Pattern Recognition Letters 30 (2009) 1507–1515

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

IP-LSSVM: A two-step sparse classifier

B.P.R. Carvalho *, A.P. Braga

Depto. Engenharia Eletrônica, Campus da UFMG, Pampulha, 31.270-901 Belo Horizonte, MG, Brazil

ARTICLE INFO

Article history:
Received 25 February 2008
Received in revised form 5 February 2009
Available online 7 August 2009

Communicated by P. Sarkar

Keywords:
Sparse classifier
Least squares support vector machine
Support vector automatic detection

ABSTRACT

We present in this work a two-step sparse classifier called $IP-LSSVM$ which is based on Least Squares Support Vector Machine (LS-SVM). The formulation of LS-SVM aims at solving the learning problem with a system of linear equations. Although this solution is simpler, there is a loss of sparseness in the feature vectors. Many works on LS-SVM are focused on improving support vectors representation in the least squares approach, since they correspond to the only vectors that must be stored for further usage of the machine, which can also be directly used as a reduced subset that represents the initial one. The proposed classifier incorporates the advantages of either SVM and LS-SVM: automatic detection of support vectors and a solution obtained simply by the solution of systems of linear equations. $IP-LSSVM$ was compared with other sparse LS-SVM classifiers from literature, LS^2-SVM, $Pruning$, $Ada-Pinv$ and $RRS+LS-SVM$. The experiments were performed on four important benchmark databases in Machine Learning and on two artificial databases created to show visually the support vectors detected. The results show that $IP-LSSVM$ represents a viable alternative to SVMs, since both have similar features, supported by literature results and yet $IP-LSSVM$ has a simpler and more understandable formulation.

1. Introduction

The success of Support Vector Machine (SVM) (Vapnik, 1995) is mainly due to its solid formal basis and elegant approach in margin maximization and support vectors selection. Maximum margin hyperplane can be obtained thanks to the quadratic programming (QP) approach to the learning problem, while support vectors are outlined by the sensitivity of the corresponding Lagrange multipliers (Vapnik, 1995), which are non-zero in the separation margin. Nevertheless, alternatives to the quadratic programming approach, such as the Least Squares Support Vector Machine (LS-SVM) (Suykens and Vandewalle, 1999) are found in the literature. LS-SVM yields simplicity by solving the primal problem as a system of linear equations. The least squares (LS) solution is less computationally intensive than the quadratic programming one, but it also results on loss of sparseness of the Lagrange multipliers vector. Therefore, selecting LS-SVM support vectors by the non-zero criterion usually results on all training patterns being considered as support vectors, what is sometimes regarded as a drawback of the LS approach.

The importance of an optimal number of support vectors can not be neglected in a classification problem, since they represent the most relevant samples for outlining the separation boundary. Support vectors are useful for representing large static and dynamic data sets for classification purposes and can also help in problem analysis by pointing out to the most relevant cases (Tax and Duin, 1999; Ganapathiraju and Picone, 2000). As a consequence of this trade-off between sparseness and complexity, many works on LS-SVM are focused on improving support vectors representation of the LS approach (Suykens et al., 2000; Valyon and Horváth, 2004; Carvalho and Braga, 2005; Carvalho et al., 2007). The motivation behind these works are that LS-SVM may still provide a reduced set of support vectors, by simply observing the proper features from the LS solution.

SVM's constrained optimization problem is formalized in the LS-SVM approach as a least squares problem in the form $\mathbf{AX}=\mathbf{B}$, where $\mathbf{A}$ contains mainly kernel mapping information, $\mathbf{X}$ contains the optimization parameters (Lagrange multipliers α and bias b) and $\mathbf{B}$ is a vector of equality constrains. The problem of support vectors identification in this approach can be regarded as the one of solving the optimization problem with the smallest possible vector $\mathbf{X}$. This would result on a maximum margin with minimum number of support vectors. The problem is therefore on selecting rows of $\mathbf{X}$ without changing the separating hyperplane and yet maintaining the original LS-SVM formulation.

In order to avoid kernel mapping information loss due to dimensionality reduction of $\mathbf{A}$ as a consequence of eliminating rows of $\mathbf{X}$, the $IP-LSSVM$ approach presented in this paper maintains labeling information in $\mathbf{A}$ for all patterns in the data set, including those that had their corresponding rows eliminated

* Corresponding author. Fax: +55 3132416175.
E-mail addresses: bpenna@gmail.com (B.P.R. Carvalho), apbraga@cpdee.ufmg.br (A.P. Braga).

0167-8655/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2009.07.022

in **X**. The problem is solved in two steps. The first one corresponds to a feed-forward LS-SVM phase with the objective of obtaining the Lagrange multipliers. In the second one, vector elimination is followed by feed-forwarding the inputs with support vectors only. The mapping obtained in both phases should match, despite of the dimensionality reduction in the last one. In spite of the LS Lagrange multipliers vectors not being sparse, their magnitude do contain boundary information. *IP – LSSVM* takes advantage of this by selecting support vectors according to the magnitudes of their Lagrange multipliers. The new criterion, that is based on the support of two parallel hyperplanes, is more consistent with the concept of a SVM classifier and has yielded better results than those obtained with current approaches (Section 2).

Some of the most recent sparse LS-SVM classifiers found in the literature (Section 3) were evaluated on four benchmark classification databases (Blake and Merz, 1998) in the experiments of Section 5: Ionosphere, Pima Indian Diabetes, Bupa Liver Disorder and Tic Tac Toe. A high rank of similarity among the support vectors obtained with *IP – LSSVM* and those generated with QP SVMs was achieved for different real and synthetic data sets (Section 5). This suggests that the proposed approach can take advantage of least squares simplicity and still detect quadratic programming support vectors.

2. Least Squares Support Vector Machine

Given the training set $\{\mathbf{x}_i, y_i\}_{i=1}^N$ with $\mathbf{x}_i \in \Re^n$ and $y_i \in \{-1, +1\}$, the basic principle of SVMs is to map the input data into a high dimensional feature space by means of kernel functions. Kernel mapping results on a linearly separable problem in the feature space that can be solved with a hyperplane in the form $\boldsymbol{\omega}^T\varphi(\mathbf{x}) + b = 0$ where $\boldsymbol{\omega}$ is the parameter's vector, b is the bias term and $\varphi(\cdot)$ is the mapping function. Margin maximization is obtained by minimizing the squared norm of $\boldsymbol{\omega}$ while also minimizing the error of the training set. The resulting optimization problem is usually formulated within constrained optimization principles. The primal LS-SVMs expression for solving this problem is presented in Eq. (1). The slack variable e_i that appears in both the cost function and in the constrain of the equation has the function of controlling the margin width or, in other words, the distance between the separating hyperplane and the two parallel hyperplanes that encapsulate the margin. The error of the training data is optimized by

$$min_{\omega,b,\mathbf{e}}\ J_p(\boldsymbol{\omega}, b, \mathbf{e}) = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \gamma\frac{1}{2}\sum_{i=1}^{N} e_i^2 \tag{1}$$

subject to

$$y_i[\boldsymbol{\omega}^T\varphi(\mathbf{x}_i) + b] = 1 - e_i, \quad i = 1, \ldots, N$$

where γ is a margin parameter, analogous to SVM's C.

After deriving the Lagrangean of Eq. (1) in relation to its primal and dual variables, Eq. (2) is obtained.

$$\begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \alpha \sum_{i=1}^{N}\sum_{j=1}^{N}(y_i y_j \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j) + \frac{1}{\gamma}) + yb = 1 \end{cases} \tag{2}$$

Eq. (2) can be written as a linear system $\mathbf{AX} = \mathbf{B}$ where

$$\mathbf{A} = \begin{bmatrix} 0 & -\mathbf{Y}^T \\ \mathbf{Y} & \mathbf{H} \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} \tag{3}$$

$$\text{with} \quad \mathbf{H} = \mathbf{Z}\mathbf{Z}^T + \frac{\mathbf{I}}{\gamma} \quad \text{and} \quad \mathbf{Z} = \begin{bmatrix} \varphi(x_1)y_1 & \cdots & \varphi(x_1)y_1 \\ \vdots & \ddots & \vdots \\ \varphi(x_N)y_N & \cdots & \varphi(x_N)y_N \end{bmatrix} \tag{4}$$

Once the Lagrange multipliers and bias term are obtained from Eq. (3), the output of the LS-SVM can be calculated by simply applying the expression $f(\mathbf{x}) = sign\left[\sum_{i=1}^{N}\alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right]$ where $K(\mathbf{x}, \mathbf{x}_i) = \varphi(\mathbf{x}) \cdot \varphi(\mathbf{x}_i)$.

Considering that $\alpha_i = \gamma e_i$ (Eq. (2)) it is possible to assume that rarely a Lagrange multiplier α will be zero in the solution of a LS-SVM (Suykens and Vandewalle, 1999), what makes the range of values of α different from those obtained by the quadratic programming solution. This happens because γ does not impose a range constraint in α like the parameter C does in QP SVMs where $0 \leqslant \alpha \leqslant C$. Nevertheless, it will be shown in the next sections that *IP – LSSVM* is able to express support vectors that are very close to those obtained by QP SVMs.

3. Sparse methods for LS-SVM

The most relevant methods for enhancing sparseness in LS-SVM Lagrange multiplier vector are described in this section. The sparse methods presented are $LS^2 - SVM, Pruning, Ada - Pinv$ and $RRS + LS - SVM$. These methods are compared with our proposed classifier *IP – LSSVM* in the results and discussions section.

3.1. $LS^2 - SVM$

This method was proposed on Valyon and Horváth (2004), using some ideas from RSVM (Lee, 2001), such as the elimination of columns of **A** without eliminating the corresponding rows. Likewise our *IP – LSSVM* approach, this is a two-phase method that attempts to reduce **A** in order to detect the support vectors. The first phase is carried out by reducing the dimension of matrix **A** in Eq. (3) with a column elimination algorithm only (Valyon and Horváth, 2004). The objective is to perform elementary operations in matrix **A** with the aim of obtaining its echelon reduced form matrix **A**′. A threshold function is applied to **A**′ so that its elements that are smaller than a threshold $\epsilon \in \Re$ are set to zero. After obtaining the reduced matrix **A**′, the corresponding columns that have only zero elements are eliminated. Since all rows of **A**′ were maintained, while some columns were removed, the new matrix **A**′ is not square. Therefore, in the second step, the reduced linear system $\mathbf{A}'\mathbf{X} = \mathbf{B}$ becomes over-determined and the pseudo-inverse $(\mathbf{A}')^+$ needs to be calculated in order to find a solution for **X**. This method has an extra training parameter ϵ that corresponds to a numeric tolerance used by the process of reduction to the echelon form, described above.

3.2. Pruning

In this method (Suykens et al., 2000), training vectors $\mathbf{x}_i$ are eliminated according to the absolute value of their Lagrange multipliers $|\alpha_i|$. The process is accomplished recursively, with gradual vector elimination at each iteration, until a stop criterion is reached, which is usually associated with decrease in performance on a validation set. Vectors are eliminated by setting the corresponding Lagrange multipliers to zero, without any change in matrix dimensions. The resolution of the current linear system, for each new reduced set, is needed at each iteration, and the reduced set is selected from the best iteration. This is a multi-step method, since the linear system needs to be solved many times until the convergence criterion is reached.

3.3. Ada − Pinv

This is a two-phase approach (Carvalho and Braga, 2005) that takes advantage of an iterative method, the gradient descent of Adaline, in order to eliminate vectors in the first phase. This method differs from previous one by the fact that there is no need of matrix reduction to the echelon form. It is accomplished by training LS-SVM with Adaline, what is much easier to implement. The corresponding Lagrange multipliers are sorted according to their absolute values $|\alpha_i|$ and column elimination of $\mathbf{A}$ is carried out according to a threshold value of $|\alpha_i|$. This is based on the principle that $\alpha_i = \gamma e_i$ and, therefore, smaller $|\alpha_i|$ indicates correct classification and distance from the margin. Notice that only the columns of $\mathbf{A}$ are removed, what also results on an over-determined linear system. Therefore, once the column elimination is performed, the solution of the reduced system in the second phase is obtained by using the pseudoinverse in $\mathbf{X} = \mathbf{A}^{+}\mathbf{B}$.

3.4. RRS + LS − SVM

This is a hybrid sparse classifier that is based on a variation of the condensed nearest neighbor rule, called Reduced Remaining Subset (RRS), that aims at providing LS-SVM with a reduced set that is likely to match the support vectors. After that, LS-SVM uses the samples selected by RRS as support vectors in order to solve the linear system and to find the decision surface between the classes (Carvalho et al., 2007).

4. IP-LSSVM

Before describing *IP − LSSVM*, a new relevance criterion for the Lagrange multipliers will be presented in this section. It is based on the idea of using the Lagrange multiplier α_i associated to each training vector $\mathbf{x}_i$ instead of its absolute value $|\alpha_i|$ (Suykens et al., 2000). Since LS-SVMs are based on the construction of two parallel hyperplanes, the distance (and direction) of each vector $\mathbf{x}_i$ to these two support hyperplanes are given by the slack variable e_i, which is proportional to the Lagrange multiplier α_i. While $|\alpha_i|$ indicates the distance of $\mathbf{x}_i$ to the hyperplanes, α_i indicates also in which side of the hyperplane $\mathbf{x}_i$ is located (Carvalho et al., 2007). This can be best understood by observing Figs. 1 and 2.

Fig. 1 shows patterns of a binary classification problem and the corresponding values of α resulting from constrained optimization

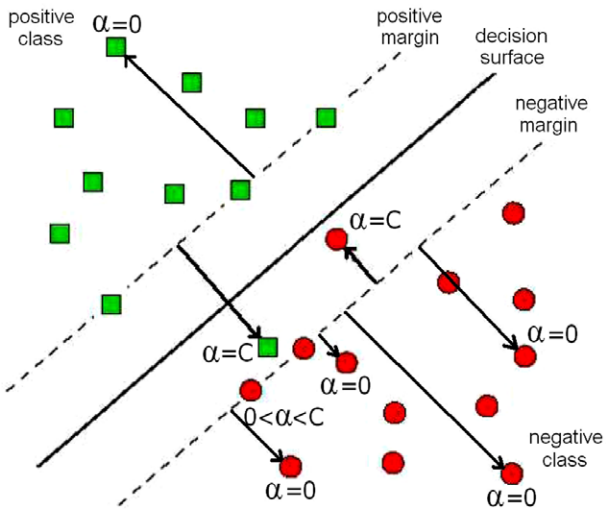

Fig. 1. Constrained values of α and their relation to the margin parameter C as a result of QP-SVM optimization.

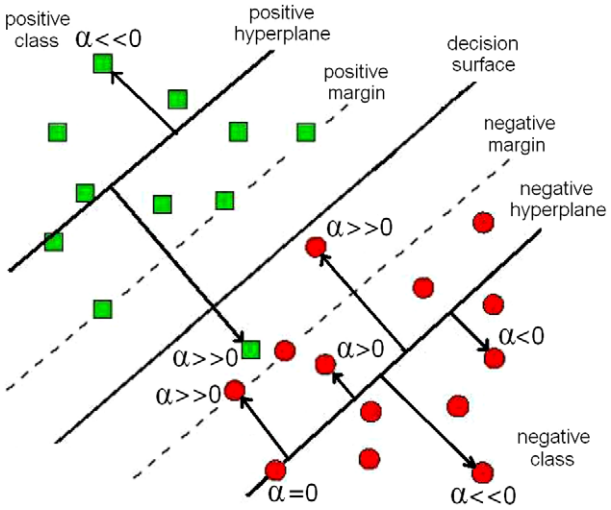

Fig. 2. Unconstrained values of α as a result of LS-SVM optimization and pattern location in relation to the separating and support hyperplanes.

QP-SVM. In this case, the error learning function is optimized by constraining α to the limits $0 \leq \alpha \leq C$ where C is the margin parameter. Support vectors are those that have $0 < \alpha < C$ as can be seen in the Fig. 1. Patterns with $\alpha = 0$ are located outside the margin hyperplanes and those with $\alpha = C$ are just between them.

A similar analysis can be made for LS-SVM by observing Fig. 2. As can be observed, patterns with $\alpha < 0$ and $\alpha \ll 0$ are analogous to those with $\alpha = 0$ in QP-SVM and, consequently, should not be considered as support vectors. The same reasoning holds for those patterns with $\alpha \geqslant 0$ since they are correctly classified patterns that are close to the support hyperplanes. Patterns that are close to the separating surface and far from the support hyperplanes correspond to $\alpha \gg 0$ and are, therefore, likely to become support vectors, as can be observed in Fig. 2. It is also important to observe that the support hyperplanes do not correspond to the margin limits in LS-SVM. In fact, since patterns with $\alpha \gg 0$ are likely to become support vectors, the margin limits are located closer to the separating surface than the support hyperplanes.

According to these arguments, the new relevance criterion proposed for LS-SVM can be described as:

- $\mathbf{x}_i$ with $\alpha_i \gg 0$ is a support vector: $\mathbf{x}_i$ is located *on the border between the two classes or on the opposite class region*, corresponding to vectors associated with non-zero α_i in QP SVM.
- $\mathbf{x}_i$ with $\alpha_i \geqslant 0$ is eliminated: $\mathbf{x}_i$ is correctly classified, *near the support hyperplane*, corresponding to vectors associated with null α_i in SVM.
- $\mathbf{x}_i$ with $\alpha_i < 0$ or $\alpha_i \ll 0$ is eliminated: $\mathbf{x}_i$ is correctly classified, *far from the decision surface*, corresponding to vectors associated with null α_i in SVM.

The effect of adopting this new criterion is presented in Figs. 3 and 4. In Fig. 3, support vectors are selected according to $|\alpha|$ and, as can be observed, selected support vectors are not only in the separation margin, but also in the inner regions of the classes. With the use of the new criterion, only vectors in the margin were selected (Fig. 4). It is shown in Section 5 that this also leads to a higher similarity with the support vectors obtained by QP SVM.

4.1. The proposed classifier

The proposed criterion is applied to *IP − LSSVM* in order to eliminate non-relevant columns of the original matrix $\mathbf{A}$ and to build a

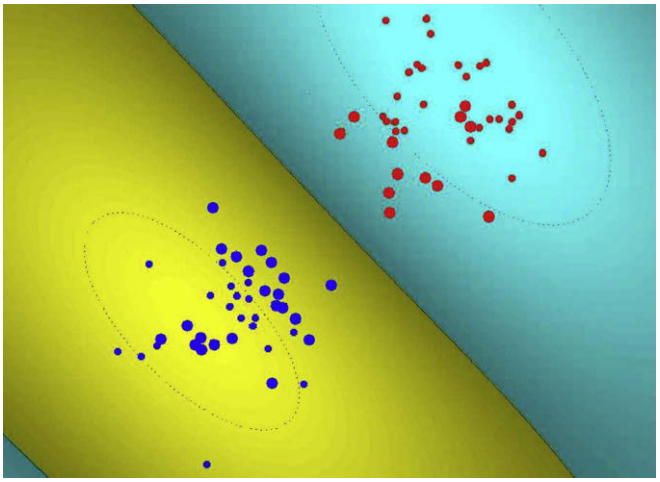

Fig. 3. Common relevance criterion: support vectors (bigger points) of LS-SVM using $|\alpha|$ criterion.

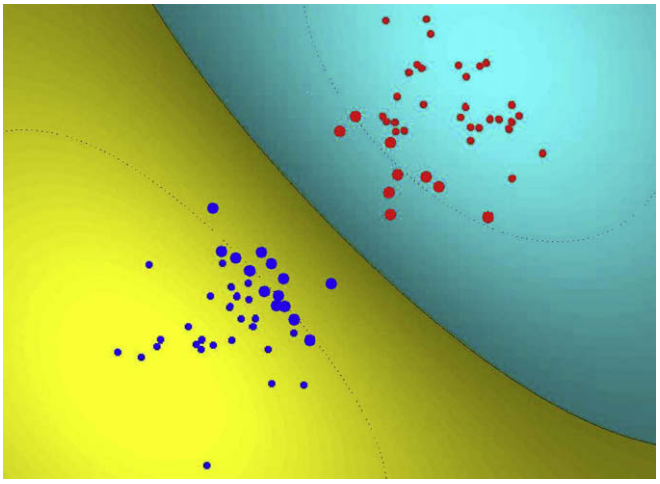

Fig. 4. Proposed relevance criterion: support vectors (bigger points) of LS-SVM using α criterion.

non-squared reduced matrix $\mathbf{A_2}$ to be used in the second step. The removed columns correspond to the least relevant vectors for the classification problem, selected according to their Lagrange multiplier values. The rows of $\mathbf{A}$ are not removed, because its elimination would lead to a loss of labeling information and in performance (Valyon and Horváth, 2004).

The first step is accomplished by using the inverse function to solve the system of linear equations represented by Eq. (3). The solution of this system is $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$, being $\mathbf{X}$ a vector of length $N+1$, where N is the number of training points. The first element of $\mathbf{X}$ is discarded, since it corresponds to a bias value and only α values are aimed in this vector elimination phase.

In second step, a system $\mathbf{A_2X_2} = \mathbf{B}$ is solved using the pseudo-inverse function, whose solution $\mathbf{X_2}$ corresponds to the bias term b (first element) and the Lagrange multipliers α_i associated to the training vectors $\mathbf{x}_i$ (other elements).

The training process of the proposed sparse classifier can be described as:

(1) The system of linear equations (3) is solved, with all training vectors, using $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$, since $\mathbf{A}$ is a square matrix.
(2) The parameter $\tau \in \{0, 1\}$ defines the fraction of training vectors that will be considered support vectors.
(3) The training vectors $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$ are ordered by their α values.
(4) The fraction $(1-\tau)$ of training data that corresponds to the smaller α values is selected.
(5) The non-squared matrix $\mathbf{A_2}$ is generated by removing from $\mathbf{A}$ the columns associated to the selected elements of α.
(6) The new system of linear equations, represented by $\mathbf{A_2X_2} = \mathbf{B}$, is solved as $\mathbf{X_2} = \mathbf{A_2^+B}$.
(7) The training points of $\mathbf{A_2}$ are the support vectors.
(8) The α and b values are obtained from the solution $\mathbf{X_2}$.

The proposed sparse classifier has the same training parameters of LS-SVM, the regularization parameter γ and the parameters of the kernel function. It has also another parameter, τ, which corresponds to the fraction of training vectors that will be selected as support vectors. Since the value of τ is not critical it can be easily adjusted after γ is set.

5. Results and discussion

The results of the experiments carried out with four UCI databases (Blake and Merz, 1998) (Ionosphere, Pima Indians Diabetes, Bupa Liver Disorder and Tic Tac Toe) are presented in this section. The description of these databases can be found in Gestel et al. (2004). The experiments were carried out by applying *SVM* (QP), $IP-LSSVM, LS^2-SVM$, $Pruning, Ada-Pinv$ and $RRS+LS-SVM$ with both linear and RBF kernels. In order to visualize the decision surfaces, two synthetic two-dimensional databases, three clusters and Chess board, were also included in the experiments with RBF kernel. The parameters used for SVM and LS-SVM for the UCI databases, presented in Tables 1 and 2, were obtained from Gestel et al. (2004). In order to select SVM and LS-SVM parameters for the synthetic databases, a *grid search* approach, that gradually refines the parameters, was used.

All the classifiers were fully implemented on Matlab 6.1 (Math Works Inc., 1991). The SVM implementation was obtained from Gunn (2000) and the LS-SVM from Carvalho and Braga (2005). Every result presented corresponds to the mean value of 20 trials, each one with randomly selected training and testing data. For each trial, all classifiers were evaluated with the same data partition. All classifiers use 2/3 of the data for training and 1/3 for testing, except *Pruning* that uses half of its training set for validation. Training time, testing accuracy and the percentage of support vectors are presented for all experiments. The surfaces obtained with the synthetic datasets are presented in Figs. 5–16.

The results in column 3 of the Tables 3–12 should be compared with the ones obtained by SVM. The closer they are to the

Table 1
SVM parameters used for UCI and synthetic datasets.

Database	Parameter	Kernel	Value
Three clusters	C	RBF	100
	σ	RBF	0.05
Chess board	C	RBF	1.5
	σ	RBF	1.5
Pima indians diabetes	C	Linear	0.008
	C	RBF	1.096
	σ	RBF	15.5
Tic tac toe	C	Linear	0.000056
	C	RBF	0.389
	σ	RBF	9.0
Bupa liver disorder	C	Linear	37.15
	C	RBF	43.65
	σ	RBF	9.0
Ionosphere	C	Linear	15.85
	C	RBF	3.24
	σ	RBF	3.3

Table 2
LS-SVM parameters used for UCI and synthetic datasets.

Database	Parameter	Kernel	Value
Three clusters	γ	RBF	100
	σ	RBF	0.05
Chess board	γ	RBF	1.5
	σ	RBF	1.5
Pima indians diabetes	γ	Linear	0.617
	γ	RBF	1096.5
	σ	RBF	240.0
Tic tac toe	γ	Linear	0.002
	γ	RBF	158.49
	σ	RBF	2.93
Bupa liver disorder	γ	Linear	1.38
	γ	RBF	1023.3
	σ	RBF	41.25
Ionosphere	γ	Linear	0.01
	γ	RBF	4.27
	σ	RBF	3.3

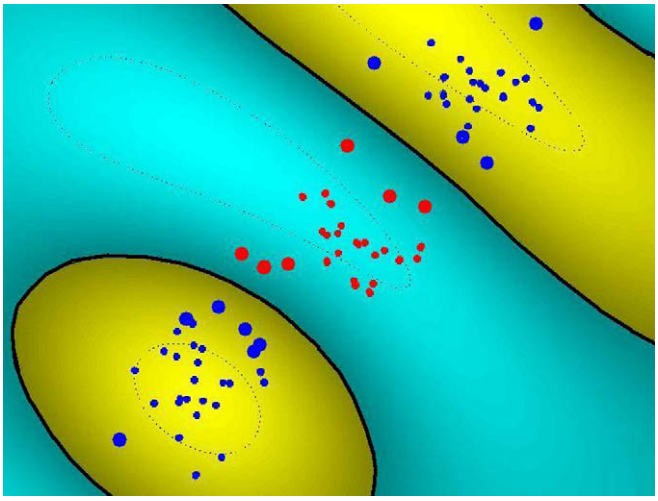

Fig. 5. *SVM* decision surface using RBF kernel for three clusters database.

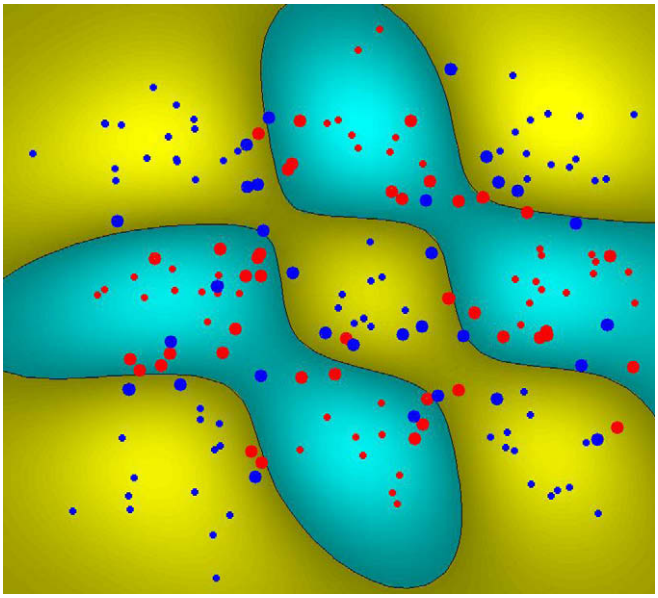

Fig. 6. *SVM* decision using RBF kernel for Chess board database.

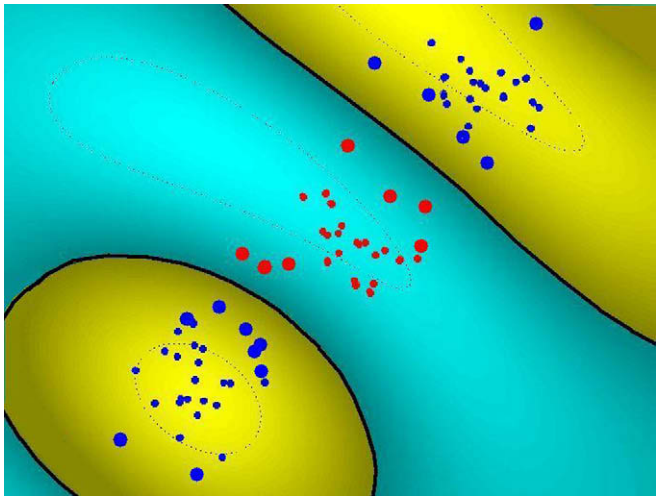

Fig. 7. *IP – LSSVM* decision surface using RBF kernel for three clusters database.

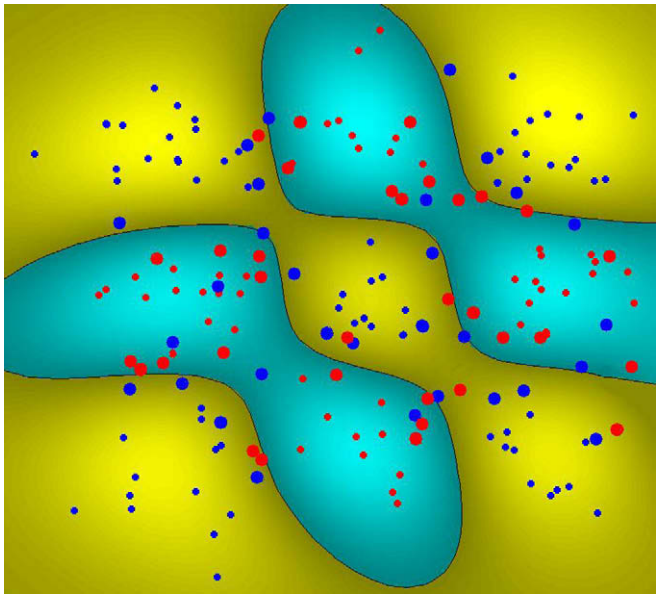

Fig. 8. *IP – LSSVM* decision using RBF kernel for Chess board database.

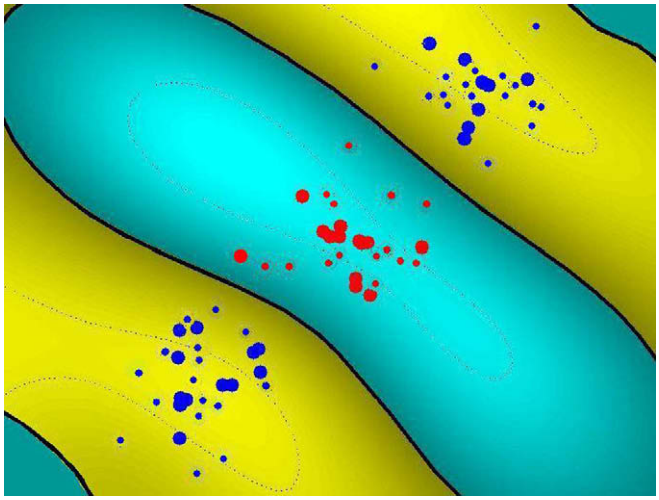

Fig. 9. LS^2 *– SVM* decision surface using RBF kernel for three clusters database.

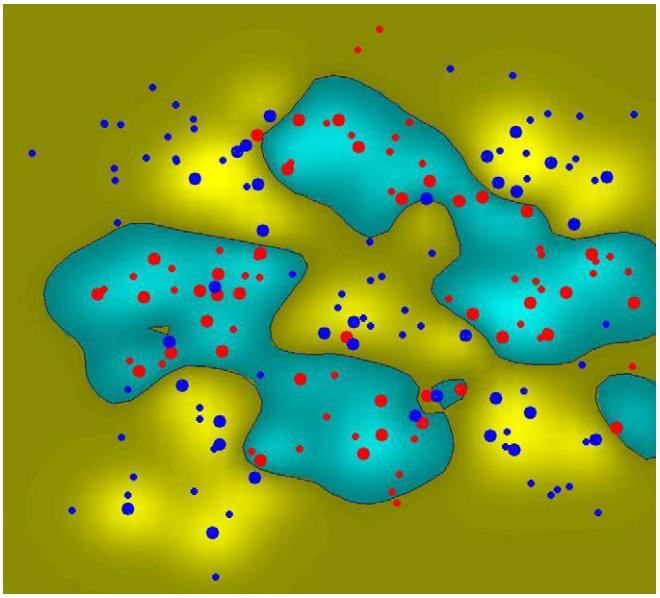

Fig. 10. $LS^2 - SVM$ decision using RBF kernel for Chess board database.

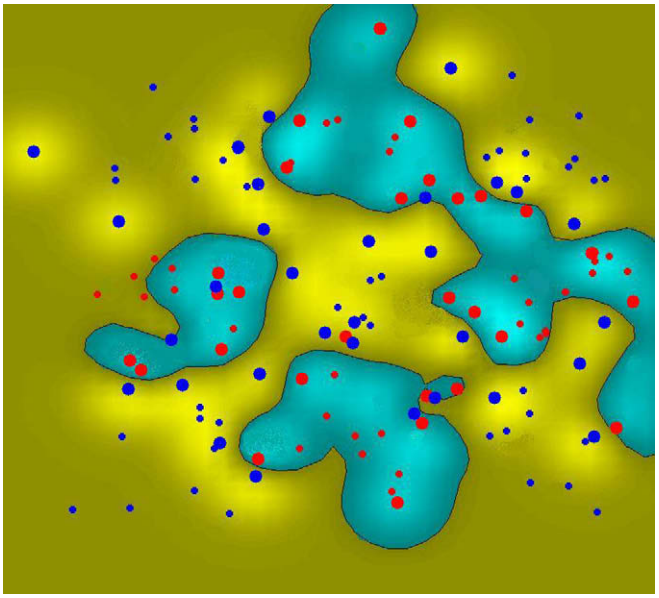

Fig. 12. *Pruning* decision using RBF kernel for Chess board database.

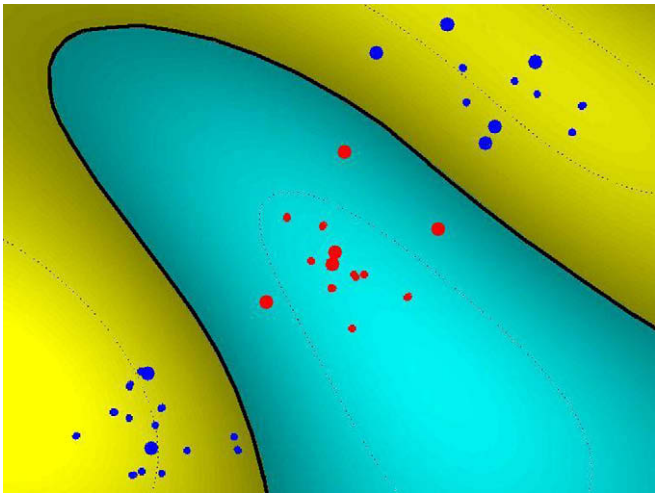

Fig. 11. *Pruning* decision surface using RBF kernel for three clusters database.

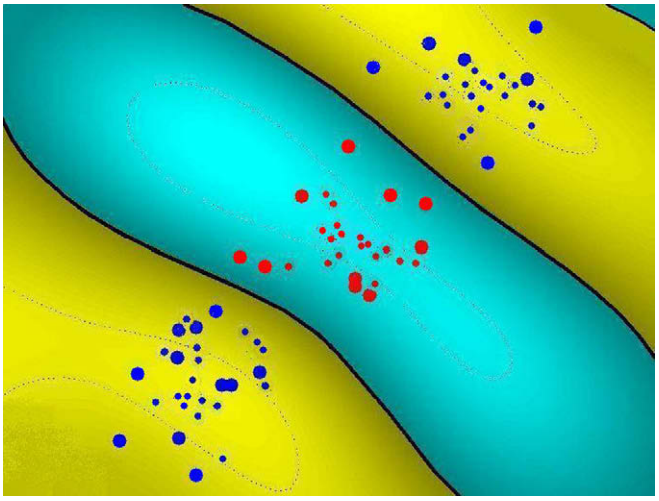

Fig. 13. *Ada – Pinv* decision surface using RBF kernel for three clusters database.

percentage obtained by SVMs, the better is the model in terms of support vectors detection. For instance, in Table 7 it is possible to observe that $IP - LSSVM, LS^2 - SVM$ and $Ada - Pinv$ have selected nearly the same percentage of support vectors as the quadratic programming SVM (0.50 compared to 0.46), but $Ada - Pinv$ has the lowest Testing Accuracy (94.2) and between $IP - LSSVM$ and $LS^2 - SVM$, the former had smaller training time with the same accuracy. So, by accomplishing this analysis, it is possible to conclude that $IP - LSSVM$ had the best performance.

5.1. Experiments with linear kernel

The experiments with linear kernels have shown that the testing accuracy differs a bit according to the model. Although the discrepancy among them is not larger the 4%. It is noticeable that $IP - LSSVM$ has one of the top testing performances, comparable to SVMs, in all datasets. In addition, $IP - LSSVM$ has one of the smallest training times, much lower than SVMs, in all experiments. Support vector detection numbers are also very close to SVMs, except for the Pima Indians dataset (Table 4), for which a larger support vectors set was associated with a higher performance of $IP - LSSVM$. In general, $Ada - Pinv$ and $RRS + LSSVM$ had the worst training time performance in all datasets, while *Pruning* was regularly among the best test set classifiers.

5.2. Experiments with RBF kernel

As can be observed in the experiments of Tables 7–12, $IP - LSSVM$ was also the fastest classifier in all experiments with RBF kernels, followed by *Pruning* and $LS^2 - SVM$, while $Ada - Pinv$ and $RRS + LSSVM$ are still the slowest ones. The largest difference between $IP - LSSVM$ and $RRS + LSSVM$ training times is presented in the results of Table 9 in which they differ by 100 times. Since all models are large margin classifiers, as expected, testing accuracy presented in Tables 7–11 are relatively close, not differing more than 5%. $IP - LSSVM$ results are the only ones that are among the three best testing accuracies of all experiments, followed by $LS^2 - SVM$.

It can be observed in the results of Tables 3–12 that $IP - LSSVM$ appears seven times in the top-two highest testing accuracies,

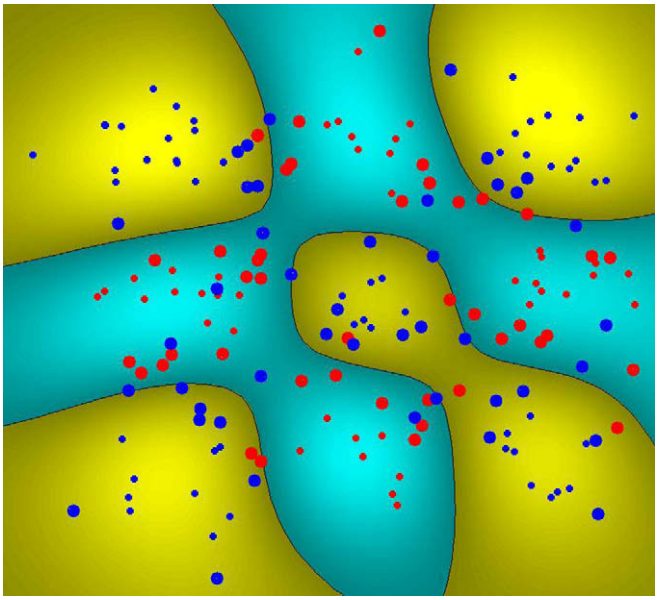

Fig. 14. *Ada – Pinv* decision using RBF kernel for Chess board database.

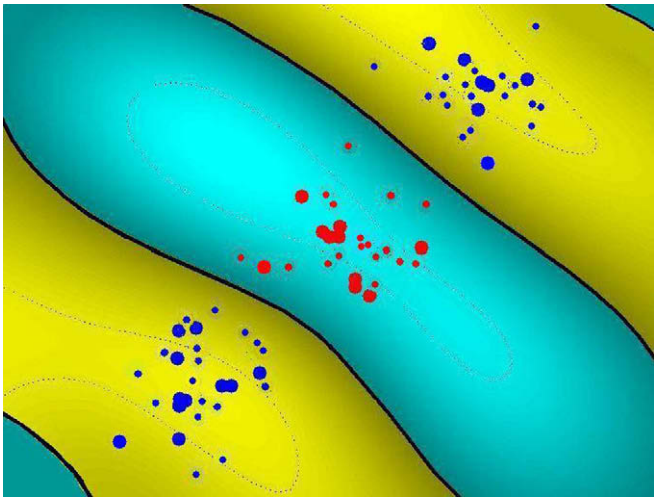

Fig. 15. *RRS + LS – SVM* decision surface using RBF kernel for three clusters database.

while *SVM* and $LS^2 - SVM$ have appeared only five and four times, respectively. The results of Tables 7–12 show also that the fraction of support vectors *IP – LSSVM* and *Ada – Pinv* yield were closer to those obtained by *SVM*, followed by $LS^2 - SVM$. In contrast with that, *Pruning* and *RRS + LS – SVM* resulted on different support vectors sets compared with *SVMs*, what was often associated with testing accuracy lower than other methods.

5.3. Comparison with literature results

Table 13 presents the comparison of testing accuracies between *IP – LSSVM* and SVM's state-of-the-art results (Gestel et al., 2004). As can be observed, the results obtained by *IP – LSSVM* are very close to those found in the literature for SVMs. It is important to notice that the slight deviations in the results are related to the random choice of training and testing sets used in the experiments, since fixed partitions are not provided for comparison. The table indicate that the results obtained by *IP – LSSVM* are comparable with those found in the literature for SVMs, confirming that the proposed approach holds the simplicity of the least squares approach without loss in performance.

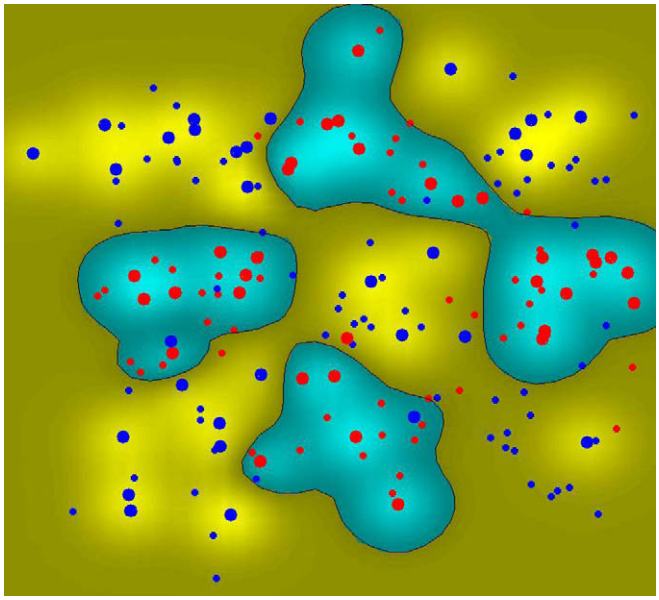

Fig. 16. *RRS + LS – SVM* decision using RBF kernel for Chess board database.

Table 3
Results for Ionosphere database with linear kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	37.20 ± 2.84	88.8 ± 2.7	15.0 ± 1.0
IP – LSSVM	5.05 ± 0.02	88.7 ± 3.2	15.0 ± 0.0
$LS^2 - SVM$	10.53 ± 0.82	84.2 ± 3.9	17.0 ± 1.0
Pruning	9.42 ± 3.11	85.4 ± 5.1	26.0 ± 9.0
Ada – Pinv	467.4 ± 19.5	84.5 ± 5.3	15.0 ± 1.0
RRS + LS – SVM	233.1 ± 11.8	84.8 ± 5.2	23.0 ± 0.0

Table 4
Results for Pima indians diabetes database with linear kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	560.9 ± 31.2	73.1 ± 3.1	76.0 ± 10.0
IP – LSSVM	27.75 ± 0.08	76.1 ± 2.8	90.0 ± 0.0
$LS^2 - SVM$	75.01 ± 0.24	75.9 ± 2.9	90.0 ± 0.0
Pruning	18.00 ± 6.72	76.3 ± 3.2	47.0 ± 8.0
Ada – Pinv	2411.3 ± 9.5	76.0 ± 3.0	90.0 ± 3.0
RRS + LS – SVM	1545.2 ± 40.6	76.2 ± 2.6	63.0 ± 2.0

Table 5
Results for Bupa liver disorder database with linear kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	24.34 ± 1.80	67.3 ± 2.3	72.0 ± 3.0
IP – LSSVM	4.36 ± 0.02	67.6 ± 2.4	75.0 ± 0.0
$LS^2 - SVM$	12.32 ± 0.12	67.1 ± 3.0	76.0 ± 2.0
Pruning	4.62 ± 2.42	67.1 ± 3.0	44.0 ± 8.0
Ada – Pinv	447.3 ± 13.0	66.9 ± 2.6	75.0 ± 5.0
RRS + LS – SVM	333.7 ± 20.7	64.8 ± 4.0	37.0 ± 3.0

5.4. Decision surfaces

In order to provide a qualitative analysis of the support vectors detected by all methods, the decision surfaces obtained for the three clusters and Chess board datasets are presented in

Table 6
Results for Tic tac toe database with linear kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	1138.6 ± 5.8	64.6 ± 3.5	69.0 ± 3.0
IP − LSSVM	50.11 ± 0.27	70.4 ± 2.0	70.0 ± 0.0
$LS^2 - SVM$	127.37 ± 0.74	71.4 ± 3.0	71.0 ± 2.0
Pruning	20.70 ± 0.71	70.0 ± 3.9	50.0 ± 0.0
Ada − Pinv	3652.4 ± 20.0	65.2 ± 2.1	70.0 ± 2.0
RRS + LS − SVM	1657.0 ± 19.3	64.4 ± 5.0	66.0 ± 6.0

Table 7
Results for Ionosphere database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	24.96 ± 4.17	94.6 ± 1.7	46.0 ± 1.0
IP − LSSVM	5.96 ± 0.02	95.3 ± 2.1	50.0 ± 0.0
$LS^2 - SVM$	13.85 ± 0.19	95.3 ± 1.7	50.0 ± 2.0
Pruning	10.9 ± 7.7	93.1 ± 3.2	35.0 ± 1.0
Ada − Pinv	455.0 ± 27.7	94.2 ± 4.2	50.0 ± 0.0
RRS + LS − SVM	227.1 ± 8.9	90.2 ± 2.8	32.0 ± 7.0

Table 8
Results for Pima indians diabetes database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	695.9 ± 87.3	75.5 ± 1.3	71.0 ± 1.0
IP − LSSVM	35.25 ± 0.37	74.9 ± 2.8	75.0 ± 0.0
$LS^2 - SVM$	79.00 ± 0.36	73.8 ± 2.5	74.0 ± 0.0
Pruning	46.2 ± 11.7	73.9 ± 2.3	45.0 ± 9.0
Ada − Pinv	2427.3 ± 15.0	73.8 ± 2.4	75.0 ± 0.0
RRS + LS − SVM	1388.5 ± 27.4	74.5 ± 3.0	81.0 ± 6.0

Table 9
Results for Bupa liver disorder database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	49.19 ± 4.51	62.9 ± 7.6	81.0 ± 1.0
IP − LSSVM	4.21 ± 0.03	66.7 ± 3.7	85.0 ± 2.0
$LS^2 - SVM$	12.08 ± 0.09	67.3 ± 3.8	86.0 ± 1.0
Pruning	4.72 ± 1.96	64.7 ± 4.3	47.0 ± 8.0
Ada − Pinv	452.6 ± 21.9	66.6 ± 4.1	85.0 ± 8.0
RRS + LS − SVM	339.8 ± 30.1	65.7 ± 2.3	62.0 ± 9.0

Table 10
Results for Tic tac toe database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	1276.9 ± 12.8	96.0 ± 0.6	71.0 ± 0.0
IP − LSSVM	67.36 ± 0.42	98.7 ± 0.6	75.0 ± 0.0
$LS^2 - SVM$	543.22 ± 2.45	98.5 ± 0.6	84.0 ± 0.0
Pruning	116.1 ± 29.2	96.4 ± 3.7	22.0 ± 8.0
Ada − Pinv	3762.0 ± 79.0	98.8 ± 0.3	75.0 ± 8.0
RRS + LS − SVM	1255.9 ± 34.2	96.6 ± 5.0	55.0 ± 9.0

Figs. 5–16. As pointed out in the initial sections, it can be noted from Figs. 5 and 6 that *SVM* considers the training points at the border region as support vectors. The support vectors detected by *IP − LSSVM* (Figs. 7 and 8), by adopting the criterion of selecting the patterns with positive Lagrange multipliers only, were nearly the same as those obtained by *SVM*. $LS^2 - SVM$ considered some

Table 11
Results for Three clusters database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	1.61 ± 0.28	97.7 ± 1.8	7.0 ± 1.0
IP − LSSVM	0.54 ± 0.04	98.0 ± 1.9	12.0 ± 2.0
$LS^2 - SVM$	1.18 ± 0.05	98.2 ± 1.6	11.0 ± 2.0
Pruning	1.06 ± 0.50	98.7 ± 1.3	17.0 ± 9.0
Ada − Pinv	2.83 ± 0.16	97.7 ± 0.8	12.0 ± 3.0
RRS + LS − SVM	2.48 ± 0.15	98.0 ± 0.9	10.0 ± 1.0

Table 12
Results for Chess board database with RBF kernel used.

Method	Training time (s)	Testing accuracy (%)	Support vectors (%)
SVM	13.98 ± 0.06	92.0 ± 1.0	46.0 ± 2.0
IP − LSSVM	1.77 ± 0.12	91.0 ± 0.6	45.2 ± 1.4
$LS^2 - SVM$	7.78 ± 0.41	89.2 ± 0.4	41.0 ± 2.4
Pruning	12.44 ± 0.30	86.5 ± 1.5	32.5 ± 4.0
Ada − Pinv	23.21 ± 0.80	88.8 ± 1.3	47.0 ± 2.2
RRS + LS − SVM	23.77 ± 0.10	91.0 ± 1.8	35.2 ± 3.5

Table 13
State-of-art accuracies of UCI databases used in this work.

Database	Kernel	IP-LSSVM's accuracy (%)	Literature's accuracy (%)
Ionosphere	Linear	88.7 ± 3.2	87.9 ± 2.0
Pima indians diabetes	Linear	76.1 ± 2.8	76.8 ± 1.8
Bupa liver disorder	Linear	67.6 ± 2.4	65.6 ± 3.2
Tic tac toe	Linear	70.4 ± 2.0	66.8 ± 3.9
Ionosphere	RBF	95.3 ± 2.1	96.0 ± 2.1
Pima indians diabetes	RBF	74.9 ± 2.8	76.8 ± 1.7
Bupa liver disorder	RBF	66.7 ± 3.7	70.2 ± 4.1
Tic tac toe	RBF	98.7 ± 0.6	99.0 ± 0.3

different training points as support vectors, which are not at the border, although the decision surface is very similar to that obtained by *Ada − Pinv* and *RRS + LS − SVM* for three clusters database. *Pruning* support vectors differ a bit from those obtained by *SVM*, what explains the decision surface displayed in Figs. 11 and 12. In general, the other methods found also support vectors that are in the inner regions of the distributions, what is explained by the loss of sparseness of their Lagrange multipliers vector.

6. Conclusions

The description of SVMs caused a large impact in machine learning research, since it provides a maximum margin classifier with minimum error in the testing set. Maximum margin is yielded by the quadratic programming formulation that results also on non-zero Lagrange multipliers associated to patterns located near the separation margin. The support vectors outlined by the non-zero Lagrange multipliers are in fact a product of optimization. Alternatives to the quadratic programming approach have appeared in the literature, since other formulations for the general constrained optimization problem are possible. These alternative formulations, such as the LS-SVM, are not able to embody all the features of the original formulation. In particular, there is a loss in sparseness of the Lagrange multipliers vector, that do not follow the constraint of not being null only in the separation margin. We have shown in this paper, that the proposed model, *IP − LSSVM*, is also able to detect support vectors in the margin, while

maintaining the simplicity of the least squares approach. It represents a viable alternative to SVMs, since both have similar features, supported by literature results and yet *IP – LSSVM* has a simpler and more understandable formulation.

Acknowledgements

Antônio de Pádua Braga would like to thank CNPq for the academic support and Bernardo P.R. Carvalho would like to thank Ottimah Process Improvements Ltd for providing the means for writing this work.

References

Blake, C.L., Merz, C.J., 1998. UCI Repository of Machine Learning Databases. University of California, Irvine.

Carvalho, B.P.R., Braga, A.P., 2005. New strategies for training least squares support vector machines. In: National Meeting on Artificial Intelligence (ENIA'2005), Sao Leopoldo, Brazil.

Carvalho, B.P.R., Lacerda, W.S., Braga, A.P., 2007. RRS + LS-SVM: A New Strategy for a Priori Sample Selection. Neural Computing and Applications. Springer, London.

Ganapathiraju, A., Picone, J., 2000. Support vector machines for automatic data cleanup. In: Internat. Conf. on Spoken Language Processing (ICSLP'2000), Beijing, China.

Gestel, T.V., Suykens, J.A.K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., Moor, B., Vandewalle, J., 2004. Benchmarking least squares support vector machine classifiers. Machine Learning 54 (1), 5–32.

Gunn, S., 2000. <http://www.isis.ecs.soton.ac.uk/isystems/kernel/>.

Lee, Y.J., Mangasarian O.L., 2001. RSVM: Reduced support vector machines. In: First SIAM Internat. Conf. on Data Mining, Chicago, United States.

Math Works Inc., 1991. MATLAB for Windows User's Guide.

Suykens, J.A.K., Vandewalle, J., 1999. Least squares support vector machine classifiers. Neural Process. Lett. 9 (3), 293–300.

Suykens, J.A.K., Lukas, L., Vandewalle, J., 2000. Sparse least squares support vector machine classifiers. In: European Symposium of Artificial Neural Networks (ESANN'2000), Bruges, Belgium.

Tax, D.M.J., Duin, R.P.W., 1999. Data domain description using support vectors. In: European Symposium on Artificial Neural Networks (ESANN'1999), Bruges, Belgium.

Valyon, J., Horváth, G., 2004. A sparse least squares support vector machine classifier. In: Internat. Joint Conf. on Neural Networks (IJCNN'2004), Hungary, Budapest.

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer-Verlag.