ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

Sparse Least Squares Support Vector Machines via Multiresponse Sparse Regression

David Clifte S. Vieira, Ajalmar R. Rocha Neto, Antonio Wendell de O. Rodrigues *

Federal Institute of Ceará (IFCE), Department of Teleinformatics
Av. Treze de Maio, 2081, Benfica, Fortaleza, Ceará, Brazil

Abstract. Least Square Support Vector Machines (LSSVMs) are an alternative to SVMs because the training process for LSSVMs is based on solving a linear equation system while the training process for SVMs relies on solving a quadratic programming optimization problem. Despite solving a linear system is easier than solving a quadratic programming optimization problem, the absence of sparsity in the Lagrange multiplier vector obtained after training a LSSVM model is an important drawback. To overcome this drawback, we present a new approach for sparse LSSVM called Optimally Pruned LSSVM (OP-LSSVM). Our proposal is based on a ranking method, named Multiresponse Sparse Regression (MRSR), which is used to sort the patterns in terms of relevance. After that, the leave-one-out (LOO) criterion is also used in order to select an appropriate number of support vectors. Our proposal was inspired by a recent methodology called OP-ELM, which prunes hidden neurons of Extreme Learning Machines. Therefore, in this paper, we put LSSVM and MRSR to work together in order to achieve sparse classifiers, as well as one can see that we achieved equivalent (or even superior) performance for real-world classification tasks.

1 Introduction

Large margin classifiers such as Support Vector Machines (SVMs) and Least Squares SVMs (LSSVMs) have been used to handle classification tasks after being introduced by Vapnik and Suykens [1, 2]. A theoretical advantage of large margin classifiers is the empirical and structural risk minimization which balances the complexity of the model against its success at fitting the training data. Besides that, SVMs are able to produce sparse solutions [3]. By sparseness we mean that the decision surface, i.e, a hyperplane in the feature space built by the induced model (classifier or regressor) depends only on a relatively small number of input examples, the so-called support vectors (SVs).

The LSSVM is an alternative to the standard SVM formulation, since the solution for LSSVM is achieved by solving the linear systems resulting from the optimality conditions that appear from minimizing the primal optimization problem in a least square sense [2]. Therefore, the solution follows directly from solving a linear equation system, instead of solving a quadratic optimization (programming) problem (QP problem). On the one hand, in general, it is less computationally intensive to solve a linear system than a QP problem, since we only need to be able to compute the inverse of a matrix. On the other hand, the

*Corresponding authors: david.clifte@ppgcc.ifce.edu.br, ajalmar@ifce.edu.br

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

resulting solution is too far from sparse so that it is common to have all training samples being used as support vectors. To handle the lack of sparseness in LSSVM classifiers, several reduced set and pruning methods have been proposed. These methods comprise a bunch of techniques which aims at simplifying the internal structure of LSSVM classifiers, while keeping the decision boundaries as similar as possible to the original ones [4, 5].

To overcome this drawback, we present a new approach for sparse LSSVM called Optimally Pruned LSSVM (OP-LSSVM) based on a ranking method, named Multiresponse Sparse Regression (MRSR), which is used to sort the patterns in terms of relevance and, after that, we use the leave-one-out (LOO) criterion in order to select an appropriate number of support vectors.

In the next section, the LSSVM for classification is presented. Section 3 presents the Multiresponse Sparse Regression; as well as in Section IV, we present our proposal. In Section 4, we present some simulations we carried out, as well as some discussion about the results. At last, we present the conclusion in Section 6.

2 Least Square Support Vector Machines

Consider a training data set $\{\mathbf{x}_i, y_i\}_{i=1}^{L}$, so that $\mathbf{x}_i \in \mathbb{R}^p$ is an input vector and $y_i \in \{-1, +1\}$ are the corresponding class labels. For classification tasks, the primal problem formulation for LSSVM [2] is given by

$$\min_{\mathbf{w},\xi_i,b} \left\{ \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{i=1}^{L}\xi_i^2 \right\} \text{ subject to } \quad y_i[(\mathbf{w}^T\mathbf{x}_i) + b] = 1 - \xi_i, \qquad (1)$$

where b is the bias, γ is a positive regularization parameter and $\{\xi_i\}_{i=1}^{L}$ are the slack variables. By rearranging the Eq. (1), the Lagrangian function is achieved

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{i=1}^{L}\xi_i^2 - \sum_{i=1}^{L}\alpha_i(y_i(\mathbf{x}_i^T\mathbf{w} + b) - 1 + \xi_i), \qquad (2)$$

where $\{\alpha_i\}_{i=1}^{L}$ are the Lagrange multipliers.

In order to solve the problem, this Lagrangian function must be optimized with respect to $\mathbf{w}$, b, α_i and ξ_i. Therefore, we need to compute the following differentiations

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0}, \quad \frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial b} = 0 \quad \frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \alpha_i} = 0 \text{ and } \quad \frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha})}{\partial \xi_i} = 0,$$

resulting on

$$\mathbf{w} = \textstyle\sum_{i=1}^{L}\alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^{L}\alpha_i y_i = 0, \quad y_i(\mathbf{x}_i^T\mathbf{w} + b) - 1 + \xi_i = 0 \text{ and } \quad \alpha_i = \gamma\xi_i,$$

respectively. Thus, based on such achievements, one can formulate a linear system in order to represent the classification problem as

$$\mathbf{D}\mathbf{z} = \mathbf{l}, \qquad (3)$$

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

where

$$\mathbf{D} = \left[\begin{array}{c|c} 0 & \mathbf{y}^T \\ \hline \mathbf{y} & \boldsymbol{\Omega} + \gamma^{-1}\mathbf{I} \end{array}\right], \mathbf{z} = \left[\begin{array}{c} b \\ \hline \boldsymbol{\alpha} \end{array}\right] \text{and } \mathbf{l} = \left[\begin{array}{c} 0 \\ \hline \mathbf{1} \end{array}\right]. \tag{4}$$

Moreover, $\mathbf{I}$ is the identity matrix of size L, the symbol $\mathbf{1}$ denotes a L-dimensional vector of ones and $\Omega_{i,j} = y_i y_j {\mathbf{x}_i}^T \mathbf{x}_j$, where $i, j = 1, \ldots, L$.

The solution $\mathbf{z}$ for the linear system can be obtained by least squares such as

$$\mathbf{z} = \mathbf{D}^{-1}\mathbf{l}. \tag{5}$$

and then with the Lagrange multipliers and the bias in the solution $\mathbf{z}$, one can compute the output for classification problems by

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{l} \alpha_i y_i \mathbf{x}^T \mathbf{x}_i + b\right). \tag{6}$$

It is worthy emphasizing that whenever a Lagrange multiplier α_i is zero, we do not have to keep the associated input vector $\mathbf{x}_i$ on hand for future usage, see Eq. (6). It is also straightforward the usage of the *kernel trick*, which is applied to generate non-linear versions of the standard linear SVM classifier. One can do that by replacing the dot product $\mathbf{x}^T\mathbf{x}_i$ presented in Eq. (6), as well as in the calculation of $\Omega_{i,j}$ with the kernel function $k(\mathbf{x}, \mathbf{x}_i)$.

3 Multiresponse Sparse Regression

Consider the approximation of the linear system presented in Eq. (3) as

$$\mathbf{D}\mathbf{z}^k = \mathbf{l}^k \tag{7}$$

where, in terms of the multiresponse sparse regression (MRSR), $\mathbf{D}$ is the regressor matrix, $\mathbf{z}^k$ is the weight matrix (solution for the linear system) and, of course, $\mathbf{l}^k$ is the k-th approximation of $\mathbf{l}$. The matrix $\mathbf{z}^k$ is updated by

$$\mathbf{z}^{k+1} = (1 - \beta^k)\mathbf{z}^k + \beta^k \bar{\mathbf{z}}^{k+1}, \tag{8}$$

where β^k is the step size at k-th iteration so that

$$\beta^k = \min\{\beta | \beta^k \geq 0 \text{ and } \beta \in \Gamma_i \text{ for some } j \notin \mathcal{A}\}, \tag{9}$$

Γ_i is the set

$$\Gamma_i = \left\{ \frac{c^k_{max} + s^T(\mathbf{l} - \mathbf{l}^k)^T \mathbf{d}_j}{c^k_{max} + s^T(\bar{\mathbf{l}}^{k+1} - \mathbf{l}^k)^T \mathbf{d}_j} \right\}, \tag{10}$$

and c^k_{max} is the maximum cumulative correlation (i.e., $c^k_{max} = \max_j \left\{c^k\right\}$, such that $c^k = ||(\mathbf{l} - \mathbf{l}^k)^T \mathbf{d}_j||$) from the set of regressors that satisfy the maximum $\mathcal{A} = \{j | c^k_j = c^k_{\max}\}$. Note that $\mathbf{d}_j$ is the i-th column of $\mathbf{D}$.

In a nutshell, the idea behind the MRSR is to have, at the beginning, the matrix $\mathbf{z}^k$ with zero values and then to add a new nonzero row at each new step. Therefore, the weight matrix has k nonzero rows at k-th step of the MRSR.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

As each row is added one after another, the sequence formed represents the rank of rows. The row to be added is chosen by computing the cumulative correlation between the regressor (the vector $d_j|j \notin \mathcal{A}$) and the current residuals (the difference $\mathbf{l} - \mathbf{l}^k$). As the linear system output is a vector, the MRSR coincides with the LARS algorithm [6]. In fact, MRSR is an extension of LARS. More details about MRSR can be found in [7].

4 Our Proposal: Optimally Pruned LSSVM

In this section our proposal called Optimally Pruned LSSVM (OP-LSSVM) is presented. In a nutshell, our proposal relies on three main steps. The first step is to build the matrix $\Omega + \gamma^{-1}I$ from the data, as explained in the Eq. (4). The second one is to rank columns of $\mathbf{D}$ (which also means to rank patterns) by MRSR in order to obtain the most relevant columns. After that, the last step is the leave one-out optimization so that the best set of columns is achieved. The main idea is to leave the less important columns (patterns) out of the solution by eliminating such columns with low rank from the matrix $\mathbf{D}$ and then solving the problem by the pseudo-inverse. We highlight that the rows of $\mathbf{D}$ also associated with a certain pattern are not removed, because its elimination would lead to a loss of labeling information and in performance [8]. At this point, it is worth emphasizing that our proposal results from bringing the idea of ranking hidden neurons to prune ELMs proposed in [9] to LSSVMs, but ranking patterns. Despite the MRSR ranks the rows, we can use the row ranking as column ranking since the i-th row equals the i-th column (i.e., $\mathbf{D}^T = \mathbf{D}$). After ranking the patterns, the decision for the best set of support vectors in the model is taken by LOO validation method. In order to speed up the computation of LOO, the PRESS Statistic was used. More details can be found in [10].

At last, the Lagrange multipliers and bias are calculated in order to obtain the final model. We highlight that our proposal is inspired by a recent methodology, called OP-ELM, proposed to prune hidden layer neurons in Extreme Learning Machines (ELM) [9].

We present the proposed algorithm for OP-LSSVM below.

OP-LSSVM($\mathbf{D}, \mathbf{l}$)

$\mathbf{D}$ $\triangleright$ matrix build from the data
$\mathbf{l}$ $\triangleright$ vector of LSSVM linear system
1 $\mathbf{r} \leftarrow$ RANKING-BY-MRSR($\mathbf{D}$) $\triangleright$ Column Ranking
2 $\mathbf{s} \leftarrow$ SELECTING-BY-LOO($\mathbf{D}, \mathbf{r}$) $\triangleright$ Column Set
3 $\mathbf{D}_* \leftarrow$ PRUNE($\mathbf{D}, \mathbf{s}$) $\triangleright$ Non-square matrix
4 $\mathbf{z} = (\mathbf{D}_*^T\mathbf{D}_*)^{-1}\mathbf{D}_*^T\mathbf{l}$ $\triangleright$ pseudo-inverse solution
5 **return** $\mathbf{z}$ $\triangleright$ Lagrange multipliers and the bias

5 Simulations and Discussion

The results for some simulations carried out are presented in this section. From the total, two-thirds of the data examples were randomly selected for training

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Dataset	**HAB**	**PCV**	**DER**	**BCW**	**DIA**	**RIP**
Samples	306	310	366	699	768	1250
Training	201	204	241	461	506	825
Test	105	106	125	238	262	425
Variables	3	6	34	9	8	2

Table 1: Data sets used in our simulations.

purposes and so the remaining (one-third) of the examples were used for assessing the classifiers' generalization performances. Tests with real-world benchmarking datasets were also evaluated in this work. We used six datasets: Haberman (HAB), Pathologies of Vertebral Column (PVC), Dermatology (DER), Breast Cancer Wisconsin (BCW), Diabetes (DIA) and Ripley (RIP) from UCI Machine Learning Repository. Some information about the datasets is shown in Table 1. Since some datasets we are dealing with are not binary, we transform some of them into two-class problems for LSSVMs, such as PVC (normal or pathological individuals), DER (individuals with psoriasis or not).

Classifier		**HAB**	**PCV**	**DER**	**BCW**	**DIA**	**RIP**
SVM	ACC	**72.4 ± 3.3**	**87.1 ± 2.6**	**99.9 ± 0.3**	**96.4 ± 1.0**	**76.9 ± 2.3**	**88.0 ± 1.4**
	#SVs	97.4 ± 6.9	53.3 ± 6.3	22.9 ± 4.2	39.5 ± 10.9	240.4 ± 14.4	226.6 ± 36.5
	Time	9.6e+2±1.4e+3	2.5e+2±4.7e+2	9.6e-1±2.5e-1	2.3e+2±6.2e+2	6.0e+3±8.6e+3	3.9e+2±9.2e+2
LSSVM	ACC	**73.7 ± 3.2**	**83.5 ± 4.0**	**99.7 ± 0.5**	**96.1 ± 1.1**	**76.8 ± 2.1**	**87.9 ± 1.0**
	#SVs	201.0 ± 0.0	204.0 ± 0.0	241.0 ± 0.0	461.0 ± 0.0	506.0 ± 0.0	825.0 ± 0.0
	Time	6.6e+0±1.3e+0	5.6e+0±1.5e+0	8.0e+0±2.0e+0	4.0e+1±9.1e+0	4.7e+1±7.8e+0	1.4e+2±1.8e+1
IP-LSSVM$_{10\%}$	ACC	**70.6 ± 5.7**	**76.3 ± 6.4**	**99.2 ± 1.3**	**96.6 ± 0.8**	**73.5 ± 3.3**	**87.5 ± 2.3**
	#SVs	181.0 ± 0.0	184.0 ± 0.0	217.0 ± 0.0	415.0 ± 0.0	455.0 ± 0.0	742.0 ± 0.0
	Time	5.8e+0±2.4e+0	5.2e+0±2.2e+0	8.6e+0±3.5e+0	3.4e+1±8.4e+0	4.3e+1±1.1e+1	1.3e+2±2.0e+1
IP-LSSVM$_{20\%}$	ACC	**62.9 ± 9.1**	**67.9 ± 13.3**	**98.9 ± 1.7**	**96.8 ± 1.0**	**69.1 ± 10.0**	**87.2 ± 1.8**
	#SVs	161.0 ± 0.0	163.0 ± 0.0	193.0 ± 0.0	369.0 ± 0.0	405.0 ± 0.0	660.0 ± 0.0
	Time	5.3e+0±1.4e+0	4.7e+0±1.6e+0	7.8e+0±3.0e+0	3.6e+1±7.7e+0	4.5e+1±1.3e+1	1.4e+2±2.2e+1
IP-LSSVM$_{30\%}$	ACC	**60.2 ± 11.5**	**65.5 ± 15.6**	**98.6 ± 2.8**	**97.0 ± 1.0**	**53.1 ± 14.4**	**86.7 ± 1.6**
	#SVs	141.0 ± 0.0	143.0 ± 0.0	169.0 ± 0.0	323.0 ± 0.0	354.0 ± 0.0	577.0 ± 0.0
	Time	4.6e+0±1.7e+0	5.2e+0±1.9e+0	7.4e+0±2.8e+0	3.2e+1±8.4e+0	4.0e+1±1.1e+1	1.2e+2±1.4e+1
P-LSSVM	ACC	**72.2 ± 3.8**	**85.7 ± 3.0**	**98.5 ± 1.8**	**95.6 ± 2.5**	**76.1 ± 2.4**	**87.3 ± 2.1**
	#SVs	106.5 ± 19.1	131.7 ± 54.0	39.0 ± 58.1	38.7 ± 9.5	273.2 ± 59.8	224.9 ± 17.0
	Time	3.5e+1±8.5e+0	2.4e+1±2.0e+1	5.5e+1±1.8e+1	2.4e+2±2.9e+1	2.9e+2±8.0e+1	1.3e+3±1.1e+2
OP-LSSVM	ACC	**74.2 ± 3.1**	**83.9 ± 3.0**	**99.7 ± 0.5**	**95.8 ± 0.9**	**76.4 ± 2.6**	**88.0 ± 1.3**
	#SVs	83.0 ± 16.7	78.6 ± 12.8	89.8 ± 10.1	83.6 ± 13.0	82.8 ± 14.5	80.6 ± 17.8
	Time	1.4e+2±1.0e+2	1.3e+2±6.7e+1	1.6e+2±8.3e+1	5.8e+2±2.9e+2	6.5e+2±2.7e+2	1.6e+3±6.5e+2

Table 2: Accuracy and standard deviation after 30 independent runs. The time is present in seconds.

In Table 2, we present accuracy, average number of SVs (#SVs) and training time (Time) for aforementioned methodology on testing set averaged over 30 independent runs for SVM [11], LSSVM[2], P-LSSVM[4], IP-LSSVM [5] and our proposal (OP-LSSVM).

The classifier parameters were tuned by applying grid search with a 10-fold cross-validation over the training dataset, in which the regularization parameter γ for LSSVM was chosen from the interval $[2^{-5}, 2^{12}]$. For IP-LSSVM the fraction of training vectors that will be considered was chosen so that the number of training vector was defined as 70%, 80%, 90% of the full training dataset. We named them as IP-LSSVM$_{10\%}$, IP-LSSVM$_{20\%}$ and IP-LSSVM$_{30\%}$, respectively. Moreover, in each interaction of the P-LSSVM 1% of the trainning dataset was pruned until the loss performance reach a maximum of 2.5%.

By analyzing the Table 2, one can conclude that the performances achieved by OP-LSSVM were equivalent or even superior for most classifiers as well as can

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

also see that our proposal achieves sparse solutions so that the OP-LSSVM has sparser solutions than the other classifiers for all the datasets, except for PCV, DER and BCW for SVM classifier. This interesting achievement shows that we succeed in building sparse classifiers with less support vectors than LSSVM, P-LSSVM and IP-LSSVM, and even than SVMs.

The training time, in general, is greater when compared to others methodologies. However, as the number of training samples increases, the training time achieved by OP-LSSVM grows slower than LSSVM. Therefore, we support that for a larger dataset the OP-LSSVM training time will be smaller than LSSVM.

6 Conclusion

Our proposal named OP-LSSVM is an approach to train LSSVMs. Thus, we are able to obtain the Lagrange multipliers and the bias from the linear systems for LSSVMs. Based on the obtained results, we can conclude that we reach competitive classifiers in terms of accuracy while improving the model sparseness. Indeed, the presented proposal is an interesting alternative to the other classifiers, since the average of support vectors for OP-LSSVM was lower than the average for LSSVM, P-LSSVM, IP-LSSVM and SVM.

References

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

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

[3] I. Steinwart. Sparseness of support vector machines. *Journal of Machine Learning Research*, 4:1071–1105, 2003.

[4] J. A. K. Suykens, L. Lukas, and J. Vandewalle. Sparse least squares support vector machine classifiers. In *Proceedings of the 8th European Symposium on Artificial Neural Networks (ESANN'00)*, pages 37–42, 2000.

[5] B. P. R. Carvalho and A. P. Braga. IP-LSSVM: A two-step sparse classifier. *Pattern Recognition Letters*, 30:1507–1515, 2009.

[6] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Ann. Statist.*, 32(2):407–499, 04 2004.

[7] Timo Simil and Jarkko Tikka. Multiresponse sparse regression with application to multidimensional scaling. In *Artificial Neural Networks: Formal Models and Their Applications ICANN 2005*, volume 3697 of *Lecture Notes in Computer Science*, pages 97–102. Springer Berlin Heidelberg, 2005.

[8] J. Valyon and G. Horvath. A sparse least squares support vector machine classifier. In *IEEE IJCNN, 2004.*, volume 1, pages –548, 2004.

[9] Yoan Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse. Op-elm: Optimally pruned extreme learning machine. *Neural Networks, IEEE Transactions on*, 21(1):158–162, Jan 2010.

[10] R.H. Myers. *Classical and Modern Regression with Applications*. Duxbury classic series. Duxbury/Thompson Learning, 1990.

[11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http:www.csie.ntu.edu.tw cjlinlibsvm.