Automatic Model Selection for the optimization of SVM Kernels

N.E. Ayat [1,2] and M. Cheriet [*1] and C.Y. Suen [2]

[1] *LIVIA, ÉTS, 1100, Notre Dame west St., Montreal, H3C 1K3, Canada*

[2] *CENPARMI, Concordia University, 1455 de Maisonneuve Blvd West, Montreal, H3G 1M8, Canada*

Emails:nedjem@livia.etsmtl.ca,Mohamed.Cheriet@etsmtl.ca,suen@cenparmi.concordia.ca

Abstract

This approach aims to optimize the kernel parameters and to efficiently reduce the number of support vectors, so that the generalization error can be reduced drastically. The proposed methodology suggests the use of a new model selection criterion based on the estimation of the probability of error of the SVM classifier. For comparison, we considered two more model selection criteria: GACV ('Generalized Approximate Cross-Validation') and VC ('Vapnik-Chernovenkis') dimension. These criteria are algebraic estimates of upper bounds of the expected error. For the former, we also propose a new minimization scheme. The experiments conducted on a bi-class problem show that we can adequately choose the SVM hyper-parameters using the empirical error criterion. Moreover, it turns out that the criterion produces a less complex model with fewer support vectors. For multi-class data, the optimization strategy is adapted to the one-against-one data partitioning. The approach is then evaluated on images of handwritten digits from the USPS database.

Key words: Model selection, SVM, kernel, empirical error, VC, GACV.

1 Introduction

In 1979, Vapnik introduced a new induction principle [1] known as structural risk minimization. Unlike the empiric risk, this principle can avoid over-fitting the data at the convergence of the training. The SVM is a structural risk based learning machine. It may be used either to classify or to predict some arbitrary patterns from a set of labeled data. During the last decade, many pattern recognition problems have been tackled using support vector machines. For instance, Cortes et al., Scholkopf et al. and Burges et al. applied the SVM for optical character recognition [2–4], whereas Blanz et al. used it to recognize scenes of two dimensional object views [5]. As well, Schmidt et al. used this classifier as a part of a speaker identification system [6] and Osuna et al. studied its performance on a face recognition task [7]. Many other applications such as gender classification, data mining, stock action prediction, etc have been tackled using the SVM. In many cases, it outperforms most state of the art classifiers. The book of Cristianini et al. in [8] gives a nice introduction of the SVM theory and its applications.

The SVM training produces a discriminant function that minimizes the training error while maximizing the margin separating the data classes. The maximization of the margin is an implicit regularization process which reduces the classifier complexity. In fact, its effect is to penalize the model parameters (multipliers α_i) in the same way as the weight-decay penalizing the strong weights of an a neural network. This enables the training process to yield a sparse model with few parameters along with few prototypes known as support vectors (SV). Support vectors are a subset from the training data set which define the decision frontier. At most, there are as many parameters of the

model as training examples. Hence, this number does not depend on the dimension of the input space (number of features). This characteristic makes the SVM more robust against the well known *dimensionality dilemma*, especially for small data sets.

However, the SVM embeds tuning parameters that control the training setting such as the kernel parameters and the trade off variable C. These parameters have a regularization effect on the cost function minimized during the training process. As well, since their values are not trained, these variables may diminish the overall performance of the classifier if not well chosen. Also, there is no systematic methodology that allows for an a priori estimation of their optimal values. In fact, given a classification task, picking the best values for these variables is a non trivial model selection problem that needs either an exhaustive search over the space of hyper-parameters or an optimization procedure that explores only a finite subset of the possible values. Until now, most SVM practitioners select these parameters empirically by trying a finite number of values and keeping those that provide the least testing error. This procedure requires a grid search over the space of parameter values and needs to locate the interval of feasible solution and a suitable sampling step. This is a tricky task since a suitable sampling step varies from kernel to kernel and the grid interval may not be easy to locate without prior knowledge of the problem. Moreover, when there are more than two hyper-parameters, the manual model selection may become intractable.

Recently, Chapelle et al. proposed an analytical criterion that is a proxy of the VC dimension known as the *radius-margin bound* [9]. This criterion is a quadratic function of the support vector multipliers. The authors also proposed an automatic minimization algorithm for it. However, the radius-margin

bound is useful only for the L2 SVM variant and is based on the estimation of the smallest enclosing hyper-sphere of the data. Its computation needs to resolve an extra quadratic programming objective which is hard to express and computationally as expensive as the training process itself. Furthermore, this criterion assumes that the data classes are separable or almost so [10].

Besides, the Generalized Approximate Cross-Validation (GACV) criterion was proposed by Wahba as an upper bound of the generalized Kullback Lebleir distance between the expected decision function and the current one, which turns out to be an upper bound of the misclassification error [11]. Unlike the VC dimension, GACV is a linear function of the support vector parameters α_i, and thus easier to compute. Two versions of the criterion are available for both L1 and L2 variants.

In this paper, we introduce a new SVM model selection methodology based on the minimization of an empirical error estimate. The latter is an approximation of the probability of error computed through an independent data set which is used to minimize the generalization error and the complexity of the classifier. The computation of the empirical error is straightforward in comparison with the Radius-Margin and GACV criteria and allows a closer estimate to the expected generalization error.

For comparison, we also considered two algebraic criteria, namely the radius-margin bound and GACV, for which we also give its minimization algorithm. A comparison with the *radius-margin bound* and GACV inicates the empirical error provides a closer expected error estimate, and is much more efficient to compute suitable values for the hyper-parameters.

Furthermore, we undertook the optimization of SVM kernels when multiple

data classes are available. So we propose a local optimization scheme based on a data partitioning w.r.t. the 'one-against-one' strategy of learning. The methodology optimizes a set of pairwise classifiers and yields $M(M-1)/2$ different kernel settings for an M-class problem.

In section 2, we present a brief review of the support vector theory. In section 3, we motivate the model selection problem by highlighting the relation between the capacity and the number of parameters of a classifier from a general point of view. Furthermore, we review a few pertinent algebraic criteria found in the literature. In section 4, we introduce the empirical error criterion and explain its relation to the well known robust error [12]. Its minimization procedure is detailed as well. In section 5, we present the GACV criterion for which we give the minimization algorithm in the appendix. In section 6, we review the VC dim criterion as proposed in [9]. In section 7, we compare the minimization of the empirical error, VC dim and GACV experimentally. The results are established on a synthetic two-class problem and prove the efficiency of the proposed criterion to reduce the classification error while minimizing the classifier complexity. On a digit recognition task, promising results are reported on USPS database. Finally, in section 8, we give a summary of the work and some concluding remarks.

2 Review of Support Vector Classifiers Theory

Let us have a data set $\{x_i, y_i\}, i = 1, \ldots, l$ of examples i.i.d., where $y_i \in \{-1, 1\}$ and $x_i \in \mathrm{R}^d$ where x_i is an arbitrary data point and y_i its corresponding bipolar label. Let us also define a linear decision surface by the equation $f(x) = w.x + b = 0$. The original formulation of the SVM algorithm seeks a linear decision

surface which maximizes the margin between the closest positive and negative examples. This may be achieved through the minimization of the penalty term $\|w\|^2/2$ [13]. It yields $w = \sum_i \alpha_i y_i x_i$ with the constraints $\sum_i \alpha_i y_i = 0$, and $0 \leq \alpha_i \leq C, \forall i = 1, \ldots, l$ where C is a regularization variable called trade-off parameter. The parameters α_i can be found after the following quadratic optimization problem is maximized:

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j \tag{1}$$

The data examples whose corresponding α_i values are not zero are called support vectors.

Instead of considering the input space, we may consider a given augmented space by replacing the inner product of equation 1 by the dot product $K(x_i, x_j) = \phi(x_i).\phi(x_j)$ which yields

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j),$$

where functions $K(x, y) = \phi(x).\phi(y)$ represent semi-definite kernels [14]. Some of the classical SVM kernels are reported in table A.1.

The resulting decision frontier is $f(x) = \sum_i \alpha_i y_i K(x_i, x) + b$, where x_i, α_i and b represent respectively the i^{th} support vector, its corresponding multiplier and the hyperplane bias. For non separable classes, the L2 SVM variant minimizes the functional $\frac{1}{C}\frac{\|w\|^2}{2} + \frac{1}{2}\sum_i \xi_i^2$, which leads to the maximization of

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \widetilde{K}(x_i, x_j), \tag{2}$$

with

$$\widetilde{K}(x_i, x_j) = K(x_i, x_j) + \frac{1}{C} \quad if \;\; i = j$$
$$\widetilde{K}(x_i, x_j) = K(x_i, x_j) \quad if \;\; i \neq j.$$

under the constraints $\sum_i \alpha_i y_i = 0$, and $\alpha_i \geq 0, \forall i = 1, \ldots, l$. Apart from that, it is worth noting that the kernel trick is not only relevant to support vector classifiers. Indeed it has allowed many pattern recognition tasks to be expressed in terms of kernels. For example, Baudat et al. in [15] presented the kernel based version of the Fished discriminant analysis. Schölkopf et al. in [16,17] introduced the Nonlinear Principal Component Analysis based on the kernel trick. Kim and Oommen in [18,19] proposed optimized kernel-based nonlinear subspace classifiers for pattern recognition. For further details regarding the topic, the book by Shawe-Taylor et al. [20] constitutes an excellent reference to refer to.

3 Model selection: related work

Model selection gained a particular interest when neural networks established effective classification models on many pattern recognition problems in the late eighties. For example, a lot of research has been done on methods to choose an adequate architecture for an MLP to lower its generalization error. In the seventies, Vapnik's work on the structural risk minimization established a strong mathematical link between the notion of the capacity of a classifier and its generalization error [1]. In the case of an MLP for example, the latter is also a function of the size of the training set, the activation function of the hidden units and the number of training epochs. An incorrect choice of one of these elements, may lead to a high generalization error. The capacity of a learning machine determines its ability to learn the examples of the training

set and is strongly dependent on the number of parameters of the model. Thus, a classifier with a huge number of parameters (great capacity) over-fits the training data and the solution is sensitive to noise. As a result, the performance on the testing set is poor. The error of variance in such a case is high. On the other hand, an insufficient number of parameters (small capacity) gives a solution which does not fit the training data. Then, the bias error is high. This observation depicts the well known *'Bias-Variance'* dilemma derived from the Occam [1] razor principle.

In practice, it is necessary to search for a compromise between simplicity and complexity of the classifier. In regularization theory, it is possible to consider a penalization term which controls the complexity of the classifier by one or more hyper-parameters [21]. The weight-decay technique developed by Hinton [22], for example, is a typical method that reduces the complexity of a Multi Layer Perceptron by penalizing strong weights with the addition of an extra term to the typical quadratic cost function used for its training. It has been proven that such a technique is very robust to over-fitting phenomena. This penalizing term may also be interpreted as an approximation of the logarithm of the prior probability of the weights in the model [23], whereas the likelihood of the data in the non regularized model reflects the degree of fidelity to training points. In fact, many approaches may be useful in tackling the model selection problem and most of them may be linked to either the regularization theory [24,21,25] or the Bayesian learning framework [26,27].

In a support vector classifier, the problem is very similar, although the notion

[1] Occam's razor is a logical principle attributed to the medieval philosopher William of Occam (1285, England).

of complexity needs further explanation. Given a classification problem, the SV number and so the number of parameters, are strongly related to the kernel setting, which can be seen as a way to cast our prior knowledge of the problem. Thus, if we expect a highly nonlinear decision function, we better use a compact kernel [2], while a spread kernel is appropriate for an almost linear decision frontier.

Many model selection criteria are useful in choosing suitable values for the hyper-parameters. Ideally, the true risk is the best criterion to be used. But the data distribution $P(x, y)$ is not known. Instead, there are alternative criteria that represent either approximations or upper-bounds to the misclassification error. For example, Joachim's bound [28], Jaakola-Haussler bound [29], Opper-Winther bound [30] and covering numbers bound [31,32] were proposed as pessimistic estimates of the true risk. Unfortunately many of them are either very conservative or discrete, which make their utilization intractable or inefficient. More recently, few contrasting approaches have addressed the model selection goal as a kernel matrix learning problem, bypassing the need for an a priori chosen analytical kernel [33]. Moreover, optimizing kernel parameters for the Kernel PCA has been addressed as done in [34].

4 The empirical error criterion

It is possible to use a subset of the data to approximate the expected risk of an SVM [35]. Given an independent data set, the number of classification

[2] Stands for a kernel with reduced spread and not with compact support since this may not be a Mercer kernel.

errors is:

$$T = \frac{1}{N}\sum_i [-y_i f(x_i)]_*, \tag{3}$$

where N is the size of the data set and (x_i, y_i) represents the data entry and its corresponding bipolar label. This quantity reflects the generalization power of the classifier and may be a powerful model selection criterion. Within a gradient descent framework, automatic model selection using the criterion of equation 3, needs to estimate the derivative of the operator $[\,]_*$ which is not a continuous function.

Let us assume that t_i is the corresponding binary label of x_i, so $t_i = 0$ corresponds to $y_i = -1$ and $t_i = 1$ corresponds to $y_i = 1$.Then we have $t_i = \frac{y_i+1}{2}$.

Let us now define $z_i = ([f_i] + 1)/2$ as the binary predicted membership of the data point x_i ($f_i = f(x_i)$). The operator $[\,]$ represents the signum function defined as

$$\begin{aligned} [x] &= +1 \quad if\ x > 0 \\ [x] &= -1 \quad if\ x \leq 0. \end{aligned}$$

Given a data example x_i, we may hope its corresponding posterior probability p_i will be equal to zero if $t_i = 0$ and equal to one if $t_i = 1$. It is thus possible to characterize the probability of error E_i of the data point x_i as

$$E_i = P(y_i \neq z_i) = \begin{cases} p_i & if\ t_i = 0 \\ 1 - p_i & if\ t_i = 1 \end{cases} \tag{4}$$

Given a data set of size N, the mean probability of error is $E = \frac{1}{N}\sum_{i=1}^{N} E_i$. If we hypothesize that p_i represents the true posterior probability and N is

infinite, the expression of E given above is the natural classification error.

The residual probability of error E_i can be written as $|t_i - \hat{p}_i|$, where $\hat{p}_i$ is an estimation of the true posterior probability p_i. This formula is a particular case of the Minkowski-R error for $R = 1$ and defined $|t_i - \hat{p}_i|^R$, where R is an integer greater than or equal to one. For $R = 1$, the error is of norm $L1$, and it is demonstrated to be equivalent to the minus of the logarithm of the data likelihood given a Laplacian noise ϵ of the form $p(\epsilon) = a \exp(-\beta|\epsilon|)$, where a, β are respectively a normalization constant and the inverse of the variance of the noise distribution. Hanson et al. mentioned that the L1 Minkowski error (sometimes referred to as the robust error) penalizes the errors of the marginal observations, presenting relatively higher error values [12].

4.1 Estimation of the posterior probability

The SVM does not provide a probability measure. Given a raw score value, the estimation of the probability is a post processing step which might be complex depending on its procedure. We propose to use a two-parameter logistic function proposed by Platt [36] of the form

$$P(y = 1|f) = \frac{1}{1 + \exp(Af + B)}. \tag{5}$$

This model is easy and requires a nonlinear optimization for the estimation of the couple (A, B). The values of A and B are chosen so the cross-entropy error through the validation data points is minimized by using a variant of Newton's algorithm [37].

4.2 Algorithm

Let us represent a set of n hyper-parameters by the vector $\underline{\theta}$ of dimension n. Knowing that the gradient of the error w.r.t. $\underline{\theta}$ vanishes at a local minima, its expression near this point may be written as

$$\nabla E(\underline{\theta}) = \frac{\partial E}{\partial \underline{\theta}} = \left.\frac{\partial E}{\partial \underline{\theta}}\right|_{\underline{\alpha}=\underline{\alpha}_0} + \left.\frac{\partial E}{\partial \underline{\alpha}}\right|_{\underline{\theta}=\underline{\theta}_0} . \frac{\partial \underline{\alpha}}{\partial \underline{\theta}}, \tag{6}$$

where $\underline{\alpha} = (\alpha_1, ..., \alpha_k)$ is the set of parameters the model embeds, k is the number of support vectors and $\underline{\alpha}_0$ and $\underline{\theta}_0$ are respectively the current parameters and hyper-parameters values. The gradient of the error w.r.t. $\underline{\theta}$ may also be written as an n-dimensional vector of values that are the partial derivatives of the error w.r.t. each hyper-parameter, as $\nabla E(\underline{\theta}) = \frac{\partial E}{\partial \underline{\theta}} = \left(\frac{\partial}{\partial \theta_1} E(\underline{\alpha}, \underline{\theta}), \ldots, \frac{\partial}{\partial \theta_n} E(\underline{\alpha}, \underline{\theta})\right)$.

The gradient $\left.\frac{\partial E}{\partial \underline{\theta}}\right|_{\underline{\alpha}=\underline{\alpha}_0}$ may be written

$$\left.\frac{\partial E}{\partial \underline{\theta}}\right|_{\underline{\alpha}=\underline{\alpha}_0} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \hat{p}_i} E_i(\underline{\alpha}_0, \underline{\theta}) \frac{\partial \hat{p}_i}{\partial f_i} \frac{\partial f_i(\underline{\alpha}_0)}{\partial \underline{\theta}}, \tag{7}$$

and $\frac{\partial f_i(\underline{\alpha}_0)}{\partial \underline{\theta}} = \sum_{j=1}^{k} \alpha_j y_j \frac{\partial K_{\underline{\theta}}(x_j, x_i)}{\partial \underline{\theta}}$, where x_j, y_j and α_j represent respectively the j^{th} support vector, its label and its corresponding multiplier.

Moreover, in equation 6, the gradient of the error w.r.t. the model parameters $\underline{\alpha}$ is a k-dimensional vector given by

$$\nabla E(\underline{\alpha}) = \left.\frac{\partial E}{\partial \underline{\alpha}}\right|_{\underline{\theta}=\underline{\theta}_0} = \left(\frac{\partial}{\partial \alpha_1} E(\underline{\alpha}, \underline{\theta}_0), \ldots, \frac{\partial}{\partial \alpha_k} E(\underline{\alpha}, \underline{\theta}_0)\right). \tag{8}$$

On the right hand-side term of equation 6, $\frac{\partial \underline{\alpha}}{\partial \underline{\theta}}$ is a $k \times n$ matrix of partial derivatives of $\underline{\alpha}$ w.r.t. $\underline{\theta}$, and is given by

$$\nabla \underline{\alpha}(\underline{\theta}) = \frac{\partial \underline{\alpha}}{\partial \underline{\theta}} = \left(\frac{\partial \underline{\alpha}}{\partial \theta_1}, \ldots, \frac{\partial \underline{\alpha}}{\partial \theta_n} \right) = \begin{pmatrix} \frac{\partial \alpha_1}{\partial \theta_1} & \cdots & \frac{\partial \alpha_1}{\partial \theta_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \alpha_k}{\partial \theta_1} & \cdots & \frac{\partial \alpha_k}{\partial \theta_n} \end{pmatrix}.$$

The gradient $\frac{\partial}{\partial \alpha_j} E(\underline{\alpha}, \underline{\theta}_0)$ of the empirical error w.r.t. the multiplier α_j is the summation of the partial derivatives of the N residual errors E_i given by

$$\frac{\partial}{\partial \alpha_j} E(\underline{\alpha}, \underline{\theta}_0) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \hat{p}_i} E_i(\underline{\alpha}, \underline{\theta}_0) \frac{\partial \hat{p}_i}{\partial f_i} \frac{\partial f_i}{\partial \alpha_j}. \tag{9}$$

where $\frac{\partial E_i}{\partial \hat{p}_i} = \frac{\partial |t_i - \hat{p}_i|}{\partial \hat{p}_i} = \begin{cases} +1 \ if \ t_i - \hat{p}_i < 0 \ (t_i = 0) \\ -1 \ if \ t_i - \hat{p}_i > 0 \ (t_i = 1) \end{cases}$

On the other hand, the gradient of the posterior probability $\hat{p}_i$ w.r.t. the raw SVM output f_i is $\frac{\partial \hat{p}_i}{\partial f_i} = -A\hat{p}_i^2 \exp(Af_i + B) = -A\hat{p}_i(1 - \hat{p}_i)$, whereas the expression of the gradient of f_i w.r.t. α_j for the current hyper-parameters is

$$\left. \frac{\partial f_i}{\partial \alpha_j} \right|_{\underline{\theta} = \underline{\theta}_0} = y_j K_{\underline{\theta}_0}(x_j, x_i), \tag{10}$$

where $y_j = \pm 1$ is the bipolar label of the observation x_j.

It remains to estimate the matrix $\nabla \underline{\alpha}(\underline{\theta})$. We may include the bias b into the parameter vector $\underline{\alpha}$ as $\underline{\alpha} = (\alpha_1, ..., \alpha_k, b)$ and include the following expansion used by Chapelle et al. [9] for the span bound computation given by :

$$\nabla \underline{\alpha}(\underline{\theta}) = \frac{\partial \underline{\alpha}}{\partial \theta} = -H^{-1} \frac{\partial H}{\partial \theta} \underline{\alpha}^T, \tag{11}$$

where

$$H = \begin{pmatrix} K^Y & Y \\ Y^T & 0 \end{pmatrix}. \tag{12}$$

In equation 12, K^Y represents the Hessian matrix of the SVM objective. Its components K^Y_{ij} equal to $y_i y_j K(x_i, x_j)$. Y is a $k \times 1$ vector of support vector labels y_i, and Y^T is its transpose. We shall refer to H as the augmented Hessian matrix. Algorithm 1 reports the the empirical error minimization procedure.

Algorithm 1 `Empirical error minimization`

Inputs :

- Training data set

- Validation data set

- Initial kernel parameters $\underline{\theta}_0$

- η : the learning factor needed for the update computation

Output : Final kernel parameters $\underline{\theta}$.

1: $\underline{\theta} \leftarrow \underline{\theta}_0$,
2: **While** not converging **Do**
3: perform training with $\underline{\theta}$,
4: estimate the parameters A and B of the activation function,
5: estimate the probability of error E,
6: compute the gradient of the error $\frac{\partial}{\partial \underline{\theta}} E(\underline{\alpha}, \underline{\theta})$,
7: update $\underline{\theta}$ as: $\Delta\underline{\theta} = -\eta \frac{\partial}{\partial \underline{\theta}} E(\underline{\alpha}, \underline{\theta})$,
8: **End While**
9: **Return** final kernel parameters $\underline{\theta}$.

5 The GACV criterion

Wahba [11] proposed an upper bound of the *Generalized Comparative Kullback Leibler distance*, which is a pessimistic estimate of the misclassification error. The estimate is called *Generalized Approximate Cross Validation* error and is an in-sample criterion computed on the training data. The approximation is established under the hypothesis that the set of support vectors is a small fraction of the whole training data.

There are two variants for the GACV criterion, corresponding respectively to the L1 SVM and the L2 SVM. We shall refer to the first $GACV_{L1}$, which is written as

$$GACV_{L1}(\underline{\alpha}, \underline{\theta}) = \frac{1}{l}\left(\sum_{i=1}^{l}\xi_i + 2\sum_{y_i f_i<-1}\alpha_i K_{ii} + \sum_{y_i f_i\in[-1,1]}\alpha_i K_{ii}\right);$$

having a general form as the following $\frac{1}{l}\left(\sum_{i=1}^{sv}\xi_i + c(i)\alpha_i\right)$. For the L2 SVM, we consider $GACV_{L2}$ which is expressed as

$$GACV_{L2}(\underline{\alpha}, \underline{\theta}) = \frac{1}{l}\left(\sum_{i=1}^{l}\frac{\xi_i^2}{2} + 2\sum_{y_i f_i<-1}\alpha_i K_{ii} + \sum_{y_i f_i\in[-1,1]}\alpha_i K_{ii}\right)$$

having the following general form $\frac{1}{l}\left(\sum_{i=1}^{sv}\frac{1}{2}\xi_i^2 + c(i)\alpha_i\right)$, where ξ_i is the slack variable defined as

$$\begin{aligned} \xi_i &= 0 \; if \; y_i f_i > 1 \\ \xi_i &= 1 - y_i f_i \, if \; y_i f_i \leq 1, \end{aligned}$$

and K_{ii} is the kernel value $K(x_i, x_i)$.

GACV is an algebraic model selection criterion which takes into account the training error through the variables ξ_i and the complexity of the classifier through its parameters α_i. The criterion exploits two categories of support

vectors. The first one includes the support vectors situated on the margin $[+1, -1]$, whereas the second one corresponds to the set of training errors located beyond the margin. Their corresponding multipliers α_i are penalized twice compared to those of the first category.

For an RBF kernel, $K_{ii} = 1, \forall i$ and the expression of $GACV_{L1}$ becomes $GACV(\underline{\alpha}, \underline{\theta}) = \frac{1}{l}\left(\sum_{i=1}^{l} \xi_i + 2\sum_{y_i f_i < -1} \alpha_i + \sum_{y_i f_i \in [-1,1]} \alpha_i\right)$. In the strictly separable case, the minimization of the criterion leads to reduced values of α_i, thus lowering the number of support vectors (the minimization procedure is given in the appendix A).

6 The VC dimension criterion

The VC dim criterion is strictly proportional to the radius-margin bound given by $T = R^2\|w\|^2$ [9]. The radius R of the smallest enclosing hyper-sphere of the training data is found after the following quadratic optimization problem is resolved:

$$\begin{aligned} &maximize\, R^2(\underline{\beta}) = \max_{\underline{\beta}} \sum_{i=1}^{l} \beta_i K(x_i, x_i) - \sum_{i=1}^{l}\sum_{j=1}^{l} \beta_i \beta_j K(x_i, x_j) \\ &subject\ to: \qquad \sum_{i=1}^{l} \beta_i = 1, \\ &\qquad\qquad\qquad \forall i\ \beta_i \geq 0, \end{aligned}$$

where K is the current SVM kernel. For a dot product kernel, the hyper-sphere is computed in the input space. The solution to this objective function is the set of variables β_i corresponding to the data samples x_i. The resulting hyper-sphere is characterized by the subset of data points with the corresponding β_i different from zero. These points are called supporting vectors. The remaining

ones are located inside the hyper-sphere. In order to minimize $T = R^2\|w\|^2$, we shall use the same procedure as Chapelle et al. which estimates the gradient of R^2 and $\|w\|^2$ w.r.t. the considered hyper-parameters [9].

7 Experimental results

7.1 Experiments with a synthetic benchmark

Let us consider synthetic data points representing a non-separable two-class problem of two attributes. The classes' distributions are bimodal and symmetric w.r.t. the origin. The optimal solution to this problem is a nonlinear frontier much closer to a hyperbole. The classes are of equal priors and their class conditional distributions are respectively $p(x|y = 1) = \frac{1}{2}\mathcal{N}(\mu_1, \Sigma) + \frac{1}{2}\mathcal{N}(-\mu_1, \Sigma)$, and $p(x|y = -1) = \frac{1}{2}\mathcal{N}(\mu_2, \Sigma) + \frac{1}{2}\mathcal{N}(-\mu_2, \Sigma)$, where $\Sigma = 0.9.\mathbf{I}, \mu_1 = (2, 2)^T, \mu_2 = (-2, 2)^T$ and $\mathcal{N}(\mu, \Sigma)$ is a Gaussian density function with its mean vector μ, Σ being its covariance matrix and $\mathbf{I}$ is the identity matrix. We have considered three distinct data sets : a training set, a validation set and a testing set. The size of each of them is 240. In what follows, we propose to assess the solutions produced by the previously studied criteria with different initial settings. As well, we will be interested in the compactness of the resulting models.

The quadratic optimization core for the SVM training is SVM^{light} chunking algorithm [38]. In order to invert the modified Hessian matrix H in equation 11, we use an LU decomposition. To prevent H from being badly conditioned, we add a small constant ϵ whose magnitude is 10^{-7} to the diagonal components of the matrix. A Cholesky decomposition might also be used instead.

We considered three initial configurations using an RBF kernel with three different values of the spread σ (Table A.1). In the first configuration, the RBF parameter is set to an intermediate value producing a classifier with low training errors and few support vectors ($\sigma_0 = \sqrt{2}$). The second value of the spread is chosen much smaller ($\sigma_0 = 0.1$), hence enforcing the SVM to overfit the training data and to produce a large number of support vectors. The third configuration attempts to reduce the initial complexity of the SVM by setting a large spread for the RBF kernel ($\sigma_0 = 45$). Note that $\sigma = \inf$ yields a quasi-linear decision surface. For each of these configurations, we use three values of C: 10^4, 10^2 and 1. During the model selection procedure, we record the magnitude of the objective function, the number of support vectors and the testing error.

Table A.2 compares the simulation results corresponding to the minimization of the empirical error, the VC dimension, $GACV_{L1}$ and $GACV_{L2}$ with three initial values of σ. The symbols Obi (Obf), SVi (SVf), Tsi (Tsf) refer respectively to the objective function, the number of support vectors and the testing error prior to (after) the optimization. Note that $GACV_{L1}$ and the empirical error criteria are used with the L1 SVM, whereas $GACV_{L2}$ and the VC dim are used with the L2 SVM.

For $\sigma_0 = \sqrt{2}$ and C=10000, the L1 and L2 models provide respectively a testing error of 9.58% and 10% and a support vectors number of 32 and 34. The VC dimension minimization reduces the test error from 10% to 6.25%, whereas $GACV_{L1}$ and $GACV_{L2}$ reduce the error to 9.17% and 8.33% respectively not without increasing dramatically the SV number to 124 and 132 respectively. Hopefully, the empirical error reduces the testing error to 4.58% with the lowest SV number. For $C = 100$ and $C = 1$, the empirical error

minimization produces the lowest error (3.75%) with equal complexities. For $GACV_{L1}$ and $GACV_{L2}$ the testing error remains slightly unchanged whereas the VC dimension deteriorates the error. For $\sigma_0 = 0.1$, we report model selection results when an initial over-fitting of the data is considered. Again, the empirical error criterion allows the most compact and the lowest testing error for C=1, 100 and 10000. The number of support vectors is reduced to 10% of its initial number for C=100. The final testing errors are equal to those found with $\sigma_0 = \sqrt{2}$. For C=1, $GACV_{L1}$ and $GACV_{L2}$ stop at the same initial configuration. The VC dimension reduction contrasts with the efficiency of the empirical error criterion. Indeed, on the initial configuration, the objective function is not convex and the performance of the classifier is altered for C=10^2 and C=10^4 whereas the reduction is insignificant for C=1. For $\sigma_0 = 45$, we show the results when an initial large spread is chosen for the RBF kernel. The VC dimension, even if reduced, increases the error dramatically whereas the error is highly reduced with $GACV_{L1}$ for C=10^2 and σ_0=45. Figures A.1-a and A.1-b show the corresponding decision frontiers before and after the optimization. For C=10^4, $GACV_{L2}$ gives 4.17% of error and 29 support vectors. The minimization of $GACV_{L1}$ however fails to decrease the error. For C=1, $GACV_{L1}$ and the empirical error criterion exit at the first iteration because the procedure grapples at a plateau. The colored zones in the table A.2 indicate the least error and SV number for each initial kernel configuration. Our criterion clearly much outperforms the other criteria.

For multi-class data, it is necessary to share the decision process among multiple classifiers and to combine their votes in order to predict the observation's class membership. Unlike monolithic classifiers such as MLP or RBF networks, optimizing the hyper-parameter values for a set of SVMs is a complex task that depends on the data, the considered kernel and the kind of dichotomy. Given a problem of M classes, in the one-against-all strategy, we need to consider nM variables, whereas in the one-against-one strategy there are $nM(M-1)/2$ variables to deal with, resulting in simpler decision frontiers with low SV number, thus reducing the size of the Hessian H in the equation 11. Indeed, the minimization of the empirical error may be used to optimize every classifier among the set of SVMs for the $M(M-1)/2$ data partitions. This strategy considers a couple of classes (i, j) independently of the remaining ones. As a result, each classifier will end up with its own kernel parameters which depend only on the distribution of the data classes i and j. Algorithm 2 gives the model selection procedure for multiple data classes with the one-against-one data partitioning.

The couple (i, j) refers to the classifier, the training and validation partitions and the logistic function (sigmoid) being considered. The algorithm given above has been experimented on images of handwritten digits from the US postal service database. It is a well known benchmark within the pattern recognition community. It contains 9298 images of handwritten digits from which 7291 images are used as the training set and 2007 images as the testing set. This data set has been collected from images of envelopes [39]. Each image consists of 16×16 pixels of grey level varying from 0 to 255. Due to the low

Algorithm 2 `Meta-algorithm`

Inputs : - Training data set

- Validation data set

- Initial values of hyper-parameters

Output : Final values of hyper-parameters.

Foreach i,j = 0:M-1 and i < j **Do**
  While not converging **Do**
    classifier(i,j) on **{training(i,j)}**,
    train **sigmoid(i,j)** on **{validation(i,j)}**,
    estimate **objective(i,j)** on **{validation(i,j)}**,
    update **hyper-parameter(i,j)**,
  End While
End For
Foreach i,j = 0:M-1 and i < j **Do**
  train **classifier(i,j)** on partition **{training(i,j)** $\cup$ **validation(i,j)}**
End For

resolution, it is a rather hard classification problem on which the human error is around 2.5 % [40]. As the original corpus does not contain any validation set, we split the training set into a training partition of 5291 samples and a validation partition of 2000 samples. Once the model selection process is terminated, the classifiers are trained with the optimized kernel parameters on the original training set.

We have implemented three variants of the minimization algorithm based on the gradient descent, the Quasi-Newton and the momentum. In Figure A.2, we show the results obtained with the momentum variant. The curves (a), (b), (c)

and (d) plot respectively the variations of the empirical error, the validation error rate, the number of support vectors and the testing error during the model selection process for the couple of classes (0,2),(0,3) and (0,4) with an RBF kernel ($\sigma_0 = \sqrt{2}$) and $C = 1000$. Figure A.2-a shows that the empirical error decreases smoothly while the corresponding objective functions being convex. Indeed, the criterion minimization goes along with a quick reduction of the validation error between the fifth and tenth iterations, which is lowered to 0% for the models (0,2) and (0,4), and 0.48% for the model (0,3) (Figure A.2-b). The decrease of the SV number in curve (c) is proportional to that of the empirical error magnitude which is reduced while the validation error stabilizes at zero. The second result is rather interesting and demonstrates that the reduction of the empirical error reduces the testing error and complexity of the classifier. If we examine the couple of classes (0,4), a manual model selection procedure would be content with the model produced at iteration five since its related validation error attains zero. But the number of support vectors at this iteration is $\simeq 600$, that is to say 4.5 times the final number of support vectors at the convergence. The plot of Figure A.2-d, shows that the testing error is consequently reduced to 1.44%, 1.90% and 0.36% for the classes (0,2), (0,3) and (0,4) respectively.

Table A.3 depicts the error rates and the corresponding SV number of the pairwise classifiers on the testing set before and after the minimization of the empirical error. The indexes i and j in the table refer to the pairwise classifier (i, j). While the errors are efficiently reduced, it can be noticed that the resulting performance depends on the difficulty of the considered classification problem. For example, because of their implicit similarity, the classes 5 and 8, or 2 and 3 are harder to resolve than 6 and 9. This fact is reflected by higher

complexities for the resulted classifiers.

In order to yield a class membership, we gather all the votes from the 45 resulting classifiers and apply a majority vote scheme. Given the vote magnitude f_{ij} of the classifier (i, j), the class membership is issued as $c = \arg\max_i \sum_{j=0, i \neq j}^{9} [\hat{f}_{ij}]_*$, where $\hat{f}_{ij}$ is equal to the raw SVM output f_{ij} normalized through the margin value. The heaviside operator $[\ ]_*$ acts as an activation function which may take smoother forms as noticed by Moreira et al. [41]. Table A.4 shows the resulted error rates for the combined classifiers with different kernels. The results are reported for three kinds of kernels namely KMOD [42], RBF and the polynomial kernel of degree two. As reported in Table A.4, the proposed methodology provides satisfactory performances. The lowest testing error is obtained for KMOD at 4.25% while RBF yields 4.43% of error, which is definitely the state of art performance for an RBF SVM.

8 Conclusion

We proposed a new model selection methodology based on the minimization of the probability of error estimated on a validation set. The method includes an automatic minimization procedure that efficiently reduces the empirical error. The work is articulated around two parts. In the first one, we compared the proposed methodology to the minimization of the radius-margin criterion and GACV. The former is an upper bound of the VC dim and the latter is an upper bound of the misclassification error. For GACV, we also presented a minimization algorithm for the L1 and L2 variants of SVM. The experimental part includes a benchmark comparison of each of the three model selection procedures on a synthetic two-class problem. The minimization of the empir-

ical error proves to be efficient and stable at whatever the initial setting of the kernel parameters. Also, the resulting solution is sparse with the lowest number of support vectors. On the other hand, the minimization of GACV has been shown to be inefficient when an initial over-fitting of the data holds. Also, the radius-margin minimization reduces the error rate in the only case where the initial parameter value σ_0 is close to the searched optimum and $C = 10^4$ (Table A.2). One major hypothesis for the radius-margin criterion to be valuable, is that the data must be separable. When the classes are not separable, its minimization causes $\|w\|$ to approach zero and puts a great portion of the data vectors into the margin, thus causing a catastrophic overfitting. The GACV criterion on the other hand, is a first-order approximation of the leave-one-out error which assumes that the set of support vectors does not change when an arbitrary data sample has been removed from the training partition. While it may be the case when the initial solution is sparse, it is no longer true when a great portion of the training examples are support vectors. When multiple data classes are available, we considered a data partitioning w.r.t. the one-against-one strategy. So, we optimized locally the whole set of pairwise classifiers by estimating and minimizing the probability of error on a validation set. On the USPS benchmark, the methodology proves to automatically yield a calibrated set of classifiers with state of the art performances.

ACKNOWLEDGMENTS: Thanks are due to the Natural Sciences and Engineering Research Council of Canada, and the NATEQ Program of the Ministry of Education of Quebec, for their support of this research project.

A Automatic minimization of GACV

In the following, we give a gradient based minimization algorithm for the GACV criterion.

Let $\nabla GACV(\underline{\theta}) = \frac{\partial}{\partial \underline{\theta}} GACV(\underline{\alpha}, \underline{\theta})$ be the vector of partial derivatives $\partial GACV(\underline{\alpha}, \underline{\theta})/\partial \underline{\theta_j}$, where θ_j is a kernel parameter for example. Again, its size corresponds to the number of considered hyper-parameters. We may write it as follows $\frac{\partial}{\partial \underline{\theta}} GACV(\underline{\alpha}, \underline{\theta}) =$

$$\frac{1}{l}\left(\sum_{i=1}^{l}\frac{\partial \xi_i(\theta)}{\partial \underline{\theta}} + 2\sum_{y_i f_i < -1}\left(K_{ii}\frac{\partial \alpha_i}{\partial \underline{\theta}} + \alpha_i \frac{\partial K_{ii}}{\partial \underline{\theta}}\right) + \sum_{y_i f_i \in [-1,1]}\left(K_{ii}\frac{\partial \alpha_i}{\partial \underline{\theta}} + \alpha_i \frac{\partial K_{ii}}{\partial \underline{\theta}}\right)\right).$$

For exponentiated radial kernels such as RBF and KMOD, $\frac{\partial K_{ii}}{\partial \underline{\theta}} = 0,\ \forall \underline{\theta} \in \mathrm{R}^n$, then,

$$\frac{\partial}{\partial \underline{\theta}} GACV(\underline{\alpha}, \underline{\theta}) = \frac{1}{l}\left(\sum_{i=1}^{l}\frac{\partial \xi_i(\theta)}{\partial \underline{\theta}} + 2\sum_{y_i f_i < -1}\frac{\partial \alpha_i}{\partial \underline{\theta}} + \sum_{y_i f_i \in [-1,1]}\frac{\partial \alpha_i}{\partial \underline{\theta}}\right).$$

The partial derivative of the ξ_i variable w.r.t. $\underline{\theta}$ is

$$\frac{\partial \xi_i(\underline{\theta})}{\partial \underline{\theta}} = \frac{\partial \xi_i(\theta)}{\partial f_i}\frac{\partial f_i}{\partial \underline{\alpha}}\frac{\partial \underline{\alpha}}{\partial \underline{\theta}} = \begin{cases} -y_i \frac{\partial f_i}{\partial \underline{\alpha}}\frac{\partial \underline{\alpha}}{\partial \underline{\theta}}\ if\ y_i f_i \leq 1 \\ \\ \underline{0}\ otherwise \end{cases}$$

In the above equation, $\underline{0}$ refers to a $1 \times n$ vector of zeros.

The vector of partial derivatives $\partial \alpha_i/\partial \underline{\theta}$ is the i^{th} row of the $k \times n$ matrix $\partial \underline{\alpha}/\partial \underline{\theta}$ whose computation is given in equation 11. The partial derivatives $\partial f_i/\partial \underline{\alpha}$ comprise a $1 \times k$ vector of values $\partial f_i/\partial \alpha_j$ given in equation 10. The estimation of $\partial K_{ii}/\partial \underline{\theta}$ for an arbitrarily shaped kernel depends on its current parameters and the magnitude of the data point x_i. For an L2 SVM, the computation of

the gradient of $GACV_{L2}$ is slightly similar except for the derivative of $\xi_i^2/2$ which is $\xi_i \frac{\partial \xi_i(\theta)}{\partial \underline{\theta}}$.

References

[1] V.N. Vapnik. *Estimation of Dependences Based on Empirical Data [in Russian]*. Nauka, Moscow, Russia, 1979.

[2] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[3] B. Schölkopf, C. Burges, and A. Smola. Introduction to support vector learning. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 1–15. MIT Press, 1999.

[4] C.J.C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector machines. In M.C. Mozer, M.I. Jordan, and T.Petsche, editors, *Advances in Neural Information Processing Systems 9: Proceedings of the 1996 Conference*, volume 9, pages 375–381. The MIT Press, 1997.

[5] V. Blanz, B. Schölkopf, H.H. Bülthoff, C. Burges, V. Vapnik, and T. Vetter. Comparison of view-based object recognition algorithms using realistic 3d models. In C. von der Malsburg, W. von Seelen, J.-C. Vorbrüggen, and B. Sendhoff, editors, *Artificial Neural Networks*, volume 1112 of *Lecture Notes in Computer Science*, pages 251–256. Springer Verlag, Bochum, Germany, 1996.

[6] M. Schmidt. Identifying speakers with support vector networks. In *Proceedings of the 28th Symposium on the Interface*, Sydney, Australia, 1996.

[7] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *CVPR '97: Proceedings of the 1997 Conference*

on Computer Vision and Pattern Recognition (CVPR '97), pages 130–136. IEEE Computer Society, 1997.

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

[9] O. Chapelle and V.N. Vapnik. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

[10] O. Chapelle and V.N. Vapnik. Model selection for support vector machines. In S. Solla, T. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12: Proceedings of the 1999 Conference*, volume 12, pages 230–236. MIT Press, 2000.

[11] G. Wahba, X. Lin, F. Gao, D. Xiang, R. Klein, and B. Klein. The bias-variance trade-off and the randomized gacv. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems 11: Proceedings of the 1998 Conference*, volume 11, pages 620–626. MIT Press, 1999.

[12] S.J. Hanson and D.J. Burr. Minkowski-r back-propagation: Learning in connectionist models with non-euclidian error signals. In D.Z. Anderson, editor, *Advances in Neural Information Processing Systems 0: Proceedings of the 1987 Conference*, volume 0, pages 348–357. American Institute of Physics, 1988.

[13] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical report, MIT, 1997.

[14] R. Courant and D. Hilbert. *Methods of Mathematical Physics*. Interscience, 1953.

[15] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

[16] B. Schölkopf, A.J. Smola, and K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[17] S. Mika, B. Schölkopf, A.J. Smola, K.R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de–noising in feature spaces. In M.S. Kearns, S.A. Solla, and D.A. Cohn, editors, *Advances in Neural Information Processing Systems 11: Proceedings of the 1998 Conference*, volume 11, pages 536–542. MIT Press, 1999.

[18] S.-W. Kim and B.J. Oommen. On utilizing search methods to select subspace dimensions for kernel-based nonlinear subspace classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):136–141, January 2005.

[19] S.-W. Kim and B.J. Oommen. On using prototype reduction schemes and classifier fusion strategies to optimize kernel-based nonlinear subspace methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):455–460, March 2005.

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

[21] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(26):314–319, 1985.

[22] G.E. Hinton. Learning translation invariant recognition in a massively parallel network. In J.W. de Bakker, A.J. Nijman, and P.C. Treleaven, editors, *Proceedings of PARLE Conference on Parallel Architectures and Languages Europe*, pages 1–13, Berlin, 1987. Springer Verlag.

[23] D. MacKay. *Bayesian Methods for Adaptive Models*. PhD thesis, 1992.

[24] T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19:201–209, 1975.

[25] A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-Posed Problems*. V.H. Winston and Sons, Washington, DC, 1977.

[26] D. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

[27] S. Sigurdsson, J. Larsen, and L.K. Hansen. On comparison of adaptive regularization methods. In B. Widrow, L. Guan, K. Paliwa, T. Adali, J. Larsen, E. Wilson, and S. Douglas, editors, *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, volume 11, pages 221–230, 2000.

[28] T. Joachims. *The maximum- margin approach to learning text classifiers: method, theory and algorithms.* PhD thesis, University of Dortmund, 2000.

[29] T.S. Jaakola and D. Haussler. Probabilitic kernel regression models. In D. Heckerman and J. Whittaker, editors, *Proceedings of the 1999 Conference on AI and Statistics*, San Mateo, CA, 1999. Morgan Kaufmann.

[30] M. Opper and O. Winther. Gaussian processes and svm: Mean field and leave-one-out. In A.J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 311–326. MIT press, 2000.

[31] A. Smola. *Learning with Kernels.* PhD thesis, GMD First, Berlin, Germany, 1998.

[32] R.C. Williamson, A.J. Smola, and B. Schölkopf. Entropy numbers, operators and support vector kernels. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 9, pages 127–144. MIT Press, 1999.

[33] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[34] J. Shawe-Taylor, C.K.I. Williams, N. Cristianini, and J. Kandola. On the eigenspectrum of the gram matrix and the generalisation error of kernel pca. *IEEE Transactions In Information Theory*, 2005 (to appear).

[35] N.E. Ayat, M. Cheriet, and C.Y. Suen. Optimization of the svm kernels using an empirical error minimization scheme. In S.W. Lee and A. Verri, editors, *Pattern Recognition with Support Vector Machines*, volume 2388 of *Lecture Notes in Computer Science*, pages 354–369. Berlin Heidelberg, July 2002.

[36] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3), October 1999.

[37] H. Lin, C. Lin, and R. C. Weng. A note on platt's probabilistic outputs for support vector machines. Technical report, National Taiwan University, Taipei 116, Taiwan, 2003.

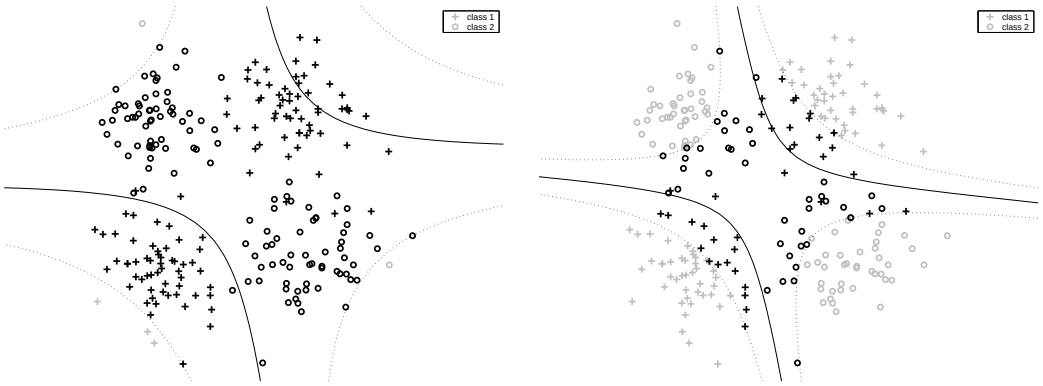
[38] T. Joachims. Making large-scale svm learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11. 1999.

[39] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Winter 1989.

[40] J. Bromley and E. Sackinger. Neural-network and k-nearest-neighbor classifiers. Technical Report 11359-910819-16TM, ATT, 1991.

[41] M. Moreira and E. Mayoraz. Improved pairwise coupling classification with correcting classifiers. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 160–171. Springer-Verlag, 1998.

[42] N.E. Ayat, M. Cheriet, and C.Y. Suen. Kmod-a two parameter svm kernel for pattern recognition. In *Proceedings of the IEEE International Conference on Pattern Recogntion*, volume 3, pages 30331–30334, Quebec, Canada, August 2002.



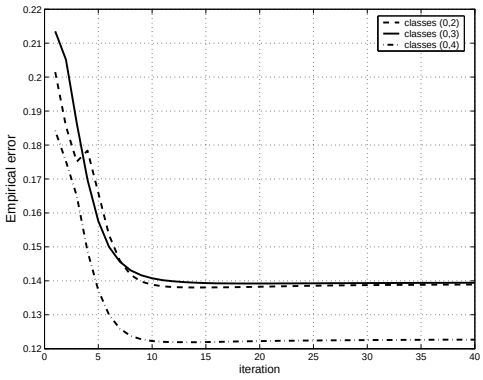
(a) before optimization (b) after optimization

Fig. A.1. Frontiers of the L1 SVM before and after GACV minimization for C=100 and σ_0=45 (Table A.2). Dark points represent support vectors.

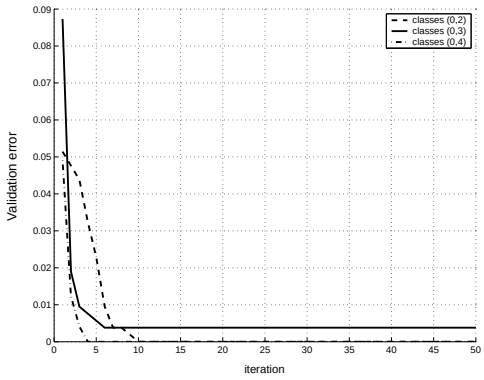
Table A.1

Classical common kernels (a in KMOD is a normalization constant [42]).

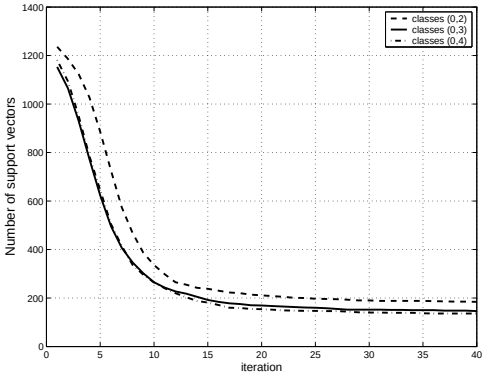
Kernel	**Formula**
Linear	$K(x,y) = x.y$
Polynomial	$K(x,y) = (ax.y + b)^d$
RBF	$K(x,y) = \exp(-\|x-y\|^2/\sigma^2)$
KMOD	$K(x,y) = a\left(\exp(\frac{\gamma^2}{\|x-y\|^2+\sigma^2}) - 1\right)$



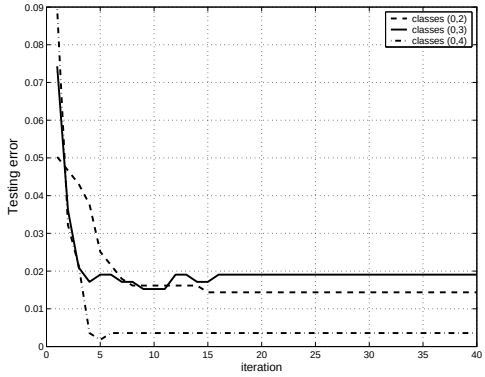
(a) Variation of the empirical error



(b) Variation of the validation error



(c) Variation of the SV number



(d) Variation of the testing error

Fig. A.2. Optimization on USPS for the models (0,2),(0,3) and (0,4) (RBF, C=1000).

Table A.2

Comparison of the criteria on the two-class problem for an RBF kernel with three initial values of σ_0). The grey boxes indicate the lowest error and SV number.

	criterion	**Emp. Err.**			**VC dim**			$GACV_{L1}$			$GACV_{L2}$		
	σ_0	$\sqrt{2}$	0.1	45	$\sqrt{2}$	0.1	45	$\sqrt{2}$	0.1	45	$\sqrt{2}$	0.1	45
$c = 10^4$	Obi	0.15	0.35		16000	109.37	5786	199.1	0.92	2301.7	143.2	0.92	1839.1
	SVi	32	238	N.D	34	238	91	32	238	57	34	238	91
	Tsi (%)	9.58	27.08		10.0	27.08	5.0	9.58	27.08	6.67	10.0	27.08	5.0
	Obf	0.10	0.12		96.0	114.24	102	1.7	0.92	1.0	2.8	0.92	689.0
	SVf	22	23	N.D	225	240	240	124	238	240	132	238	29
	Tsf (%)	4.58	6.25		6.25	39.58	52.08	9.17	27.08	50.0	8.33	27.08	4.17
$c = 10^2$	Obi	0.10	0.35	0.18	524.0	108.33	236.3	4.8	0.92	98.14	4.7	0.92	76.13
	SVi	40	238	235	48	238	240	40	238	235	48	238	240
	Tsi (%)	5.42	27.08	25.0	5.0	27.5	5.83	5.42	27.08	25.0	5.0	27.5	5.83
	Obf	0.09	0.08	0.08	110.0	112.83	12.6	2.4	0.92	31.32	4.5	0.92	75.03
	SVf	25	25	25	238	240	240	82	238	77	50	238	240
	Tsf (%)	3.75	3.75	3.75	34.6	38.75	50.83	5.38	27.08	7.5	5.42	27.5	5.42
$c = 1$	Obi	0.09	0.35		16.3	56.36	3.13	0.25	0.91		0.3	1.06	2.48
	SVi	61	238	N.D	138	240	240	61	238	N.D	138	240	240
	Tsi (%)	3.75	28.33		4.17	27.92	42.5	3.75	28.33		4.17	27.92	42.5
	Obf	0.08	0.08		13	56.22	2.55	0.25	0.91		0.3	1.06	2.48
	SVf	50	50	N.D	240	240	240	62	238	N.D	138	240	240
	Tsf (%)	3.75	3.75		48.8	27.92	43.33	3.75	28.33		4.17	27.92	42.5

Table A.3

Testing error (in percentage) and SV number on USPS before and after model selection (C=1000,rbf).

j	1	2	3	4	5	6	7	8	9
i	before optimization								
0	3.05/899	5.03/1236	7.43/1153	8.94/1181	24.08/1198	10.02/1284	6.92/1203	19.24/1215	7.65/1235
1	-	4.11/598	4.42/523	4.31/555	4.25/468	4.15/555	4.38/473	4.42/486	4.31/504
2	-	-	25.55/947	22.36/977	35.20/892	14.67/982	10.43/900	29.95/909	11.20/931
3	-	-	-	3.83/902	11.96/817	10.42/907	9.27/825	22.89/834	10.50/855
4	-	-	-	-	6.39/847	9.46/937	8.36/857	19.67/864	10.34/893
5	-	-	-	-	-	10.30/852	9.12/769	21.78/779	10.09/800
6	-	-	-	-	-	-	7.26/859	7.44/869	6.34/890
7	-	-	-	-	-	-	-	8.63/787	3.70/826
8	-	-	-	-	-	-	-	-	8.75/818
i	after optimization								
0	0.48/58	1.44/184	1.90/146	0.36/134	0.96/205	0.38/187	0.59/96	1.14/151	0.37/108
1	-	1.08/92	0.70/74	1.51/95	0.71/73	1.38/90	0.97/61	0.93/96	0.68/80
2	-	-	2.47/198	2.01/242	1.12/205	0.82/226	1.16/148	3.57/219	0.27/152
3	-	-	-	0.55/127	4.60/232	0.60/123	1.60/119	3.31/197	0.87/141
4	-	-	-	-	1.11/174	1.89/177	2.31/199	0.82/182	2.39/227
5	-	-	-	-	-	0.91/195	0.65/130	2.76/204	0.89/148
6	-	-	-	-	-	-	0.32/99	0.60/144	0.29/102
7	-	-	-	-	-	-	-	1.60/135	3.40/235
8	-	-	-	-	-	-	-	-	1.46/175

Table A.4

Resulted testing errors (in percentage) for different kernels.

kernel	KMOD	RBF	poly2
Testing error	4.25	4.43	5.37