Optimizing Resources in Model Selection for Support Vector Machines

Mathias M. Adankon, Mohamed Cheriet, Nedjem E. Ayat
Laboratory for Imagery, Vision, and Artificial Intelligence
École de Technologie Supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada, H3C 1K3
mathias@livia.etsmtl.ca, mohamed.cheriet@etsmtl.ca, nedjem@livia.etsmtl.ca

***Abstract*— Tuning SVM kernel parameters is a an important step for achieving a high-performing learning machine. The usual automatic methods used to tune these parameters require an inversion of the Gram-Schmidt matrix or a resolution of an extra quadratic programming problem. In the case of a large dataset these methods require the addition of huge amounts of memory and a long CPU time to the already significant ressources used in the SVM training. In this paper, we propose a fast method based on an approximation of the gradient of the empirical error along with incremental learning, which reduces the resources required both in terms of processing time and of storage space.**

I. Introduction

Support vector machines (SVM) are particular classifiers that are based on the margin-maximization principle. They perform structural risk minimization which was introduced to machine learning by Vapnik[16], and which have yielded excellent generalization performance [8].

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 product $\Phi(x).\Phi(y) = k(x, y)$, where $k()$ is called the kernel function [14], [3].

The SVM architecture comprises various types of parameters, including the support vector coefficients and the kernel parameters. In particular, the choice of the kernel parameters can significantly affect the SVM performance. As an illustration, Figure 1 shows the variation of the error rate on a validation set versus the variance of the Gaussian kernel. The task is to classify the digits 1 and 8 taken from the MNIST data benchmark. Clearly, the best performance is realized from an optimum choice of the kernel parameters.

Several methods have been proposed for choosing the best value for the kernel parameters. Among them, we may cite:
• Generalized Approximate Cross-Validation (GACV) proposed in [19], [20], where the authors suggest the minimization of an upper bound of the expected classification error;

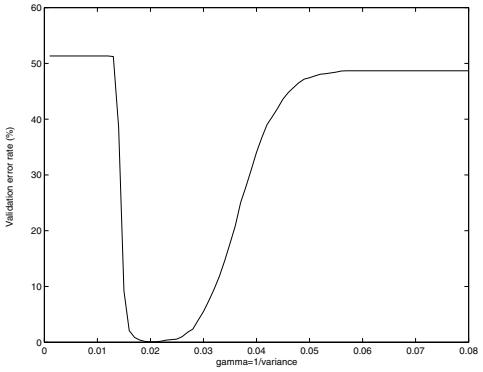

Fig. 1. Validation error rate for different values of the variance of the RBF kernel for binary problem

• Optimization methods based on a radius-margin bound and span bound developped in [6] and [7] where an analytical proxy of the VC dimension [17] is applyed for the minimization of the generalization error;
• Technique using an empirical error as proposed in [1] and [2], where an empirical estimate of the generalization error is minimized through a provided validation set.

All these methods are quite costly in computing time requiring among other things to invert the Gram-Schmidt matrix of support vectors during gradient computation or to resolve an additional QP problem. When a large dataset is used, the cost of this operation can be quite significant compared to the actual training of the SVM. In this paper, we propose an improved method of model selection based on the empirical error by using two techniques:
(i) approximation of the gradient of error
(ii) incremental learning strategy

This paper is structured as follows. In section 2, we give a review of the optimization of SVM kernel parameters using an empirical error. In section 3, we develop an approximation of the gradient. In section 4, we describe the technique of incremental learning for SVMs. In section 5, we present the experimental results confirming our algorithm. In the last section, we conclude the paper.

II. Model selection using Empirical Error criterion

In this section, we describe the optimization of the SVM kernel parameters using the empirical error as developed in [1], [2].

We first consider a binary classification problem. Let us have a dataset $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. In the feature space, the optimal separating hyperplane for the Support Vector Machine is defined by :

$$f(x_i) = \sum_{j=1}^{NVS} \alpha_j y_j k(x_j, x_i) + b \tag{1}$$

where α_j and b are found after resolving the quadratic optimization problem maximizing the margin; $j = 1, \ldots, NVS$ are the Support Vector indices corresponding to non-zero α_j and $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is the kernel function [14].

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

$$E_i = |t_i - \hat{p}_i| \tag{2}$$

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

The estimated posterior probability is determinated by :

$$\hat{p}_i = \frac{1}{1 + exp(A.f_i + B)} \tag{3}$$

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 [12]. In this paper, we use its improved algorithm developped by Lin [10].

The use of the model developed by Platt to estimate the probability makes it possible to quantify the distance from one observation to the hyperplane determined by the SVM using a continuous and derivable function. Indeed, the estimate of probability makes it possible to calibrate the distance $f(x_i)$ between 0 and 1 with the following properties:
- the observations of the positive class which are well classified and located apart from the margin have probabilities considered very close to 1 ;
- the observations of the negative class which are well classified and located apart from the margin have probabilities considered very close to 0;
- and the observations located in the margin have probabilities considered proportional to $f(x_i)$.

Thus with the empirical error criterion, only the misclassified observations and those located in the margin determined by the SVM are very important, since the other observations give almost null errors. Consequently, the minimization of the empirical error involves the reduction of the support vectors (observations being in the margin). In other words, the minimization of the empirical error makes it possible to select hyper-parameters defining a margin containing fewer observations. We then construct a machine with fewer support vectors, which reduces the complexity of the classifier. The results of the tests reported in [1] confirm this property of the SVM constructed using the empirical error.

In fact, we have :

$$|t_i - \hat{p}_i| = \begin{cases} \hat{p}_i & \text{if } y_i = -1 \\ 1 - \hat{p}_i & \text{if } y_i = 1 \end{cases}$$

Then :
$E_i \rightarrow 0$ when $\hat{p}_i \rightarrow 0$ for $y_i = -1$ and $\hat{p}_i \rightarrow 1$ for $y_i = 1$
Consequently :
$E_i \rightarrow 0$ if $f(x_i) < -1$ for $y_i = -1$ and $f(x_i) > 1$ for $y_i = 1$ (see Figure 2)

We notice that the minimization of the empirical error forces the maximum of the observations to be classified apart from the margin. This criterion is thus useful for regularizing the maximization of the margin for Support Vector Machines.

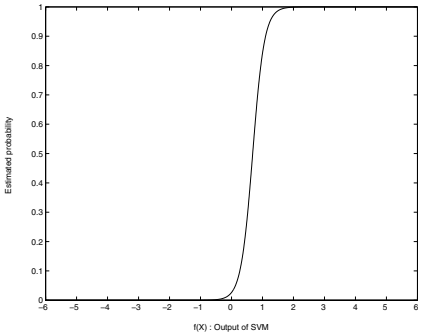

Fig. 2. Variation of estimated probability w.r.t. the output of the SVM

We assume that the kernel function depends on one or several parameters, encoded within the vector $\theta = (\theta_1, \ldots, \theta_n)$. The optimization of these parameters is performed by a gradient descent minimization algorithm[4] where the objective function is $E = \sum E_i$ (see Figure 3).

III. Gradient approximation of the empirical error

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 :

$$\frac{\partial E}{\partial \theta} = \frac{\partial}{\partial \theta}\left(\frac{1}{N}\sum_{i=1}^{N} E_i\right) = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial E_i}{\partial \theta} \tag{4}$$

with

$$\frac{\partial E_i}{\partial \theta} = \frac{\partial E_i}{\partial f_i}.\frac{\partial f_i}{\partial \theta}$$

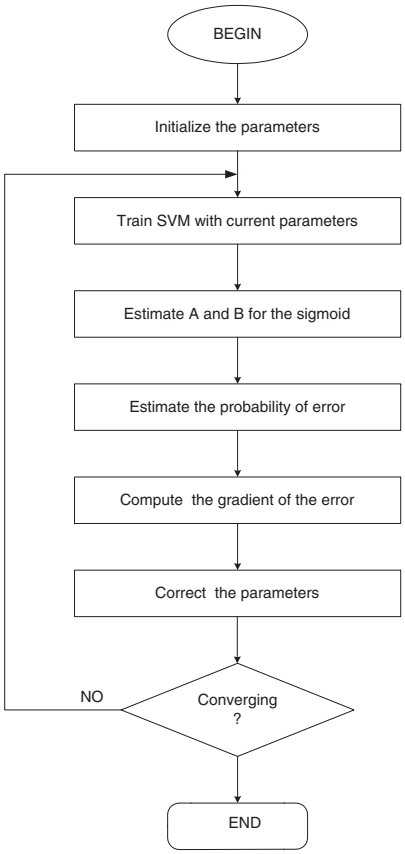

Fig. 3. Model selection using the empirical error

* Computation of $\frac{\partial E_i}{\partial f_i}$

$$\frac{\partial E_i}{\partial f_i} = \frac{\partial E_i}{\partial \hat{p}_i} \cdot \frac{\partial \hat{p}_i}{\partial f_i}$$

where

$$\frac{\partial E_i}{\partial \hat{p}_i} = \frac{\partial |t_i - \hat{p}_i|}{\partial \hat{p}_i} = \begin{cases} -1 & \text{if } t_i = 1 \\ +1 & \text{if } t_i = 0 \end{cases}$$

and

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

Then $\frac{\partial E_i}{\partial f_i}$ is equal to:

$$\frac{\partial E_i}{\partial f_i} = Ay_i\hat{p}_i(1 - \hat{p}_i) \quad (5)$$

* Computation of $\frac{\partial f_i}{\partial \theta}$

$$\frac{\partial f_i}{\partial \theta} = \frac{\partial}{\partial \theta}\left(\sum_{j=1}^{NVS} \alpha_j y_j k(x_j, x_i) + b \right)$$

$$\frac{\partial f_i}{\partial \theta} = \sum_{j=1}^{NVS} y_j \frac{\partial}{\partial \theta}\left[\alpha_j k(x_j, x_i) \right] + \frac{\partial b}{\partial \theta}$$

$$\frac{\partial f_i}{\partial \theta} = \sum_{j=1}^{NVS} y_j \left[\frac{\partial k(x_j, x_i)}{\partial \theta}\alpha_j + \frac{\partial \alpha_j}{\partial \theta}k(x_j, x_i) \right] + \frac{\partial b}{\partial \theta} \quad (6)$$

This derivative is composed of two parts. We may include the bias b into the parameter vector α as $(\alpha_1, \ldots, \alpha_{NVS}, b)$. Then, we use the following approximation proposed by Chapelle et al.[6].

$$\frac{\partial \alpha}{\partial \theta} = -H^{-1}\frac{\partial H}{\partial \theta}\alpha^T \quad (7)$$

where

$$H = \begin{pmatrix} K^Y & Y \\ Y^T & 0 \end{pmatrix} \quad (8)$$

In equation (8), K^Y represents the Hessian matrix of the SVM objective function. H is called the modified Gram-Schmidt matrix and its size is $(NVS + 1) \times (NVS + 1)$. Its components K^Y_{ij} are equal to $y_i y_j k(x_i, x_j)$ and Y is a vector of size $NVS \times 1$ containing support vector labels y_i. During the experiments, we note that the derivate $\frac{\partial \alpha_j}{\partial \theta}k(x_j, x_i)$ is negligible in comparison to $\frac{\partial k(x_j, x_i)}{\partial \theta}\alpha_j$. Then we approximate the equation (6) by:

$$\frac{\partial f_i}{\partial \theta} = \sum_{j=1}^{NVS} y_j \alpha_j \frac{\partial k(x_j, x_i)}{\partial \theta} \quad (9)$$

With this approximation, we do not need to invert the matrix H that has minimal time complexity of order $O((NVS+1)^2)$. Therefore the CPU time and the size of the memory required by the gradient descent algorithm will be lower than with traditional approaches.

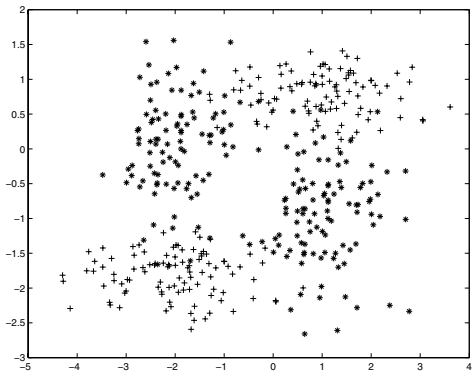

Fig. 4. Data of XOR problem

In order to assess our methodology we tested it on numerous synthetic and real problems. In Figures 5 and 6, we plotted the empirical error versus the number of iterations and the validation error versus the number of iterations during the gradient descent algorithm for the XOR problem with overlap (see Figure 4). We first note that the minimization of the empirical error involved the minimization of the error rate when we used the complete gradient or the approximation of gradient. We also noticed that the curves are the same in each case. Hence, we can efficiently use equation (9) when we compute the gradient of empirical error without inverting the modified Gram-Schmidt matrix.

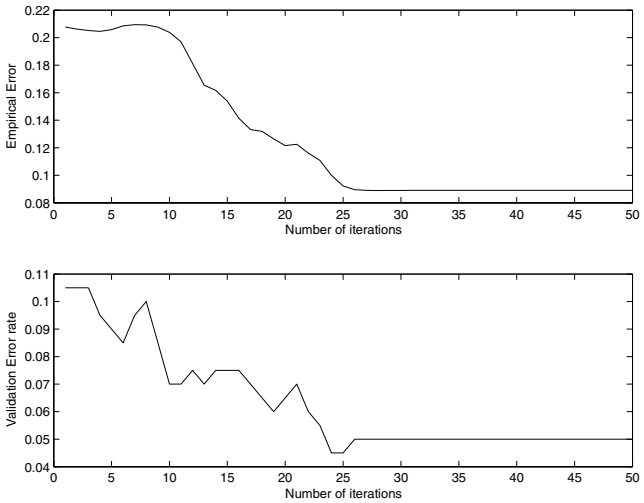

Fig. 5. Empirical error and Validation error rate versus the number of iterations during the optimization process with the complete gradient.

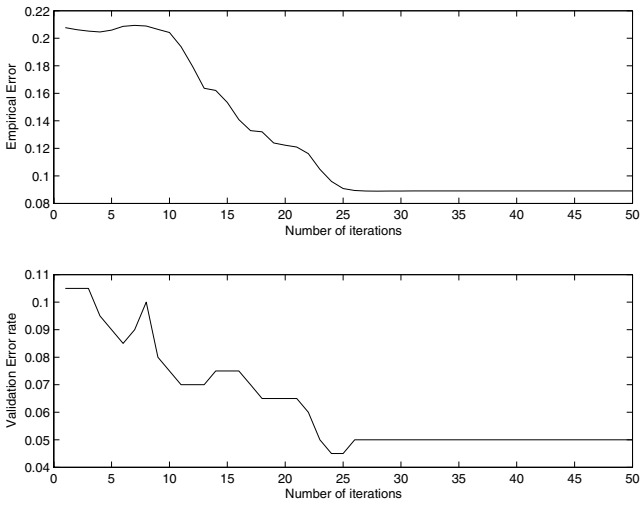

Fig. 6. Empirical error and Validation error rate versus the number of iterations during the optimization process with the approximate gradient.

IV. Optimisation of kernel parameters with Incremental learning

Support Vector Machines, unlike the other classifiers, offer a good power of generalization when the size of the training set is small. We exploit this property of SVMs to develop our method which consists of beginning the optimization process with a subset S of the training set that we called the working set, and adding ΔS, a part of the remaining samples, at each step.

The idea of incremental learning for SVMs was introduced in [15], where the training preserves only the support vectors at each incremental step. In this work, however, we preserve the samples that are in the margin or close to it because during the process, the samples that are outside the margin can become support vectors when the kernel parameters are updated in the next steps. Then, in our case, we remove from the working set S only the samples that are far from the temporary margin and add ΔS whose size is chosen dynamically, with respect to the behavior of the gradient norm. The idea behind this is to minimize the size of the working set when the current value of the kernel parameters is far from the optimal value. That is, if the gradient norm is large, the size of ΔS added to S is small because the large gradient norm means the current value of the parameters is not close to the optimal value (see Figure 7).

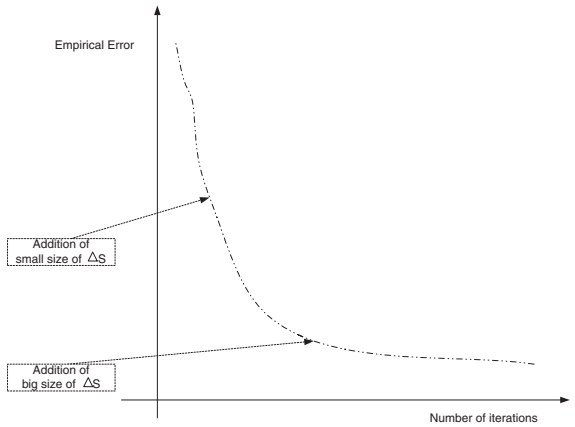

Fig. 7. Behavior of ΔS size w.r.t. gradient norm value.

At each incremental step, we update the working set S and the kernel parameters, after we retrain the SVM and select the optimal parameters using the minimization of empirical error. All in all, this combinaison makes it possible to optimize both the parameters α_i of the SVM and the parameters of the kernel and to drastically save on computing time. Figure 8 presents the details of the incremental learning algorithm.

1. *Initialize the kernel parameters*
2. *Initialize a working set S*
3. *Repeat until convergence*
  3.1 *Train the SVM with the set S*
  3.2 *Estimate the parameters A and B of the sigmoid*
  3.3 *Compute the gradient of error*
  3.4 *Update the kernel parameters*
  3.5 *Subtract from S the samples not close to the margin*
  3.6 *Add to S a part ΔS of remaining samples*

Fig. 8. Algorithm of Optimization of kernel parameters with incremental learning

TABLE I

TEST ERROR FOUND BY DIFFERENT ALGORITHMS FOR SELECTING THE SVM PARAMETERS. THE FIRST COLUMN REPORTS THE RESULTS FROM [13] AND THE TWO FOLLOWING COLUMNS ARE FROM [6].

	Cross-validation[13]	Radius margin bound[6]	Span bound[6]	Our approach
Breast cancer	26.04±4.7	26.84±4.71	25.59±4.18	25.48±4.38
Diabetes	23.53±1.73	23.25±1.70	23.19±1.67	23.41±1.68
Heart	15.95±3.26	15.92±3.18	16.13±3.11	15.96±3.13
Thyroid	4.80±2.19	4.62±2.03	4.56±1.97	4.70±2.07
Titanic	22.42±1.02	22.88±1.23	22.5±0.88	22.90±1.16

V. EXPERIMENTS

We used two types of benchmark databases, the first being the UCI benchmark tested by Chapelle et al. [6] and by Rätsch et al.[13] and the second being the MNIST database [9].

A. UCI Benchmark

We used 5 datasets from the UCI benchmark repository: breast cancer, diabetes, heart, thyroid and titanic. Each dataset is composed of 100 different training and test sets describing the binary classification problem.

We followed the same experimental setup as in [13] and [6]. On each of the first five training and test sets, the kernel parameters are optimized using our algorithm. Finally, the model parameters are computed as the median of the five estimations. Like in [13] and [6], RBF kernels are employed.

The results obtained are shown in Table 1, where we report the results obtained with the different model selection techniques. Our results are similar to those obtained by cross-validation[13] or Chapelle's methods[6]. However, the gain in complexity with our algorithm is significant, because, on one hand we do not invert the matrix H size $(NVS+1)\times(NVS+1)$ and on the other hand, we use incremental learning strategy.

B. MNIST database

In this section, we tested our algorithms on an isolated handwritten recognition problem, using the MNIST database. The MNIST (Modified NIST) database [9] is a subset extracted from the NIST database. The digits have been size-normalized and centered in a fixed-size image. The learning dataset contains 60,000 samples (50,000 for training and 10,000 for validation) and the testing dataset consists of 10,000 other samples.

For the test vote, we use pairwise coupling. We then train 45 classifiers, and each SVM is optimized by using the empirical error criterion. We use the RBF kernel with parameter C=100 and the size of set S is initialized at 2500.

For the test phase, we applied different types of couplings after mapping the 45 SVM outputs into probabilities. The probability that a given observation belongs to class $\omega_i (i = 1, ..., 10)$ is:

$$p_i = \frac{1}{45}\sum_{i\neq j}\sigma(p_{ij}) \tag{10}$$

$$\text{where} \qquad p_{ij} = P(x \in \omega_i / x \in \omega_i \cup \omega_j)$$

and σ is a coupling function. For a complete description see [11] where the following functions are reported.

PWC1 $\quad \sigma(x) = \begin{cases} 1 & \text{if } x > 0.5 \\ 0 & \text{otherwise} \end{cases}$

PWC2 $\quad \sigma(x) = x$

PWC3 $\quad \sigma(x) = \frac{1}{1+e^{-12(x-0.5)}}$

PWC4 $\quad \sigma(x) = \begin{cases} 1 & \text{if } x > 0.5 \\ x & \text{otherwise} \end{cases}$

PWC5 $\quad \sigma(x) = \begin{cases} x & \text{if } x > 0.5 \\ 0 & \text{otherwise} \end{cases}$

TABLE II

RESULTS OBTAINED WITH MNIST USING SEVERAL TYPES OF COUPLINGS

Couplaing model	Error test(%)
PWC1	1.6
PWC2	1.5
PWC3	1.7
PWC4	1.6
PWC5	1.5

We tested three algorithms for optimizing the kernel parameters : the initial algorithm reported in [1], the modified algorithm using the approximation of gradient and the algorithm of optimization based on incremental learning with approximation of gradient. The results are identical for the three cases and are shown in Table II.

The three algorithms yield the same results, but there is a difference with respect to the CPU time, which becomes very significant when the size of the training set is large. Figure 9

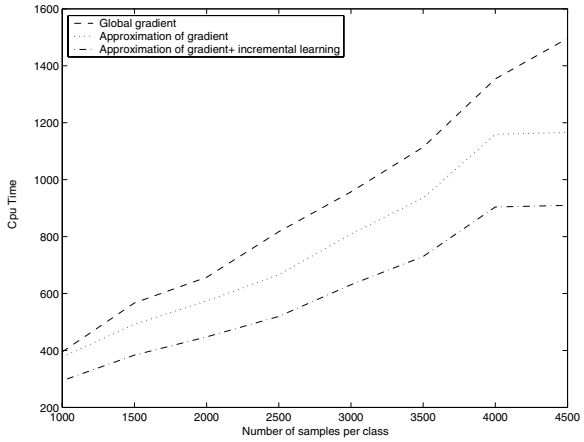

Fig. 9. Variation of CPU time depend on the size of training set with the different algorithms

shows the relationship between the size of the training set and the CPU time during the optimization process with the three algorithms. We notice that the CPU time is reduced from 5% to 25% when we use the approximation of gradient and with the incremental learning there is a 20 to 40% improvement in the CPU time gain.

VI. Conclusions

In this paper, we have described a model selection for SVMs using the empirical error criterion improved by incremental learning and the approximation of the gradient. Then the resources required during the optimization process, i.e. the CPU time and memory size, are drastically reduced. We have tested our method on many benchmarks which have produced promising results confirming our approach. We are undertaking actually a theoretical justification for the approximated gradient strategy.

Acknowledgments

The authors would like to thank Dr. Alain Biem from the IBM at Watson Research Center, N.Y., for his thoughtful suggestions that helped to improve this manuscript. Our thanks are also due to ACDI-PCBF for supporting Mr. Adankon's fellowship.

References

[1] N. E. Ayat, M. Cheriet and C. Y. Suen , "Empirical error based optimization of SVM kernels: application to digit image recognition", Proceedings of the International Workshop on Handwriting Recognition, pp. 292-297 Niagara, Canada, 2002.

[2] N. E. Ayat, "Sélection automatique de modèle des machines à vecteurs de support : Application à la reconnaissance d'images de chiffres manuscrits", École de Technologie Supérieure, Montréal, Canada, 2003.

[3] G. Baudat and F. Anouar, "Kernel-based methods and function approximation", Proceedings of the IJCNN, pp. 1244-1249, Washington, 2001.

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

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

[6] O. Chapelle , V. Vapnik , O. Bousquet and S. Mukherjee, "Choosing multiple parameters for support vector machines", Machine Learning, 2001.

[7] K.-M. Chung, W.-C. Kao, C.-L. Sun, L.-L. Wang, and C.-J. Lin. Radius Margin Bounds for Support Vector Machines with the RBF Kernel. Neural Computation, 15(2003), 2643-2681.

[8] C. Cortes and V.N. Vapnik, "Support-Vector Networks", Machine Learning, vol. 20, no. 3, pp. 273-297, 1995 .

[9] Y. LeCun, L. Bottom, Y. Bengio, P. Haffner, "Gradient-Based Learning Applied to Document Recognition", *Proceedings of the the IEEE,* vol. 86, no. 11, pp. 2278-2324, Nov 1998.

[10] H.-T. Lin, C.-J. Lin, and Ruby. C. Weng. "A Note on Platt's Probabilistic Outputs for Support Vector Machines", Technical Report, May 2003.

[11] M. Moreira and E. Mayoraz, "Improved Pairwise Coupling Classification With Correcting Classifiers", Proceedings of the 10th European Conference on Machine Learning, pp. 160-171, 1998.

[12] J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods", in the book of A.J. Smola, P. Bartlett, B. Schoelkopf and D. Schuurmans, pp. 61-74, 2000.

[13] G. Ratsch ,T. Onoda and K.-R. Muller, "Soft Margins for AdaBoost", Machine Learning,vol. 42,pp. 287-320, 2001.

[14] B. Scholkopf , C. J. C. Burges and A. J. Smola, "Advances in Kernel Methods: Support Vector Learning ", The MIT Press, Cambridge, Massachusetts, 1999.

[15] N.A. Syed and H. Liu and K.K. Sung, "Incremental Learning with Support Vector Machines", Proceedings of the International Joint Conference on Artificial Intelligence, 1999.

[16] V.N. Vapnik, "Statistical learning theory", John Wiley and Sons, New York, 1998.

[17] V.N. Vapnik, "Estimation of Dependences Based on Empirical Data", Springer verlag, Berlin, 1982.

[18] V. N. Vapnik, "Principles of Risk Minimization for Learning Theory", pp. 831838 in John E. Moody et al. (Eds.) Advances in Neural Information Processing Systems 4, Morgan Kaufman Publishers, San Mateo, CA, 1992.

[19] G. Wahba, Y. Lin and H. Zhang, "Margin-like quantities and generalized approximate cross validation for support vector machines", Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop, pp. 12-20, 23-25 Aug. 1999

[20] G. Wahba, X. Lin, F. Gao, D. Xiang, R. Klein and B. Klein, "The bias-variance tradeoff and the randomized GACV", Advances in Information Processing Systems 11, pp. 620-626. MIT Press,1999.