of WLS-SVC, the first classifier is given by

$$\hat{f}(\vec{x}) = \mathrm{sign}\left( \sum_{k=1}^{M} w_k \mathrm{k}(\vec{x}, \vec{c}_k) \right). \tag{18}$$

Afterwards, the decision boundary is improved by increasing the complexity of the projection $\vec{\phi}_{\mathrm{PO}}$. For this goal, some new centroids are incorporated to the projection function. These new centroids are selected using the heuristic introduced in [11]. According to SVM principles, the SVs are the critical input data that determine the boundary, i.e. all the information needed to construct the decision boundary that minimizes the structural risk is contained in these samples. WLS-SVC has solved the optimal SVM classifier in $\mathscr{F}_{\mathrm{PO}}$, so this result can be used to identify the data points that support the initial boundary. Therefore, the heuristic to select the new centroids consists in choosing them from the current SVs. In addition, to obtain good projections, orthogonality is largely preserved by selecting as new centroids samples lying far apart from those which have already been selected. To proceed this way, the selection is refined by sorting the candidates according to their distance to the nearest centroids, and forming the final pool of candidates with those lying farther from the previous centroids. This procedure incurs in no additional cost since the value of the already computed components of $\vec{\phi}_{\mathrm{PO}}$ can be used as a measure of distance for this maximal orthogonality purpose (the $i$-th component of $\vec{\phi}_{\mathrm{PO}}(\vec{x})$ is considered to be the distance between $\vec{x}$ and the $i$-th centroid). Finally, the new $N$ centroids are selected at random from the final candidates. After this update, projection $\vec{\phi}_{\mathrm{PO}}$ is enhanced by centroids $\vec{c}_{M+1}$ to $\vec{c}_{M+N}$ (with $N$ not necessarily equal to $M$) and given by

$$\begin{aligned} &\vec{\phi}_{\mathrm{PO}}(\vec{x})^{\mathrm{new}} = [\vec{\phi}_{\mathrm{PO}}^{\mathrm{T}}(\vec{x}), \vec{\phi}_{\mathrm{PO}}^{\mathrm{T}}(\vec{x})^{\mathrm{inc}}]^{\mathrm{T}} \\ &\quad \text{with } \vec{\phi}_{\mathrm{PO}}(\vec{x})^{\mathrm{inc}} = [\mathrm{k}(\vec{x}, \vec{c}_{M+1}), \ldots, \mathrm{k}(\vec{x}, \vec{c}_{M+N})]^{\mathrm{T}}. \end{aligned} \tag{19}$$

The growing stage continues with the application of WLS-SVC and the whole procedure is iterated until the quality of projection $\vec{\phi}_{\mathrm{PO}}$ is observed to worsen because of overfitting. To detect this fact, a cross-validation procedure is applied after each iteration to determine whether to stop or continue incorporating centroids. This stopping criterion, carried out on a separate subset of the training patterns that are not fed into WLS-SVC, has proved to endow the machine with excellent generalization capabilities

[11]. Moreover, it is worth remarking the efficiency of POKER in terms of computational resources since there is no need to recalculate from scratch the $\mathbf{\Phi}$ matrix after each iteration. Furthermore, WLS-SVC provides a classifier of maximum margin since it actually solves an SVM problem.

## 4. Experimental work

In this section, the proposed POKER algorithm is first applied to two bidimensional datasets in order to illustrate the Problem-Oriented (PO) built kernels, as well as serving for an initial discussion on the implicit feature extraction performed by the method. Then, we present results on several real-world datasets extracted from the UCI machine learning repository [2] to evaluate its performance against other SVM technologies: classical SVM, SVM pruned with a cross-validation-driven strategy [15] (p-SVM, pruned-SVM), and a linear SVM on a subspace $\mathscr{F}_{\text{random}}$ determined by means of an a priori random selection of centroids. We call the last procedure RRSVM (randomly reduced SVM).

Through all the experimental work, Gaussian kernels have been used for all the coordinates of the PO mapping function, $\vec{\phi}_{\text{PO}}(\vec{x})$:

$$\mathrm{k}(\vec{x},\vec{c}_i) = \exp\left(-\frac{\|\vec{x}-\vec{c}_i\|^2}{2\sigma^2}\right). \tag{20}$$

### 4.1. Graphical interpretation of the 'ad hoc' built kernels

We use here two simple bidimensional datasets to graphically illustrate the kernel designed by POKER. The first one is that of separating two classes drawn in concentric circles. The results have been depicted in Fig. 1.

Notice how the obtained boundary perfectly represents the maximal margin solution (Fig. 1(a)), equivalent to the solution which could have been obtained with a second-degree polynomial kernel (known to be optimal for this particular problem). Therefore, POKER is able to semi-parametrically approximate the best kernel for the problem at hand (without 'a priori' information about the problem to be solved). A trivial feature extraction would be that of separating between 'positive' and 'negative' contributions to the decision, what results in a mapping of input patterns into a bidimensional space $(\phi_+, \phi_-)$. This way, the components of the projection that push towards the positive class are grouped in $\phi_+ = \sum_{w_i>0} w_i\vec{\phi}_i(\vec{x})$ and those pushing towards the negative are $\phi_- = \sum_{w_i<0} w_i\vec{\phi}_i(\vec{x})$. Fig. 1(b) displays $\phi_-$ in the vertical axis and $\phi_+$ in the horizontal one. The problem is linearly separable in that space, and the maximal margin is also preserved. Preliminary representation of these two extracted features can be observed in Fig. 1(c) (contour plot of $\phi_+$) and Fig. 1(d) (contour plot of $\phi_-$). These features tend to concentrate their influence area nearby the boundary, so they are appropriate to determine the decision boundary. Possibly, further processing

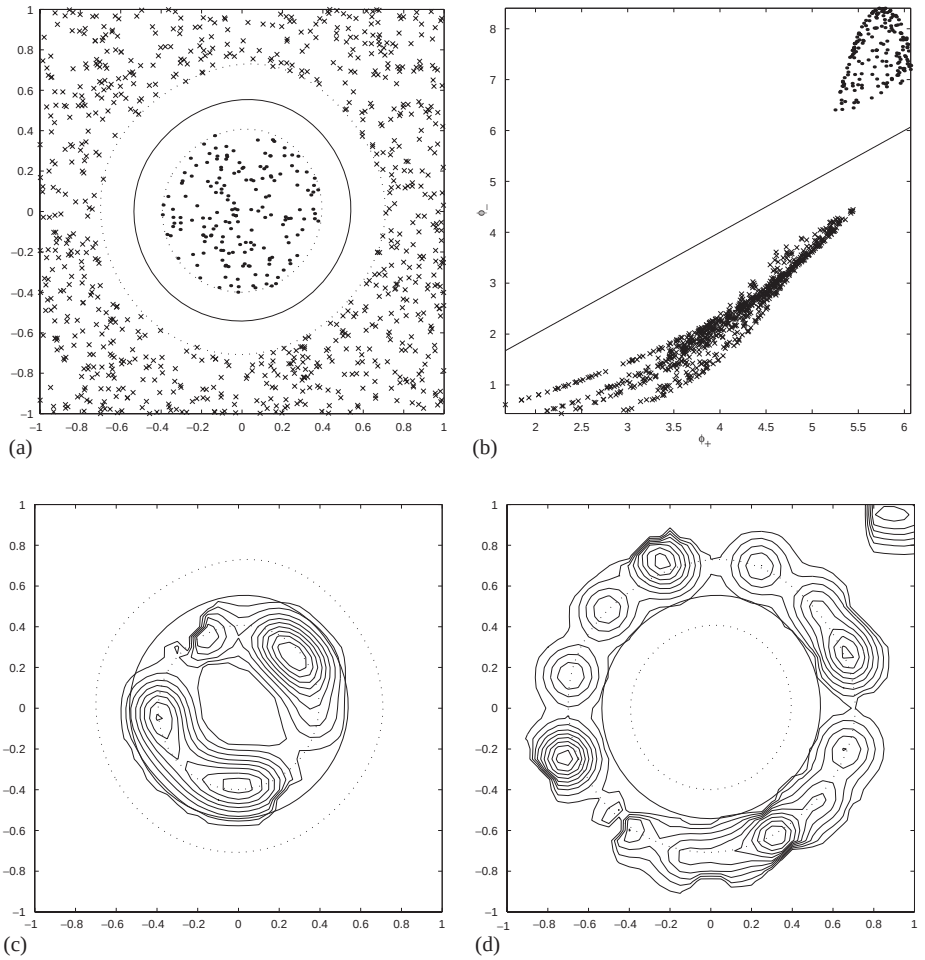


Fig. 1. Results of POKER on the concentric circle problem. The obtained boundary is shown in (a), the representation of patterns in $(\phi_+, \phi_-)$ space, where they become linearly separable in (b). Contour plots for features $\phi_+$ and $\phi_-$ in input space are displayed in (c) and (d), respectively. Plots are completed with the decision boundaries (solid) and margins (dotted) determined by POKER.

of the projected data could lead to finer grain feature analysis, but that exceeds the scope of this paper, and it is proposed as further work.

Analogous results have been represented for Kwok's dataset [6] in Fig. 2, where similar comments could be done. Note, however, that the problem is no longer separable, which is reflected in both input space (Fig. 2(a)) and projected $(\phi_+, \phi_-)$ space (Fig. 2(b)), as well as in the extended area of influence of features $\phi_+$ (Fig. 2(c)) and $\phi_-$ (Fig. 2(d)). It should be pointed out how the kernel adapts to the decision boundary, reflected by the extracted features concentrating on the decision boundary, not in the areas of few relevance for the decision.

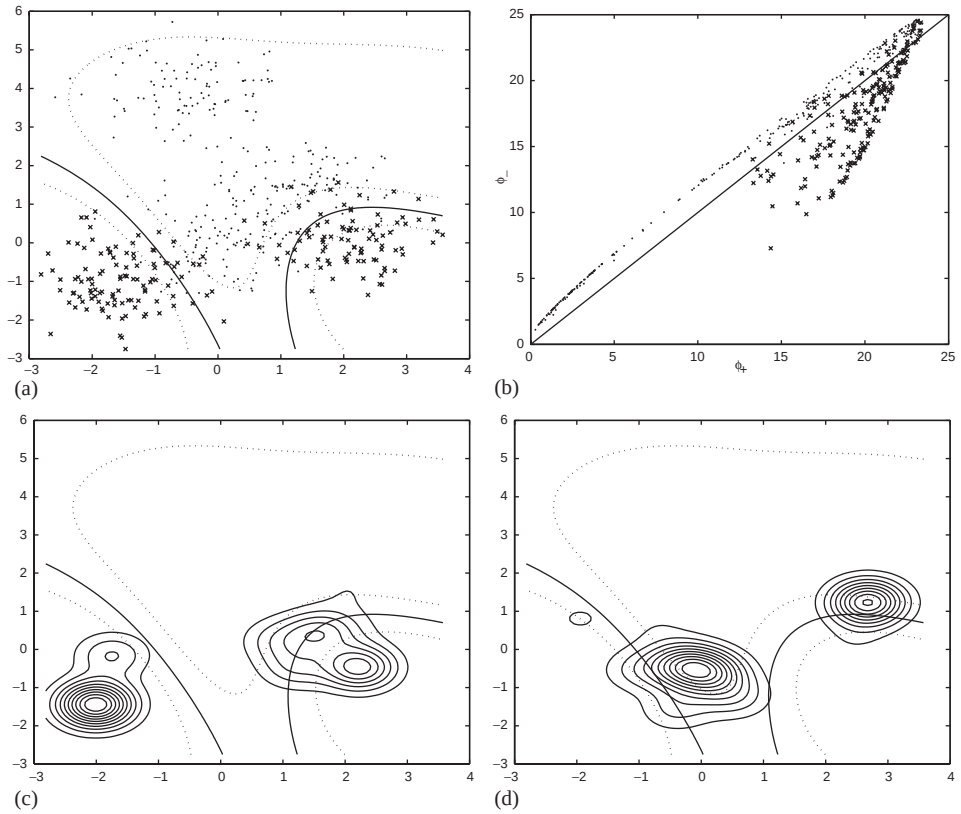


Fig. 2. Results of the POKER on Kwok's training dataset. The obtained boundary is shown in (a), the representation of patterns in $(\phi_+, \phi_-)$ space in (b), as well as the area of influence of feature $\phi_+$ (c) and $\phi_-$ (d) in input space. Contour plots for features $\phi_+$ and $\phi_-$ in input space are displayed in (c) and (d), respectively. Plots are completed with the decision boundaries (solid) and margins (dotted) determined by POKER.

### 4.2. Other experiments

We have selected for performance comparison a binary synthetic dataset *kwok* presented in [6] and several benchmark problems from the UCI Machine Learning Repository [2]: *pima*, *waveform*, *image* and *abalone*. The last dataset has been converted into a two-class problem according to [14]. *Kwok* has a two-dimensional input space; its training and test data sets are formed by 500 and 10 200 patterns, respectively. *Pima* has eight-dimensional inputs and the training and test sets sizes are 576 and 192 patterns. *Waveform* has inputs of 21 dimensions and the training and test sets sizes are 4000 and 1000 patterns. *Image* has 18-dimensional inputs and the number of training and test patterns are 1848 and 462. Finally, the partition of *abalone* resulted in a training set of 2507 patterns and a test set of 1670, with inputs of eight dimensions.

The algorithms involved in the empirical comparison with POKER are the following: first, RRSVM consists in selecting at random the kernels to be included instead of using the heuristic described in Section 3. This method resembles that of RSVM [7] and the modification proposed in [8], but we solve the exact SVM optimization problem, instead of the modified optimization problem of [7] or [8], respectively. Comparison with this method will point out how POKER effectively finds kernels that efficiently represent the problem being solved. Then, standard SVM serves as a baseline result to evaluate the classification accuracy of our method. We have used the SVM-light implementation of [5] to carry out the experimental work. The third algorithm, p-SVM consists in a standard SVM pruned until the error in a validation set reaches a minimum, following the procedure indicated in [15] for obtaining the new weights. For this purpose, the SVMs that are going to be pruned are trained with only 80% of the training patterns, reserving the remaining 20% for the validation set. Comparison with p-SVM will give an idea about POKER ability to achieve compact machines.

For all the classifiers, we have selected the value of both $\sigma$ (Gaussian kernel parameter) and $C$ (SVM regularization parameter) using a five-fold cross-validation method and exploring a range of values of $\{0.1, 0.2, 0.5, 1, 2, 5, 10\}$ for $\sigma$ and $\{1, 5, 10, 50, 100, 500, 1000, 5000, 10\,000\}$ for $C$. RRSVM needs to set a priori the number of columns of the reduced kernel matrix. According to [7], we have explored values of 1%, 5% and 10% of the total training dataset for the size of this matrix. All the results involving non-deterministic models (POKER and RRSVM) correspond to the average value and standard deviation obtained through 20 experiments.

Table 1 shows the test classification error (CE) and machine size in terms of number of nodes in the classifier obtained by POKER, SVM, p-SVM and RRSVM. Field *Machine size* refers to the number of kernel evaluations required to classify a pattern. We also display in the field *% Patterns* the percentage of training patterns that are used to determine the architecture of the classifier.

The results in Table 1 show that POKER and RRSVM achieve similar CE rates to those of SVM in all the studied problems, although they involve classifiers with a very reduced number of nodes. However, pruned SVM has only reached POKER and RRSVM good performances in both machine size and CE in *kwok*. In addition to this, neither SVM nor p-SVM enable any analysis in feature space, since they use a projection $\vec{\phi}$ of infinite dimension.

In relation to POKER and RRSVM comparison, both systems achieve similar CE percentages. A deeper analysis of the results in Table 1 points out that in the problems where the machine sizes are similar (*image* and *kwok*), POKER performs slightly better than RRSVM. On the other problems (*abalone*, *pima* and *waveform*), RRSVM's CE percentages are better than POKER's, although RRSVM implies more complex architectures (between 0.5 and 3 times more nodes). These results show that, in certain problems, POKER classification rate could be slightly improved by the addition of more nodes to the architecture, although this increase in machine complexity may not be worthy. However, this fact must be interpreted in the perspective of the main objective of this paper: the design of a problem-oriented kernel that focuses in the representation of the decision boundary. Fig. 3 displays the contour plots of features $\phi_+$ and $\phi_-$ obtained by RRSVM best performance in problem *kwok*. A comparison

Table 1
Performance comparison between POKER, SVM, p-SVM and RRSVM in several benchmark problems

| Problem | | POKER | SVM | p-SVM | RRSVM |
|---|---|---|---|---|---|
| Abalone | CE | $19.5 \pm 0.3$ | 20.9 | 44.9 | $18.9 \pm 0.2$ |
| | Machine size | $41.3 \pm 16.6$ | 1248 | 569 | 126 |
| | % Patterns | 1.65 | 49.78 | 22.7 | 5.03 |
| | $(\sigma, C)$ | $(0.5, 10000)$ | $(10, 5000)$ | $(0.5, 50)$ | $(0.5, 10000)$ |
| Image | CE | $3.8 \pm 0.7$ | 3.5 | 2.8 | $4.5 \pm 0.7$ |
| | Machine size | $170 \pm 50.2$ | 209 | 397 | 185 |
| | % Patterns | 9.2 | 11.31 | 21.48 | 10.01 |
| | $(\sigma, C)$ | $(2, 500)$ | $(5, 1000)$ | $(1, 100)$ | $(2, 10000)$ |
| Kwok | CE | $12.1 \pm 0.2$ | 11.9 | 12.4 | $12.4 \pm 0.4$ |
| | Machine size | $25 \pm 6$ | 134 | 21 | 25 |
| | % Patterns | 5.00 | 26.80 | 4.20 | 5.00 |
| | $(\sigma, C)$ | $(2, 100)$ | $(2, 10)$ | $(1, 1)$ | $(0.5, 1)$ |
| Pima | CE | $23.6 \pm 2$ | 22.4 | 22.9 | $22.5 \pm 0.4$ |
| | Machine size | $19.3 \pm 6.8$ | 304 | 248 | 29 |
| | % Patterns | 3.35 | 52.78 | 6.60 | 5.03 |
| | $(\sigma, C)$ | $(1, 1000)$ | $(2, 50)$ | $(1, 5)$ | $(2, 1000)$ |
| Waveform | CE | $8.6 \pm 0.3$ | 8.4 | 22.8 | $8.4 \pm 0.5$ |
| | Machine size | $97.2 \pm 16.4$ | 1012 | 134 | 400 |
| | % Patterns | 2.43 | 25.30 | 3.35 | 10.00 |
| | $(\sigma, C)$ | $(5, 100)$ | $(5, 1)$ | $(5, 1)$ | $(2, 5)$ |

Test classification error (in percentage) and standard deviation are showed, as well as final machine sizes and the percentage of training patterns that are used to determine the architecture of the classifiers.

of Figs. 2(c) and (d) with the corresponding Figs. 3(c) and (d) shows that POKER features focus on the boundary, while RRSVM ones are spread all over the region within the margins. The reason for this behavior is that POKER classifiers are grown by expanding the kernel only with terms near the boundary (those critical to improve the definition of the boundary) and discarding the irrelevant data, what agrees with SVM philosophy of selecting the critical input data. On the other hand, RRSVM's kernel includes terms from the whole input space since it positions the centroids at random. Therefore, POKER classifier performance could be slightly improved by expanding the kernel with components that characterize other parts of the input space than the boundary, but it would mean the loss of the PO characteristic, and the closing of opportunities for a feature selection analysis oriented to the interpretability of the decision boundary.

We finish the discussion of the results by commenting on the ability of POKER to project data into a space where vector images can be explicitly calculated. We illustrate the potential feature representation and extraction capabilities with projections onto feature space $(\phi_+, \phi_-)$ for some of the datasets (Fig. 4). It seems clear that further

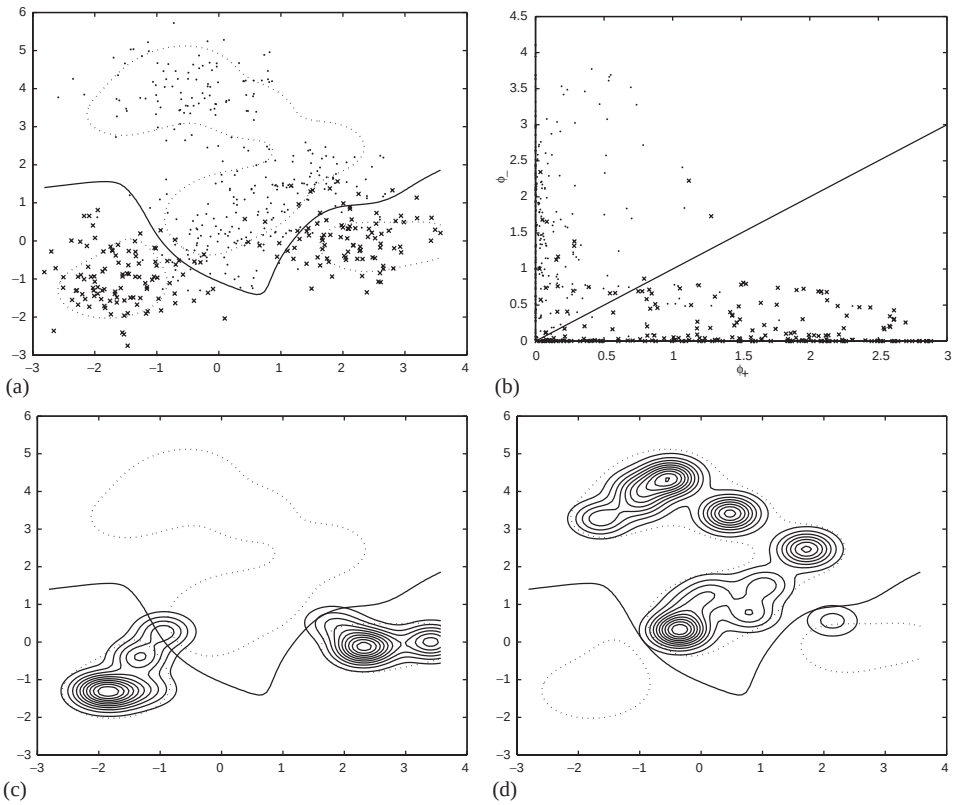


Fig. 3. Results of the RRSVM on Kwok's dataset. The obtained boundary is shown in (a), the representation of patterns in $(\phi_+, \phi_-)$ space in (b), as well as the area of influence of feature $\phi_+$ (c) and $\phi_-$ (d) in input space. Contour plots for features $\phi_+$ and $\phi_-$ in input space are displayed in (c) and (d), respectively. Plots are completed with the decision boundaries (solid) and margins (dotted) determined by RRSVM.

processing of these somewhat 'aggregated' two features could lead to more detailed features, for instance by applying Vector Quantization methods on this space or by differently aggregating kernels into more than two features. We propose this topic as future work, though.

## 5. Concluding remarks and future work

We have proposed a problem-oriented kernel extraction method, relying on a growing procedure to build SVCs. Experimental results reveal the good generalization capabilities of the method and the compactness of the resulting machines in terms of number of nodes. Furthermore, the graphical analysis of the implicit feature extraction carried out by the POKER method shows that it actually extracts features that concentrate on

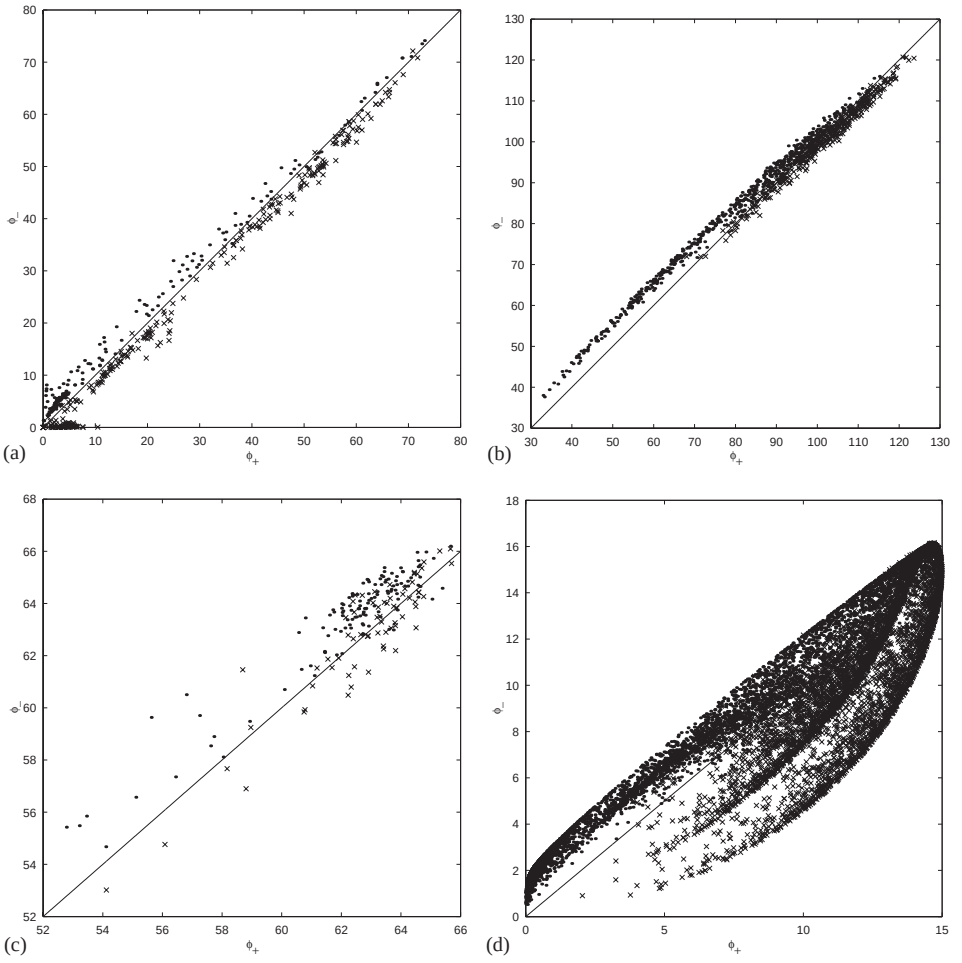


Fig. 4. Projections of input patterns onto feature space $(\phi_+, \phi_-)$ for image (a), waveform (b), pima (c) and kwok (d) test datasets. Patterns of both classes are plotted using different symbols.

the areas close to the decision boundary, or, alternatively interpreted, that the method is able to build ad hoc kernels for a given problem. Preliminary analysis of the projection onto a bidimensional feature space indicates that it would be possible to further process these features to extract a finer-grain feature representation, for instance by performing different groupings in the semiparametric kernels built by the algorithm. Furthermore, this method is directly extensible to multiclass problems: we believe that exploiting this information could lead to feature extraction methods that find common features across categories, and which may help in many cases to obtain interpretability in complex classification tasks.

## Acknowledgements

The authors would like to thank the anonymous referees for their valuable comments and references, that have helped to improve the quality of this paper.

## References

[1] D.P. Bertsekas, Nonlinear Programming, Athena Scientific, Belmont, MA, 1999.
[2] C.L. Blake, C.J. Merz, UCI Repository of machine learning databases, Department of Information and Computer Science, University of California, Irvine, CA, 1998. [http://www.ics.uci.edu/∼mlearn/MLRepository.html].
[3] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining Knowledge Discovery 2 (1998) 121–167.
[4] T. Jaakkola, M. Diekhans, D. Haussler, Using the Fisher kernel method to detect remote protein homologies, in: Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, Menlo Park, CA, 1999.
[5] T. Joachims, Making large-scale SVM learning practical, in: B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 169–184.
[6] J.T. Kwok, Moderating the output of support vector machine classifiers, IEEE Trans. Neural Networks 10 (5) (1999) 1018–1031.
[7] Y.-J. Lee, O.L. Mangasarian, RSVM: Reduced Support Vector Machines, in: Proceedings of the SIAM International Conference on Data Mining, Chicago, IL, 2001.
[8] K.-M. Lin, C.-J. Lin, A study on reduced support vector machines, Technical Report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, February 2002.
[9] H. Lodhi, J. Shawe-Taylor, N. Cristianini, C.J.C.H. Watkins, Text classification using string kernels, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, Vol. 13, MIT Press, Cambridge, MA, 2001, pp. 563–569.
[10] A. Navia-Vázquez, F. Pérez-Cruz, A. Artés-Rodríguez, A.R. Figueiras-Vidal, Weighted least squares training of support vector classifiers leading to compact and adaptive schemes, IEEE Trans. Neural Networks 12 (5) (2001) 1047–1059.
[11] E. Parrado-Hernández, I. Mora-Jiménez, J. Arenas-García, A.R. Figueiras-Vidal, A. Navia-Vázquez, Growing support vector classifiers with controlled complexity, Pattern Recognition 36 (2003) 1479–1488.
[12] E. Parrado-Hernández, I. Mora-Jiménez, A. Navia-Vázquez, Growing support vector classifiers via architecture boosting, in: Proceedings of the Learning'00, Madrid, Spain, 2000.
[13] F. Pérez-Cruz, P. Alarcón-Diana, A. Navia-Vázquez, A. Artés-Rodríguez, Fast training of support vector classifiers, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, Vol. 13, MIT Press, Cambridge, MA, 2001, pp. 724–740.
[14] A. Ruiz, P.E. López-de-Teruel, Nonlinear kernel-based statistical pattern analysis, IEEE Trans. Neural Networks 12 (1) (2001) 16–32.
[15] B. Schölkopf, A.J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2002.
[16] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis in a kernel eigenvalue problem, Neural Comput. 10 (1998) 1299–1319.
[17] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
[18] V. Vapnik, S. Golowich, A. Smola, Support vector method for function approximation, regression estimation, and signal processing, in: M. Mozer, M. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems, Vol. 9, MIT Press, Cambridge, MA, 1997, pp. 281–287.

**E. Parrado-Hernández** received the Telecommunication Engineer degree from Universidad de Valladolid, Spain, in 1999. At the present moment, he is a Ph.D. student in signal and data processing at the Department of Signal Theory and Communications, Universidad Carlos III de Madrid, Spain. His research interests include nonlinear processing, specially based on kernel methods, and its application to data mining and information retrieval.

**J. Arenas-García** was born in Seville, Spain, in 1977. He received the Telecommunication Engineer degree in 2000 from Universidad Politécnica de Madrid (ranked number 1; National Award to graduation). He is currently pursuing the Ph.D. degree at the Department of Signal Theory and Communications, Universidad Carlos III de Madrid. His present research interests are focused in the fields of neural networks and learning theory.

**I. Mora-Jiménez** received the engineering degree in 1998 from the Universidad Politécnica de Valencia, Valencia, Spain. Since then she is pursuing the Ph.D. degree in artificial neural networks at the Universidad Carlos III de Madrid, Madrid, Spain. Her main research interests include machine learning, data mining and artificial neural networks.

**A. Navia-Vázquez** received his Degree in Telecommunications Engineering in 1992 (Universidad de Vigo, Spain), and finished his PhD, also in Telecommunications Engineering in 1997 (Universidad Politécnica de Madrid, Spain). He is now an Associate Professor at the Department of Communication Technologies, Universidad Carlos III de Madrid, Spain. His research interests are focused on new architectures and algorithms for nonlinear processing, as well as their application to multimedia processing, communications, data mining, knowledge management and teleeducation. He has (co)authored more than 20 international journal and conference papers in these areas.