ESANN'2007 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2007, d-side publi., ISBN 2-930307-07-2.

A-LSSVM: An Adaline Based Iterative Sparse LS-SVM Classifier

Bernardo Penna Resende de Carvalho[1] and Antônio de Pádua Braga[1]

1- Ottimah Process Improvements - Dep. of Research and Development
Padre Marinho St. 37, 13th floor, Santa Efigênia, BH (MG), 30.140-040 - Brazil
E-mail: bpenna@gmail.com

2- Federal University of Minas Gerais - Dep. of Electronic Engineering
Antônio Carlos Av. 6627, Pampulha, Belo Horizonte (MG), 31.270-901 - Brazil
E-mail: apbraga@cpdee.ufmg.br

Abstract.
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. We present in this work a new method, $A-LSSVM$, which uses the neural model Adaline to solve the LS-SVM's linear system while automatically finding the support vectors. The proposed approach is compared with other methods in literature to impose sparseness in LS-SVM: $Pruning$, LS^2-SVM, $Ada-Pinv$ and $IP-LSSVM$. The experiments, performed on three important benchmark databases in Machine Learning, show that all sparse LS-SVMs have an accuracy near SVM, but differ in training time and support vectors found.

1 Introduction

Least Squares Support Vector Machine (LS-SVM) [1] is a learning machine that corresponds to a modified version of Support Vector Machine (SVM) [2]. The problem generated by LS-SVM can be solved with a system of linear equations, which is less complex than the quadratic programming used in SVM. Although this solution is simpler, there is a loss of sparseness in the feature vectors, since most Lagrange multipliers are non zero, what leads to all training data being considered as support vectors. As a consequence of this trade-off between sparseness and complexity, many works on LS-SVM are focused on improving support vectors representation in the least squares approach. Adaline neural model [3] is an appropriate proposition to solve the LS-SVM's linear system, which isn't positive definite, and has a not easy solution using common iterative methods.

We introduce in this work a new method, called $A-LSSVM$, in order to impose sparseness in LS-SVM. This sparse LS-SVM uses a modified version of Adaline to iteratively solve the LS-SVM's linear system and also detect the support vectors automatically. $A-LSSVM$ is compared with other sparse LS-SVM classifiers presented in literature on three important benchmark databases in Machine Learning [4]: Pima Indians Diabetes, Tic-Tac-Toe and Ionosphere. Comparing with other approaches, the proposed method has the advantages that does not need to firstly solve a complete LS-SVM system to obtain a solution,

because it runs iteratively. Despite this fact, the modified version of Adaline used at $A-LSSVM$ is very easy to be implemented.

The remainder of this paper is organized as follows. In Section 2, it is presented the LS-SVM formulation. In Section 3, $A-LSSVM$ is introduced and the other sparse LS-SVMs are described. After that, it is shown in Section 4 the experiments on three UCI databases, by applying SVM [2], $Pruning$ [5], LS^2-SVM [6], $Ada-Pinv$ [7], $IP-LSSVM$ [8] and $A-LSSVM$ with RBF kernel. Finally, the last section corresponds to the conclusions of this work.

2 Least Squares Support Vector Machine

Given the training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$, with input $\mathbf{x}_i \in \Re^n$ and corresponding binary output $y_i \in \{-1, +1\}$, LS-SVM maps the input data in a high dimensional space, called feature space, to build a linear separation hyperplane $f(\mathbf{x}) = 0$, such that

$$\omega^T \varphi(\mathbf{x}) + b = 0 \tag{1}$$

where ω is the weight vector, b is the bias term and $\varphi(.)$ is the mapping function applied to data to represent them at the feature space.

The primal problem of the LS-SVM is defined as

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

subject to

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

where γ controls the two terms of $J_P(\omega, b, \mathbf{e})$ and e_i is the error of pattern $\mathbf{x}_i$.

We apply the Lagrangean, in order to incorporate the equality constraints of the primal problem in the dual cost function, using the Lagrange multipliers α. Deriving the Lagrangean problem with respect to the primal and dual variables and setting the result to zero, which is needed to find its saddle point, gives

$$\begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \alpha \sum_{i=1}^{N} \sum_{j=1}^{N} (y_i y_j \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) + \frac{1}{\gamma}) + yb = 1. \end{cases} \tag{3}$$

We can describe (3) like a linear system $\mathbf{AX} = \mathbf{B}$ where

$$\mathbf{A} = \begin{bmatrix} 0 & -\mathbf{Y}^T \\ \mathbf{Y} & \mathbf{H} \end{bmatrix}, \mathbf{X} = \begin{bmatrix} b \\ \alpha \end{bmatrix} \; and \; \mathbf{B} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}. \tag{4}$$

The matrix $\mathbf{H}$ in (4) obeys the Mercer Theorem [9], that deals with the conditions that a given function $K(\mathbf{x}_i, \mathbf{x}_j)$ must have to be a kernel function. Therefore $\mathbf{H}$ is positive definite, the matrix $\mathbf{A}$ is not. The solution of the system of linear equations (4) is the same of the primal problem (2).

Once we have got the values of the Lagrange multipliers, the output of the LS-SVM can be calculated like the SVM

$$f(\mathbf{x}) = sign[\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b]. \tag{5}$$

3 Sparse LS-SVMs

The proposed method, $A-LSSVM$, is introduced now. After that, four sparse methods existent in literature are presented: $Pruning$, LS^2SVM, $AdaPinv$ and $IP-LSSVM$. Other sparse approaches for LS-SVM can be found at [10, 11, 12].

3.1 $A-LSSVM$

This sparse LS-SVM classifier uses a modified Adaline [3] to perform the LS-SVM training process. Considering X as the input matrix and Y the output one, each sample x_i and y_i in a row, the modified gradient descent training is

1. The weights are started randomly, and the output is $Y_k = W_k X$.
2. The weights are updated as $W_k = W_{k-1} + \eta(E_k)X$, where $E_k = D_k - Y_k$ is the error of all training data X at iteration k.
3. If the error of $X_{(i)}$ at iteration k is smaller than a given parameter, $X_{(i)}$ is eliminated together with its correspondent weight $W_{k(i)}$ and output $Y_{k(i)}$.
4. The training process continues until a stopping criterion is reached.

After the modified Adaline is used to solve the LS-SVM system (4), where the non-eliminated patterns are the support vectors, and their α values are got, the solution is obtained as described in (5).

3.2 $Pruning$

This method, which makes $LS-SVM$ capable to choose a subset of the training data as support vectors, was suggested in [5]. It was proposed the usage of a *pruning* technique [13], where training vectors $\mathbf{x}_i$ are eliminated according to the absolute value of the Lagrange multiplier, $|\alpha_i|$, associated with them.

The Lagrange multiplier elimination is performed recursively, in a way that in each iteration a small quantity of vectors is eliminated. It also uses a subset of the training data as a validation set. The stop criterion, which determines when the reduction of the training set is finished, is the decreasing of the machine accuracy into the validation set.

3.3 LS^2-SVM

This method was proposed in [6]. It is a two-phase strategy which automatically eliminates the irrelevant vectors, in order to detect the support vectors. The first phase is done by reducing the matrix $\mathbf{A}$ of the linear system (4) to the echelon form with a column elimination, where values that are lower than a specified threshold are set to zero, without a corresponding row elimination.

Because some values have been set to zero, the resulted matrix has columns with all zero elements, corresponding to linear dependent vectors, that must be discarded. By the fact that all rows remain, the new matrix $\mathbf{A}$ is non-squared, so the linear system (4) is solved using the pseudo-inverse function $\mathbf{X} = \mathbf{A}^{+}\mathbf{B}$.

3.4 $Ada - Pinv$

This method was introduced in [7], as another manner to impose the sparseness in LS-SVMs using also two phases. The first one is the use of the neural model Adaline [3] to perform the LS-SVM training process. The Lagrange multipliers resulted from this process are sorted according to their absolute value $|\alpha_i|$. The columns of matrix $\mathbf{A}$ of the linear system (4) are removed according to this value, because smaller $|\alpha|$ values mean correct classification. Notice that only the columns are removed, remaining all the rows, like LS^2SVM does.

The difference of this method compared to the last one is that there is no need for a process of matrix reduction to the echelon form; it is only needed to train a LS-SVM using Adaline, which is easier to implement. Once the column elimination is performed, the solution is obtained using $\mathbf{X} = \mathbf{A}^{+}\mathbf{B}$.

3.5 $IP - LSSVM$

This method was proposed in [8], and uses a new relevance criterion to select support vectors: the Lagrange multiplier α_i associated to each training point, and not its absolute value $|\alpha_i|$, that is often used in literature [1]. $IP - LSSVM$ uses the inverse function $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$ in the first step to solve the system of linear equations represented by (4), the same way LS-SVM does.

The new relevance criterion is used in the second phase in order to eliminate some columns of the original matrix $\mathbf{A}$ and build a non-squared matrix. Like occurs in $LS^2 - SVM$ and $Ada - Pinv$, the rows of $\mathbf{A}$ are not removed, because its elimination leads to a less constrained problem to be solved, with a worst solution. Once the column elimination was performed, the solution of the linear system (4) is got with $\mathbf{X} = \mathbf{A}^{+}\mathbf{B}$.

4 Results and discussion

In this section it's presented the results of the experiments made on three UCI databases [4], Pima Indians Diabetes, Tic-Tac-Toe and Ionosphere using the sparse classifiers with RBF kernels.

All the classifiers were implemented on $Matlab$ 6.1, and no toolbox was used. Each result corresponds to a mean value of 10 rounds of experiments, each round with a different random combination of training and testing data. The parameters used for SVM in the experiments were got from [11].

From Tables 1 to 3, we can observe that $IP - LSSVM$ is the fastest method on three databases, followed by $Pruning$ and $LS^2 - SVM$. $Ada - Pinv$ is the slower one in all experiments. $A - LSSVM$ is faster than $Ada - Pinv$, but slower than the other sparse methods. $IP - LSSVM$ has a training time 4 to 20 times smaller than SVM, while $Ada - Pinv$ is 4 to 40 times slower than SVM. The training accuracy of all classifiers are very similar, except by the result of SVM in Table 2. Tables 1 and 3 show $A - LSSVM$ has the best testing accuracy. But for all databases the sparse LS-SVMs presented very similar testing accuracies.

Table 1: Pima indians diabetes

Method	Training time (secs)	Training accuracy	Testing accuracy	Support vectors
SVM	695.9 ± 87.3	0.75 ± 0.01	0.76 ± 0.01	0.71 ± 0.01
$Pruning$	36.2 ± 11.7	0.74 ± 0.02	0.74 ± 0.02	0.45 ± 0.11
$LS^2 - SVM$	79.00 ± 0.36	0.75 ± 0.01	0.74 ± 0.03	0.74 ± 0.00
$Ada - Pinv$	2427.3 ± 15.0	0.75 ± 0.01	0.74 ± 0.02	0.75 ± 0.00
$IP - LSSVM$	35.25 ± 0.37	0.76 ± 0.01	0.75 ± 0.03	0.75 ± 0.00
$A - LSSVM$	1584.2 ± 49.0	0.76 ± 0.01	0.76 ± 0.03	0.73 ± 0.12

Table 2: Tic tac toe

Method	Training time (secs)	Training accuracy	Testing accuracy	Support vectors
SVM	1276.9 ± 12.8	0.65 ± 0.01	0.96 ± 0.01	0.71 ± 0.00
$Pruning$	116.1 ± 29.2	1.00 ± 0.00	0.96 ± 0.04	0.22 ± 0.18
$LS^2 - SVM$	543.22 ± 2.45	1.00 ± 0.00	0.99 ± 0.01	0.84 ± 0.00
$Ada - Pinv$	3762.0 ± 79	1.00 ± 0.00	0.98 ± 0.00	0.75 ± 0.00
$IP - LSSVM$	67.36 ± 0.42	1.00 ± 0.00	0.99 ± 0.01	0.75 ± 0.00
$A - LSSVM$	722.4 ± 59.0	0.97 ± 0.02	0.99 ± 0.01	0.70 ± 0.05

When we look at the number of support vectors detected by the sparse LS-SVMs, except $Pruning$, compared with the ones gotten by SVM, Tables 1 and 3 present similar results, which differ by no more than 4%. Considering the support vectors found, $A-LSSVM$ presented values more similar to SVM ones than others. On Tic-tac-toe database, the number of support vectors detected by $LS^2 - SVM$ is considerably higher than SVM, while $Pruning$ detected less support vectors than SVM in all databases.

Table 3: Ionosphere

Method	Training time (secs)	Training accuracy	Testing accuracy	Support vectors
SVM	24.96 ± 4.17	1.00 ± 0.00	0.95 ± 0.02	0.46 ± 0.01
$Pruning$	10.9 ± 9.7	1.00 ± 0.00	0.93 ± 0.03	0.35 ± 0.09
$LS^2 - SVM$	13.85 ± 0.19	0.98 ± 0.01	0.93 ± 0.02	0.50 ± 0.02
$Ada - Pinv$	455.0 ± 27.7	1.00 ± 0.00	0.94 ± 0.02	0.50 ± 0.00
$IP - LSSVM$	5.96 ± 0.02	1.00 ± 0.00	0.95 ± 0.02	0.50 ± 0.00
$A - LSSVM$	157.9 ± 5.3	0.98 ± 0.02	0.96 ± 0.02	0.43 ± 0.03

5 Conclusions

This work introduced a sparse LS-SVM classifier that is based on a modified version of Adaline to iteratively solve the LS-SVM's linear system and also to detect the support vectors automatically. The experiments performed indicate that all sparse LS-SVMs have a testing accuracy near SVM, but differ in the number of support vectors found and in the time spent on training.

$IP - LSSVM$ is the fastest method while $Ada - Pinv$ is the slower one. The proposed method, $A - LSSVM$, besides being easy to implement, is the only one that does not need to firstly solve a complete LS-SVM system to reach the solution, because it is iterative. $Pruning$ and $LS^2 - SVM$ differ more with respect to the number of support vectors, when compared with SVM. All sparse LS-SVMs were able to deal with the binary classification problems given by the benchmark data.

References

[1] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[2] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[3] B. Widrow and M. E. Hoff. Adaptive switching circuits. *IRE WESCON Convention Record*, 4:96–104, 1960.

[4] C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998.

[5] J. A. K. Suykens, L. Lukas, and J. Vandewalle. Sparse least squares support vector machine classifiers. In *ESANN'2000 European Symposium on Artificial Neural Networks*, pages 37–42, 2000.

[6] J. Valyon and G. Horváth. A sparse least squares support vector machine classifier. In *Int. Joint Conf. on Neural Networks (IJCNN'2004)*, 2004.

[7] B. P. R. Carvalho and A. P. Braga. New strategies for training *Least Squares Support Vector Machines*. In *V National Meeting on Artificial Intelligence (ENIA 2005)*, São Leopoldo, RS, 2005. URL: www.cpdee.ufmg.br/~bpenna/bpenna_ENIA_2005.pdf.

[8] B. P. R. Carvalho and A. P. Braga. Ip-lsvm: A two-step sparse classifier (submitted). *Pattern Recognition Letters*, 2006.

[9] J. Mercer. Functions of pos. and neg. types and their connection with the theory of integral equations. *Trans. of London Phil. Society (A)*, 209, 1909.

[10] B. Hamers, J. A. K. Suykens, and B. De Moor. A comparison of iterative methods for ls-svm classifiers. Technical report, ESAT, K.U.Leuven, 2001.

[11] T. Van Gestel, J. A. K. Suykens, B. Baesens, B. De Moor, and J. Vandewalle. Benchmarking least squares support vector machine classifiers. Technical report, ESAT, 2000.

[12] B. P. R. Carvalho, W. S. Lacerda, and A. P. Braga. A hybrid approach for sparse least squares support vector machines. In *V International Conference on Hybrid Intelligent Systems (HIS'05)*, Rio de Janeiro, RJ, 2005. URL: www.cpdee.ufmg.br/~bpenna/bpenna_HIS_2005.pdf.

[13] R. Reed. Pruning algorithms - a survey. *IEEE Transactions on Neural Networks*, 4(5):740–747, 1993.