Model Selection for LS-SVM : Application to Handwriting Recognition

Mathias M. Adankon and *Mohamed Cheriet*
Synchromedia Laboratory for Multimedia Communication in Telepresence,
École de Technologie Supérieure, Montréal, H3C 1K3, QC, Canada
mathias@livia.etsmtl.ca, mohamed.cheriet@etsmtl.ca

Abstract

Support Vector Machine(SVM) is a powerful classifier used successfully in many pattern recognition problems. Furthermore, the good performance of SVM classifier has been shown in handwriting recognition field. Least Squares SVM, like SVM, is based on the margin-maximization principle performing structural risk, but its training is easier: it is only needed to solve a convex linear problem rather than the quadratic problem in SVM. In this paper, we propose to perform model selection for Least Squares SVM by using empirical error criterion. Experiments on handwriting character recognition show the usefulness of this classifier and demonstrate that LS-SVM generalization performance is improved with model selection.

Keywords: LS-SVM, support vector machine, model selection.

1. Introduction

Support vector machines are particular classifiers that are based on the margin-maximization principle. They perform structural risk minimization, which was introduced to machine learning by Vapnik, and which have yielded excellent generalization performance [22, 21]. For nonlinear problems, SVMs use the kernel trick to produce nonlinear boundaries. The idea behind kernels is to map training data nonlinearly into a higher-dimensional feature space via a mapping function Φ and to construct a separating hyperplane that maximizes the margin. The construction of the linear decision surface in this feature space only requires the evaluation of dot products $\Phi(x).\Phi(y) = k(x, y)$, where $k()$ is called the kernel function [10, 15, 16].

Least squares support vector machine (LS-SVM) is a variant of standard SVM. It is the result of the following question: "*how much one may simplify the SVM formulation without losing any of its advantages?* ", Suykens and Vandewalle [12] proposed LS-SVM where the training algorithm solves a convex problem like SVM. It has been shown by a meticulous empirical study that the generalization performance of LS-SVM is comparable to that of SVM [19]. In addition, the training algorithm of LS-SVM is very simplified since a linear problem is resolved instead of a quadratic programming (QP) problem in the SVM case.

Like SVM, LS-SVM is based on the margin-maximization principle that performs structural risk and it inherits the SVM generalization capacity. However, the choice of the hyperparameters (regularization parameter and kernel parameters) affects LS-SVM performance. Thus, like SVM and other classifiers, model selection is needed in order to obtain the best performance of the classifier. The classical method for choosing a good hyperparameters value is the cross-validation method based on the exhaustive search but it becomes intractable when we have many hyperparameters (for example if kernel parameters are more than two). In this paper, we propose to tune LS-SVM hyperparameters by using the empirical error criterion [2] where an empirical estimate of the generalization error is minimized through a validation set. We also use the empirical error criterion in leave-one-out cross-validation procedure where we consider the useful work of Cawley et al. [6]. Experimental results made on handwriting recognition problem demonstrate the usefulness of our method.

This paper is structured as follows. In sections 2, we give a review for LS-SVM. In section 3, we present certain of the strategies proposed to improve the sparseness of the LS-SVM. In section 4, we describe the exact leave-one-out method developed by Cawley et al. [6] and an automatic model selection for LS-SVM based on empirical error minimization. In section 5, we provide experimental results and discussion for handwriting character recognition problems. In the last section, we conclude the paper.

2 Least Squares Support Vector Machines

We first consider a binary classification problem. Let us consider a dataset $\{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$ with $x_i \in \mathcal{R}^d$ and $y_i \in \{-1, 1\}$. Nonlinear SVM classifiers

use the kernel trick to produce nonlinear boundaries. The decision function given by an SVM is :

$$f(x) = sign[w'\phi(x) + b] \quad (1)$$

where w and b are found by resolving the following optimization problem which expresses the maximization of the margin $1/\|w\|$ and the minimization of the training error :

$$\min_{w,b,\xi} \frac{1}{2} w'w + C \sum_{i=1}^{\ell} \xi_i \quad (2)$$

$$\text{subject to } : y_i[w'\phi(x) + b] \geq 1 - \xi_i \quad \forall i = 1, ..., \ell \quad (3)$$

$$\xi_i \geq 0 \quad \forall i = 1, ..., \ell \quad (4)$$

The Lagrangian of the precedent problem is :

$$\mathcal{L} = \frac{1}{2} w'w + C \sum_{i=1}^{\ell} \xi_i - \sum_{i=1}^{\ell} \alpha_i [y_i(w'\phi(x) + b) - 1 + \xi_i] - \sum_{i=1}^{\ell} \lambda_i \xi_i \quad (5)$$

with the Lagrange multipliers $\alpha_i \geq 0$ and $\lambda_i \geq 0$ for all $i = 1, ..., \ell$.

When, we apply the Lagrange differentiation theorem, we obtain :

$$f(x) = sign[\sum_{i=1}^{\ell} \alpha_i y_i k(x_i, x) + b] \quad (6)$$

with α solution of :

$$maximize : W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad (7)$$

$$subject \quad to : \sum_{i=1}^{\ell} \alpha_i y_i = 0 \quad and \quad 0 \leq \alpha_i \leq C, i = 1, ..., \ell$$

The LS-SVM is an interesting variant of SVM proposed by Suykens et al. [17, 12, 20, 25]. The standard SVM classifier of Vapnik is modified for transforming the QP problem to a linear problem. These modifications are formulated as following in LS-SVM definition :

$$\min_{w,b,\xi} \frac{1}{2} w'w + \gamma \frac{1}{2} \sum_{i=1}^{\ell} e_i^2 \quad (8)$$

$$\text{subject to } : y_i[w'\phi(x_i) + b] = 1 - e_i \quad \forall i = 1, ..., \ell \quad (9)$$

The original SVM formulation is modified at two points. First, the inequality constraints with the slack variable ξ_i expressed in (3) are replaced by the equality constraints with an error variable e_i. Second, a squared loss function is considered in the objective function. These two essential modifications simplify the problem that becomes linear.

The Lagrangian of problem (8) is expressed by :

$$\mathcal{L}(w, b, e, \alpha) = \frac{1}{2} w'w + \gamma \frac{1}{2} \sum_{i=1}^{\ell} e_i^2 - \sum_{i=1}^{\ell} \alpha_i \{y_i[w'\phi(x) + b] - 1 + e_i\} \quad (10)$$

where α_i are Lagrange multipliers.

The conditions for optimality yield

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 & \Rightarrow \quad w = \sum_{j=1}^{\ell} \alpha_j y_j \phi(x_j) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \Rightarrow \quad \sum_{j=1}^{\ell} \alpha_j y_j = 0 \\ \frac{\partial \mathcal{L}}{\partial e_j} = 0 & \Rightarrow \quad \alpha_j = \gamma e_j, \quad \forall j = 1, ..., \ell \\ \frac{\partial \mathcal{L}}{\partial \alpha_j} = 0 & \Rightarrow \quad y_j[w'\phi(x_j) + b] - 1 + e_j = 0 \quad \forall i = j, ..., \ell \end{cases}$$

We can notice that the system obtained from the Karush-Kuhn-Tucker conditions is linear. Its solution is found by solving the system of linear equations expressed in the following matrix form :

$$\begin{pmatrix} Q + \gamma^{-1} I & \vec{1}' \\ \vec{1} & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ b \end{pmatrix} = \begin{pmatrix} Y \\ 0 \end{pmatrix} \quad (11)$$

where : $Q_{ij} = k(x_i, x_j)$
$Y = (y_1, ..., y_\ell)'$
$\alpha = (\alpha_1, ..., \alpha_\ell)'$
$\vec{1} = (1, ..., 1)$

For the training of the LS-SVM, Suykens et al. [17, 12] proposed an algorithm based on the conjugate gradient technique. In [23], an improved algorithm also based on the conjugate gradient is developed with reducing the computational time. As other methods for training LS-SVM, we can cite the SMO technique adapted to find the LS-SVM solution and an algebraic method proposed by Chua in [9].

3 Sparse Least Squares Support Vector Machines

The main problem associated with LS-SVM is the lack of sparseness. Unlike SVM, almost all training points are

support vectors, $\alpha \neq 0$. Then the kernel expansions of LS-SVM are fully dense. This makes difficult to use the LS-SVM in large scale applications.

To overcome this limitation of the LS-SVM, we may suggest to train LS-SVM in primal space by using the kernel PCA strategy for reducing the dimensionality. In the literature, in order to reduce the complexity of original LS-SVM, several methods have been proposed for reducing existing support vector expansions or training the classifier with a reduced set [18, 7, 11, 24, 13].

Suykens et al. [18] proposed to prune training examples that have smaller absolute support value. In [11], a more sophisticated pruning method was proposed. In this method, the examples which introduced the smallest approximate error are removed from the training set. The two previous pruning methods kept outliers that have a negative influence on the classifier in the reduced training set. Also, the first method [18] remove from the training set the samples near the boundary and this decrease the performance of the classifier. Li and al. [24] overcome these two problems by proposing certain heuristics.

In [7, 8] a different way to obtain sparse LS-SVM is proposed. The authors suggest first to perform a selection of reference vectors forming a basis for the subspace populated by the data. Second, they modify the objective function by limiting the kernel expansions to the preselected subspace. The main difference with other method is the fact that all the training set is used, but a set of candidate support vectors is pre-selected. The objective function proposed in [7, 8] is as follows :

$$\mathcal{L}(\alpha, b) = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(z_i, z_j) + \gamma \sum_{i=1}^{\ell} \left(y_i - \sum_{j=1}^{n} \alpha_j \hat{k}(x_i, z_j) - b \right)^2 \quad (12)$$

where $n \ll \ell$ is the number of reference vectors selected z_i.

This optimization problem like the original LS-SVM can be expressed in the form of a system of linear equations :

$$(R + Z'Z) \begin{pmatrix} \alpha \\ b \end{pmatrix} = Z' \begin{pmatrix} Y \\ 0 \end{pmatrix} \quad (13)$$

with $Z = [\hat{K}, \vec{1}]$, $\hat{K}$ is a $\ell \times n$ matrix which element is $\hat{k}(x_i, z_j)$ and

$$R = \begin{pmatrix} \gamma^{-1} K & \vec{0}' \\ \vec{0} & 0 \end{pmatrix}$$

where K is a $n \times n$ matrix which element is $k(z_i, z_j)$.

There are many approaches to select a set of reference vectors [7, 3, 15], but almost all of them have heavy computational cost.

Recently, Jia and al. [13] proposed fast sparse approximation for LS-SVM (FSALS-SVM). The classifier is iteratively built by adding one basis function from a kernel-based dictionary. The basis function is selected in order to obtain the largest decrease in the LS-SVM objective function. The process is ended by using a flexible and stable criterion. The algorithm proposed in [13] is fast and has good generalization performance.

4 Model selection for LS-SVM

Model selection for LS-SVM is the task of selecting the hyperparameters which yield the best performance of the machine. The LS-SVM classifier has two types of hyperparameters: the regularization parameter γ which controls the trade-off between the training error minimization and the margin-maximization and the kernel parameters which define the given kernel function.

4.1 Exact leave-one-out

Leave-one-out is the special case of cross-validation. In k-fold cross-validation, we divide the available training data into k subsets. We train the machine k times, each time leaving out one of the subsets from training, and use only the omitted subset to compute the given error criterion. If k equals the training set size, this is called "leave-one-out" cross-validation.

Let $[\alpha^{(-i)}; b^{(-i)}]$ represent the parameters of the LS-SVM when the i^{th} sample is omitted during the leave-one-out cross-validation procedure. It is shown that :

$$\begin{pmatrix} \alpha^{(-i)} \\ b^{(-i)} \end{pmatrix} = H_{(-i)}^{-1} [y_1, ..., y_{i-1}, y_i, ..., y_\ell, 0] \quad (14)$$

where

$$H = \begin{pmatrix} Q + \gamma^{-1} I & \vec{0}' \\ \vec{0} & 0 \end{pmatrix} \quad (15)$$

and $H_{(-i)}$ is the matrix obtained when the i^{th} sample is omitted in H.

After some manipulations and using the block matrix inversion lemma (see [6] for details), the following result is obtained :

$$y_i - f^{(-i)}(x_i) = \frac{\alpha_i}{H_{ii}^{-1}} \quad (16)$$

where $f^{(-i)}(x_i)$ is the leave-one-out prediction for the i^{th} sample.

So, without computing ℓ times the parameters of the machines, it is possible to compute the criterion error as predicted residual sum-of-squares (PRESS),

$$PRESS = \sum_{i=1}^{\ell} \left[y_i - f^{(-i)}(x_i) \right]^2 \qquad (17)$$

Similar work about the sparse LS-SVM is done in [8]. So, leave-one-out cross-validation becomes a practical strategy for model selection for LS-SVM while it is intractable with SVM.

4.2 Empirical Error Minimization

In this section, we describe model selection for the LS-SVM using the empirical error criterion developed in [2, 1] for tuning the kernel parameters of the original SVM classifier.

Let us define $t_i = (y_i + 1)/2$; the empirical error is given by the following expression:

$$E_i = |t_i - \hat{p}_i| \qquad (18)$$

where $\hat{p}_i$ is the estimated posterior probability corresponding to the data example x_i.

The estimated posterior probability is determinate by :

$$\hat{p}_i = \frac{1}{1 + exp(A.f_i + B)} \qquad (19)$$

where $f_i = f(x_i)$ and the parameters A and B are fitted after minimizing the cross-entropy error [5] as Platt proposed in [14]. In this paper, we use fixed values for A and B, because we need to keep the continuity of the empirical error expression for each validation sample at each iteration.

We assume that the kernel function depends on one or several parameters, encoded within the vector $\theta = (\theta_1, \ldots, \theta_n)$ including the hyperparameter γ. The optimization of these parameters is performed by a gradient descent minimization algorithm [4] where the objective function is $E = \sum E_i$. However, sometimes our problem is not convex, so we use many starting points to overcome this situation. we can also use the simple function *fminsearch* implemented in Matlab with different starting points.

Despite the empirical error criterion was first developed for SVM, its use with LS-SVM is not just a direct mapping. So, the derivative computing for LS-SVM needs a careful analysis of its model.

1. *Initialize the learning rate*
2. *Initialize the hyperparameters*
3. *Repeat until convergence*
  3.1 *Train the LS-SVM*
  3.2 *Compute the gradient of error*
  3.3 *Estimate the learning rate*
  3.4 *Update the hyperparameters*

Figure 1. Descent gradient algorithm for the LS-SVM hyperparameters optimization

The derivative of the empirical error with respect to θ is evaluated using the validation dataset. Let us assume N the size of the validation dataset, then :

$$\frac{\partial E}{\partial \theta} = \frac{\partial}{\partial \theta} \left(\frac{1}{N} \sum_{i=1}^{N} E_i \right) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial E_i}{\partial \theta} \qquad (20)$$

E_i is expressed in terms of the estimated posterior probability, which depends on the output f_i of the LS-SVM. Therefore, we can write :

$$\frac{\partial E_i}{\partial \theta} = \frac{\partial E_i}{\partial \hat{p}_i} * \frac{\partial \hat{p}_i}{\partial f_i} * \frac{\partial f_i}{\partial \theta} \qquad (21)$$

According to equation (18), we have :

$$\frac{\partial E_i}{\partial \hat{p}_i} = \frac{\partial |t_i - \hat{p}_i|}{\partial \hat{p}_i} = -y_i \qquad (22)$$

Considering equation (19), giving the expression of the estimated posterior probability, the second part of the gradient is :

$$\frac{\partial \hat{p}_i}{\partial f_i} = -A\hat{p}_i(1 - \hat{p}_i) \qquad (23)$$

For evaluating the last part of the gradient, we consider the expression of the LS-SVM output.

$$\begin{aligned} \frac{\partial f_i}{\partial \theta} &= \frac{\partial}{\partial \theta} \left(\sum_{j=1}^{\ell} \alpha_j y_j k(x_j, x_i) + b \right) \qquad (24) \\ &= \sum_{j=1}^{\ell} y_j \left[\frac{\partial k(x_j, x_i)}{\partial \theta} \alpha_j + \frac{\partial \alpha_j}{\partial \theta} k(x_j, x_i) \right] + \frac{\partial b}{\partial \theta} \end{aligned}$$

In equation (24), the evaluation of the derivative $\frac{\partial k(x_j, x_i)}{\partial \theta}$ is easy and depends on the type of the kernel to be chosen. But, for estimating the derivative $\frac{\partial \alpha_j}{\partial \theta}$, we need the expression of the terms α_j with respect to θ. For this, we use equation (11) :

$\tilde{\alpha} = H^{-1}\tilde{Y}$ where $\tilde{\alpha} = (\alpha; b)$ and $\tilde{Y} = (Y; 0)$
Then,

$$\begin{aligned} \frac{\partial \tilde{\alpha}}{\partial \theta} &= \frac{\partial H^{-1}}{\partial \theta}\tilde{Y} + H^{-1}\frac{\partial \tilde{Y}}{\partial \theta} \\ &= \frac{\partial H^{-1}}{\partial \theta}\tilde{Y} \end{aligned}$$

For computing the components of the $\frac{\partial H^{-1}}{\partial \theta}$, we use the matrix relation proposed in [4].

$$\begin{aligned} \frac{\partial \tilde{\alpha}}{\partial \theta} &= -H^{-1}\frac{\partial H}{\partial \theta}H^{-1}\tilde{Y} \\ &= -H^{-1}\frac{\partial H}{\partial \theta}\tilde{\alpha} \end{aligned} \quad (25)$$

It is also possible to minimize the empirical error through the leave-one-out cross-validation procedure. Considering equation (16), the leave-one-out prediction for the i^{th} sample is expressed by :

$$f^{(-i)}(x_i) = y_i - \frac{\alpha_i}{H^{-1}_{ii}} \quad (26)$$

Then, using the previous equality, we can compute the empirical error for each training sample by using the leave-one-out cross-validation procedure.

5 Experiments

We tested our model selection algorithm for the LS-SVM on a handwritten recognition problem with USPS database.

5.1 USPS database

USPS is the well known US Postal Service handwritten digits recognition corpus. The digits are represented by normalized grey scale images of size 16×16. The learning dataset contains 7291 samples (5291 for training and 2000 for validation) while the testing dataset consists of 2007 other samples.

5.2 Experimental setup

We have a multi-class problem with 10 classes. Then we trained 10 machines by using the *one-against-all* strategy. We tested the original LS-SVM and the Sparse LS-SVM proposed in [13] and whose code is available at http://see.xidian.edu.cn/graduate/lfbo. For each classifier, the hyperparameters are optimized by using $PRESS$ criterion with exact leave-one-out (LOO) strategy and by using empirical error criterion. Finally, we compute the mean of the hyperparameters with the ten values obtained from each classifier.

Table 1. Error rate in percent obtained on test set with original LS-SVM and the Sparse LS-SVM. We present the results with four different model selection techniques : empirical error criterion with validation set, LOO with PRESS criterion [6], LOO with empirical error criterion and the classical grid search method done in [13].

	Original LS-SVM	Sparse LS-SVM
Empirical Error criterion	4.63	4.18
LOO with PRESS criterion	4.48	4.43
LOO with Empirical Error	4.53	4.03
Grid Search method	4.43	4.14

5.3 Results and Discussion

The results obtained are shown in Table 1, where we report the results obtained with different model selection techniques. Our results, obtained by Empirical Error minimization, are similar to those obtained by classical grid search method, while the latter is quite costly in term of computing time and becomes intractable when the hyperparameters are more than two. Note that the $PRESS$ statistic which is best suited to regression problems [8, 6], is less adequate for LS-SVM classifier.

Also, we remark that the Sparse LS-SVM achieved good generalization in comparison with the original LS-SVM. The classifier obtained with the Sparse LS-SVM algorithm has fewer support vectors, which reduces the complexity of the machine. Thus, the regularization of the classifier is reinforced. In figure 2, we plot the test error rate and the number of support vectors according to the model selection method with the Sparse LS-SVM. We point out that the LOO procedure with Empirical Error criterion give the best result in term of the machine complexity and generalization confirming our approach.

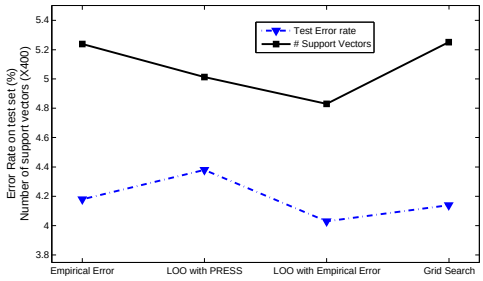

Figure 2. Figure shows the test error rate and the number of support vectors according to the model selection method with the Sparse LS-SVM.

In table II, we compare the Sparse LS-SVM with SVM where we used Empirical Error criterion for performing model selection. This experiment shows that the Sparse LS-SVM is a good alternative for SVM. The results point out that the Sparse LS-SVM achieves good performance in term of accuracy, complexity and sparseness. Moreover, the classifier yield the best generalization performance on this problem.

Table 2. Comparison between SVM and Sparse LS-SVM : we perform the training of SVM with Joachims' algorithm called SVMlight.

	Original SVM	Sparse LS-SVM
Training time	$45sec$	$44sec$
Number of support vectors	5069	1932
Testing time	$5.32sec$	$2.21sec$
Test error	4.33	4.03

6 Conclusion

In this paper, we proposed a model selection for LS-SVM which is a variant of the popular SVM. We performed the model selection by using empirical error criterion minimized on a validation set and through leave-one-out procedure. We applied our algorithms on a handwriting recognition problem which gave promising results. Comparing with SVM, the Sparse LS-SVM classifier, empowered by model selection based on empirical error criterion, achieved higher performance. We may conclude that the Sparse LS-SVM with model selection would be an interesting alternative for SVM in pattern recognition systems.

Acknowledgments

The authors would like to thank the NSERC of Canada for their financial support.

References

[1] M. M. Adankon and M. Cheriet, "Optimizing Resources in Model Selection for Support Vector Machines", *Pattern Recognition, in Computer Science*, 40(3):953–963, 2007.

[2] N. E. Ayat, M. Cheriet and C. Suen, "Automatic Model Selection for the Optimization of the SVM kernels", *Pattern Recognition, in Computer Science*, 38(10):1733–1745, 2005.

[3] G. Baudat and F. Anouar., "Feature vector selection and projection using kernels", *Neurocomputing*, 55:31–38, 2003.

[4] Y. Bengio, "Gradient-Based Optimization of Hyper-Parameters", *Neural Computation*, 12(8):1889–1900, 2000.

[5] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, Great Britain, 1995.

[6] G. Cawley, "Leave-one-out Cross-validation Based Model Selection Criteria for Weighted LS-SVMs", *proceedings IJCNN 2006, Vancouver, Canada*, July 2006.

[7] G. C. Cawley and N. L. C. Talbot, "Improved sparse least-squares support vector machines", *Neurocomputing*, 48:1025–1031, 2002.

[8] G. C. Cawley and N. L. C. Talbot, "Fast exact leave-one-out cross-validation of sparse least-squares support vector machines", *Neural Networks*, 17:1467–1475, 2004.

[9] K. S. Chua, "Efficient computations for large least square support vector machine classifiers", *Pattern Recognition Letters*, 24:75–80, 2003.

[10] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.

[11] B. J. de Kruif and T. J. deVries, "Pruning error minimization in least squares support vector machines", *IEEE Transactions on Neural Networks*, 14:696–702, 2003.

[12] J. D. B. B. D. M. J. A. K. Suykens, T. Van Gestel and J. Vandewalle, *Least Squares Support Vector Machines.*, World Scientific, Singapore, 2002.

[13] L. B. Licheng Jiao and L. Wang, "Fast sparse approximation for least square support vector machine", *IEEE Transactions on Neural Networks*, 18(3):685–697, 2007.

[14] J. Platt"Probabilistic outputs for support vector machines and comparison to regularized likelihood methods", A. Smola, P. Bartlett, B. Schoelkopf and D. Schuurmans, editors, *Advances in Large Margin Classiers*, pp 61–74. 2000.

[15] B. Scholkopf and A. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.

[16] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

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

[18] L. L. Suykens, J. A. K. and J. Vandewalle, "Sparse Least squares support vector machine classifiers", *European Symposium of Artificial Neural Networks*, 2000.

[19] B. B. V. S. V. J. D. G. D. M. B. V. J. Van Gestel T., Suykens J., "Benchmarking Least Squares Support Vector Machine Classifiers", Technical report, Internal Report 00-37, ESAT-SISTA, K.U.Leuven (Leuven, Belgium) 2000. Accepted for publication in Machine Learning.

[20] L. G. L. A. D. M. B. V. J. Van Gestel T., Suykens J., "Bayesian Framework for Least Squares Support Vector Machine Classifiers, Gaussian Processes and Kernel Fisher Discriminant Analysis", *Neural Computation*, 15(5):1115–1148, 2002.

[21] V. Vapnik, *Statistical learning theory*, John Wiley and Sons, New York, 1998.

[22] V. N. Vapnik, "Principles of risk Minimization for learning theory", *Adavances in Neural Information Processing Systems 4, Morgan Kaufman, San Mateo, CA*, pp 831–838, 1992.

[23] C. J. O. W. Chu and S. S. Keerthi, "An improved conjugate gradient scheme to the solution of least squares SVM", *IEEE Transactions on Neural Networks*, 16(2):498–501, 2005.

[24] C. l. Yuangui Li and W. Zhang, "Improved sparse least-squares support vector machine classifiers", *Neurocomputing*, 69:1655–1658, 2006.

[25] S.-F. Zheng, "Least Square Support Vector Machine and its Bayesian interpretation", Technical report, Technical Report, June 2004.