Tabu search model selection for SVM

Gilles Lebrun, Christophe Charrier, Olivier Lezoray, Hubert Cardot

International Journal of Neural Systems
Special Issue on Issue's Topic

Tabu Search Model Selection for SVM

GILLES LEBRUN, CHRISTOPHE CHARRIER, OLIVIER LEZORAY
Université de Caen Basse-Normandie, GREYC UMR CNRS 6072, Image Team,
6 bd. Marechal Juin, Caen, F-14050, France
{*gilles.lebrun, christophe.charrier, olivier.lezoray*}*@unicaen.fr*

HUBERT CARDOT
Université François Rabelais de Tours, Laboratoire Informatique EA 2101,
64 Avenue Jean Portalis, Tours, F-37200, France
hubert.cardot@univ-tours.fr

Received (to be inserted
Revised by Publisher)

A model selection method based on tabu search is proposed to build support vector machines (binary decision functions) of reduced complexity and efficient generalization. The aim is to build a fast and efficient support vector machines classifier. A criterion is defined to evaluate the decision function quality which blends recognition rate and the complexity of a binary decision functions together. The selection of the simplification level by vector quantization, of a feature subset and of support vector machines hyperparameters are performed by tabu search method to optimize the defined decision function quality criterion in order to find a good sub-optimal model on tractable times.

1. Introduction

Data mining is considered as one of the challenging research fields of the 21^{th} century. Extracting knowledge from raw data is a difficult problem which covers several disciplines: Artificial Intelligence, Machine Learning, Statistics, Data Bases. Machine learning methods aim at providing classification methods which induce efficient decision functions. Among all possible inducers, Support Vector Machine (SVM) are founded on strong statistical learning theory[1] and have become very popular in the past few years because they delivered state-of-the-art performance in many real world pattern recognition and data mining applications[2]: text categorization, hand-written character recognition, image classification, bioinformatics, etc. However, decision functions provided by SVM have a complexity which increases with the training set size[3,4,5]. A recent theoretical result by Steinwart[6] shows that the number of examples used by a given SVM decision function grows as a linear function of the number of examples in the SVM training set. Therefore, time decision with SVM inducer to classify huge datasets is not directly tractable; especially for application requiring high classification speed[7]. For instance, Martin et al[8] conclude that SVM are not well suited to edge detection. Indeed, the training time and complexity are too high to be interesting as regards the huge dataset they used. However, they use SVM as brute-force algorithm whitout taking care of model selection both training time and complexity. Our aim, in this paper, is to tackle that problem.

In recent years, there has been a lot of interest to improve learning methods using SVM. One way is to optimize the SVM algorithm to solve the associated quadratic problem more efficiently[3,9,10] (less time and less memory used). Other approaches fo-

cus on simplifying the SVM decision function. Many solutions were proposed to reduce the number of examples in SVM decision functions: (1) A preprocessing simplification step to reduce the size of SVM training set[4,11,12], (2) An incremental SVM learning method with stopping criteria[13,14,15], (3) A bound on the number of support vectors of the decision function[16,17] or (4) A post-processing to reduce the number of support vectors of the decision function[18,19]. Although SVM are less sensitive to the curse of dimensionality[20], dimensionality reduction techniques can improve their efficiencies[21,22]. Moreover, this is another way to reduce complexity of SVM decision functions[23,22]. For learning methods using SVM, model selection is critical. Indeed studies have shown that SVM generalization efficiency depends on the choices of SVM parameters[24,25]. Other studies[26,27] have shown that multiclass SVM are efficient if an efficient model selection is performed for each involved binary SVM. Therefore, as regards these considerations, new approaches aim at incorporating a simplification step into the model selection[11,5]. Previous cited works highlight that complete SVM model selection (reduction of both the number of examples and features, and selection of hyperparameters) is a hard and still unresolved problem. Solution proposed in most of these works focus only on a sub-problem of the complete model selection problem. At our knowledge, there actually exist no way in literature to achieve such difficult SVM complete model selection: simultaneously tuning the hyper parameters, selecting accurate features and relevant examples (this is usually performed in a sequential manner).

Our approach aims at unifying feature selection, simplification of training set and hyperparameters tuning as a global optimization problem is order to produce efficient and low complexities Binary Decision Functions (BDF) with SVM[28,29,30]. For this, a criterion named Decision Function Quality (DFQ) is defined which takes into account the recognition rate, the number of support vectors and the number of selected features of BDF. The objectives, for an efficient BDF, are to have a high recognition rate, few support vectors and few features used. The proposed DFQ criterion is based on regularization theory[31,32]. The fitting term is expressed in terms of recognition rate. The smoothness term is expressed in terms of model complexity (number of support vectors and features used by a BDF). With this expression of DFQ, one wants to find solutions that are simultaneously smooth (few support vectors and features) and close to the initial data (high recognition rate) in terms of a compromise. In the framework of Multi-Objective Optimization (MOO)[33,34], weighted sum of objectives in one possibility to transform MOO problem into a global single objective problem[23]. Then, regularization theory can be regarded as one way to transform the general problem of finding efficient solutions in the paradigm of multi-objective optimisation into the paradigm of single-objective optimisation. Indeed, one can produce fast and efficient decision SVM processes with a single-objective optimisation problem when DFQ criterion based on regularization theory is used to evaluate the produced solution. With the proposed approach, the selection of efficient values for all free parameters (feature selection or rejection, level of simplification and SVM hyperparameters) in order to optimize DFQ criterion is designed by the generic term of model selection.

Training set simplification is produced by the LBG (Linde, Buzo, Gray) algorithm used in vector quantization research field[35]. It has been retained because it can produce good prototypes of the initial dataset. Moreover, the simplification level is controlled by a single integer parameter the values of which are few and can range from extreme simplification with only one prototype by class to no simplification at all (*i.e.* selection of all initial examples in training set). However, the proposed learning method is enough general to be extended to other simplification methods[36,37,38]. To have a relevant tuning of SVM hyperparameters and an accurate selection of relevant features, an adapted tabu search (TS) method is proposed which includes specific intensification and diversification strategies. TS is relevant for SVM model selection since usual SVM model selection have local minima[39,19]. Moreover, TS has proved its suitability for other model selection problems[40], in particular with SVM learning problems[41,42].

Our approach is tested on several benchmark datasets and on an image segmentation problem. For the latter, the development of a microscopic cellular image segmentation application was performed[30]. This kind of application must be efficient for reliable analysis and fast to process huge quantity of images. In contrast, recent studies have focused on

improving segmentation quality[43,44,45]. Many segmentation schemes can have good qualities but their processing time is too expensive to deal with a great number of images per day[28,30]. The main reason is that pixel classification in a segmentation scheme requires most of the processing time. Therefore, the classifier design is crucial to produce fast and efficient pixel classification. Presented results show that our method satisfies those objectives for this cell segmentation application. Moreover, experimental results on benchmark datasets show that an efficient compromise between high generalization ability and low complexity can be found with our TS model selection.

Section 2 gives overviews of methods used by our TS model selection method. Section 3 describes the proposed new method and Section 4 gives experimental results. Last Sections conclude and propose future works.

2. Overviews of the underlying methods

2.1. *Support Vector Machines (SVM)*

The SVM was developed by Vapnik et al[1]. They are based on the structural risk minimization principle from statistical learning theory[1].
Let's consider a binary classification problem with training data $\{(x_i, y_i)\}_{i\in\{1,\dots,m\}}$ ($x_i \in \mathbb{R}^n$ and $y_i \in \{-1, +1\}$). A soft margin SVM[1,46] classifies an example x according to the sign of BDF h :

$$h(x) = \text{sign}(f(x)) \tag{1}$$

with

$$f(x) = \sum_{i\in\{1,\dots,m\}} \alpha_i y_i K(x_i, x) + b \tag{2}$$

and $K(\cdot,\cdot) \equiv \langle \phi(.), \phi(.)\rangle_{\mathbf{H}}$ the kernel function which defines an inner product in $\mathbf{H}$ when mapping $\phi : \mathbb{R}^n \rightarrow \mathbf{H}$ is performed. The coefficients α_i in (2) are obtained by solving a quadratic optimization problem[1] (threshold b depends of α_i values[1]). All examples x_i for which associated α_i is not equal to zero is called support vector. The set of all support vectors is noted SV (*i.e.* $SV = \{i|\alpha_i > 0\}$) and corresponds to training set examples used by BDF (1) to determine the class of each new example x. In soft margin formulation all the coefficients are bounded by a constant C (*i.e.* $\alpha_i < C$). C is the parameter that determines the trade-off between training errors and generalization capacities. An efficient algorithm SMO[3] and many refinements[9,47] were proposed to solve SVM quadratic optimization problem.

2.2. *SVM probabilities estimation*

The output (2) of an SVM is not a probabilistic value, but an un-calibrated distance measurement of an example x to the separating hyper-plane. Platt proposed a method[48] to map the SVM output into a positive class posterior probability by applying a sigmoid function to the SVM output:

$$p(y = +1|x) = \frac{1}{1 + e^{a_1 \cdot f(x) + a_2}} \tag{3}$$

This method is used in SVM combination schemes based on probabilistic estimation.

2.3. *SVM combination schemes*

SVM are specifically designed for binary problems. Several combination schemes have been developed to take into account that specificity and deal with multiclass problems[49,50,51,52]. Within all combination schemes, the *one-versus-all* scheme based on a winner-takes-all strategy and the *one-versus-one* (or pairwise) method based on a max-wins voting strategy are generally used[19,26]. When class probabilities on each binary problem are estimated (*c.f.* Section 2.2), the two above schemes have adapted decoding strategies to estimate class probabilities for the multiclass problem[53].

2.4. *Vector Quantization (VQ)*

VQ is a classification technique used in the compression field[35]. VQ maps a vector x to another vector x' that belongs to m' *prototypes* vectors which is called *codebook*. The *codebook* S' is built from a training set S_t of size m ($m >> m'$). The algorithm must produce a set S' of prototypes which minimizes the distorsion

$$d' = \frac{1}{m} \sum_{i=1}^{m} \min_{1\leq j\leq m'} d(x_i, x_j) \tag{4}$$

where $d(.,.)$ is a $\mathcal{L}_2$ norm. LBG is an iterative algorithm[35] which produces 2^k prototypes after k iterations. Table 1 provides the synopsis of the LBG

algorithm in which ϵ_j represents the added noise to create two prototypes from existing ones. Those prototypes are used to perform the *clustering* of dataset S with respect to minimal distance ($i.e\ x \in S[j] \rightarrow \forall i \in [0, \ldots, 2^k - 1],\ d(x, E[j]) \leq d(x, E[i])$). In this synopsis, the *centroid* function determines the gravity center of a dataset S.

Table 1. Synopsis of LBG algorithm.

LBG(S, k)
$S'[0] \Leftarrow \text{centroid}(S)$
FOR $i = 1$ **TO** k
FOR $j = 0$ **TO** 2^{k-1}
$E[2j] \Leftarrow S'[j] + \epsilon_j$
$E[2j+1] \Leftarrow S'[j] - \epsilon_j$
ENDFOR
$\{S[0], \ldots, S[2^k - 1]\} \Leftarrow \text{clustering}(S, E)$
FOR $j = 0$ **TO** $2^k - 1$
$S'[j] \Leftarrow \text{centroid}(S[j])$
ENDFOR
ENDFOR
RETURN S'

2.5. *Tabu Search (TS)*

Many meta-heuristics approaches exist to solve hard optimization problems, a set of them are called *tabu search*[40]. Those ones belongs to iterative neighbourhood search methods. The general step, at the it iteration, consists in searching, from a current solution θ^{it}, the next best solution θ^{it+1} in a given neighborhood. This new solution may be less efficient than the previous one; however this avoids local minimum trapping problems. That is why TS uses short memory to avoid moves which might lead to recently visited solutions (*tabu* solutions). TS methods generally incorporate explicit strategies to control the efficiency of the search space exploration. These strategies are grouped in two terms: intensification and diversification. In a promising region of space, the first strategy allows extensive search of the path to find a best solution. However, if the search is in a region of space for which the solutions are poor or if the extensive search cannot produce better solutions, the second strategy enables large changes of the solution in order to find quickly another promising region. These two strategies are generally applied alternatively. Although the basic idea of TS is straightforward, the choice of solution coding, objective functions, neighborhood, *tabu* solutions definition, intensification and diversification strategies, all depend on the application problem.

3. TS Model Selection Method

By studying the SVM formulation problem, one notices that the number of support vectors used by BDF increases with the problem size[6]. As the objective of our model selection is to produce a fast and efficient decision function, increasing the number of support vectors is interesting only if it is linked to a significant improvement in the recognition rate. For the same reason, features selected in a BDF is dependent of recognition rate improvement.

The idea of our method is to produce fast and efficient SVM BDF using few support vectors and few features. To that aim, a new Decision Function Quality (DFQ) criterion, based on regularization theory, has been defined which corresponds to a compromise between efficiency and complexity (*c.f.* Section 3.1). A SVM is therefore trained from a small dataset S'_t representative of the initial training set S_t in order to decrease the complexity of the BDF. The LBG algorithm has been used to perform the simplification of the initial dataset (*c.f.* Section 3.2). As the number of prototypes produced by LBG algorithm (2^k by class) cannot be easily fixed in an arbitrary way, a significant concept in our method is to regard parameter k as a variable of the model selection problem. The optimization of SVM DFQ thus requires, for a given kernel function K, the choice of: the simplification level k, the feature subset β, the regularization constant C and kernel parameters (σ with gaussian kernel). The search of the values of those variables is called model selection. Let θ be a model, k_θ, β_θ, C_θ and σ_θ be respectively the values of all the variables to tune, and $q(\theta)$ be the value of the DFQ criterion for a model θ (*c.f.* Section 3.1). The synopsis in Section 3.3 gives the details of the estimation of DFQ criterion from a model θ and a learning set S_l with $q(\theta) \equiv \text{SVM-DFQ}(\theta, S_l)$ the objective function which must be optimized. The search for the exact θ^* which optimizes $q(\theta)$ not being tractable, we decide to define a specific TS metaheuristic method (*c.f.* Section 3.4) for the model selection problem with adapted intensification and diversification strategies (*cf.* Sections 3.5 and 3.6).

3.1. *Decision Function Quality (DFQ)*

We consider that the DFQ of a given model θ depends on the recognition rate R_R but also on the complexity C_P of the decision function h_θ when processing time is critical. Let $q(\theta) = R_R(h_\theta) - C_P(h_\theta)$ be the DFQ. That definition is based on regularization theory [31,32]. The fitting term is expressed in terms of recognition rate (R_R). The smoothness term is expressed in terms of model complexity (C_P). For SVM, the complexity of the decision function depends on the number of both support vectors and selected features. The empirical model we propose to model the complexity of a SVM BDF is:

$$C_P(h_\theta) = c_{p_1} \log_2(n_{SV}) + c_{p_2} \log_2(\text{cost}(\beta)) \quad (5)$$

β is a boolean vector of size n representing selected features. Constants c_{p_1} and c_{p_2} fix the trade-off between classification rate improvement and complexity reduction. Let κ_i denote the cost for the extraction of the i^{th} feature, the value of $cost(\beta)$ linked to the subset of selected features is defined by: $cost(\beta) = \sum \beta_i \kappa_i$. When those costs are unknown, $\kappa_i = 1$ is used for all features. Strictly speaking, a doubling of the number of support vectors (extraction cost) is accepted in our learning method if it is related to a recognition rate increase by at least c_{p_1} (respectively c_{p_2}).

3.2. *Simplification step*

A natural way to reduce the complexity of SVM decision functions is to reduce SVM training set size. One possibility of doing that is to produce prototypes which efficiently sum up examples close to them. The LBG algorithm (*c.f.* Section 2.4) is used to produce 2^k prototypes for each class into a two class problem. The reduced dataset is a more or less simplified version of the initial one according to the parameter k value. The algorithm in Table 2 gives the details of this simplification (to speed up model selection, at each new value of k, the simplification result is stored for future steps which might use the same simplification level).

Table 2. Synopsis of simplification step.

Simplification(S,k)
$S' \Leftarrow \emptyset$
FOR $c \in \{-1, +1\}$
$T = \{x \mid (x, c) \in S\}$
IF $2^k < \|T\|$ **THEN** $T' \Leftarrow \text{LBG}(T, k)$
ELSE $T' \Leftarrow T$
$S' \Leftarrow S' \cup \{(x, c) \mid x \in T'\}$
ENDFOR
RETURN S'

3.3. *DFQ estimation*

The Decision Function Quality (DFQ) criterion of a specific model θ is evaluated from a learning dataset S_l. The synopsis provided in Table 3 gives details on how the value of that criterion is determined. Let S_t, S_v denote the datasets which are produced by a random split (`Split` function in synopsis SVM-DFQ) with $|S_t| = \frac{2}{3}|S_l|$, $|S_v| = \frac{1}{3}|S_l|$. S_t, S_v are respectively indicate databases used for training SVM (training dataset) and for recognition rate estimation (validation dataset). This dissociation is essential to avoid the risk of overfitting when empirical estimation is used. The SVM training step is realized by using the SMO algorithm version of the Torch library[47]. When SVM training is performed with unbalanced class datasets, it is more suitable to use Balanced Error Rate (BER) instead of classical Error Rate for the estimation of recognition rate. Recognition rate formulation (noted R_R) in Table 3 corresponds to BER estimation where m_y represents the number of examples in each class ($y \in \{+1, -1\}$) and $m_y^{\texttt{correct}}$ the number of examples correctly identified. . The kernel functions K_β used for training SVM are defined from a distance d_β: $d_\beta(x_i, x_j) = \sqrt{\sum_{l=1}^{n} \beta_l (x_i^l - x_j^l)^2}$. By using d_β in the kernel function, the feature selection problem is embedded in the model selection problem. For this study, only Gaussian kernels $K_\beta^G = exp(-d_\beta^2/\lambda_1^2)$ are used.

Table 3. Synopsis of DFQ estimation for a specific model θ.

SVM-DFQ(θ,S_l)
$(S_t, S_v) \Leftarrow \text{Split}(S_l)$
$S'_t \Leftarrow \text{Simplification}(S_t, k_\theta)$
$h_\theta \Leftarrow \text{TrainingSVM}(S'_t, K_{\beta_\theta}, C_\theta, \sigma_\theta)$
$(m_{-1}^{\texttt{correct}}, m_{+1}^{\texttt{correct}}) \Leftarrow \text{TestingBDF}(h_\theta, Sv)$
$R_R \Leftarrow \frac{m_{-1}^{\texttt{correct}}}{2m_{-1}} + \frac{m_{+1}^{\texttt{correct}}}{2m_{+1}}$
$C_P \Leftarrow \text{Complexity}(h_\theta)$
$q(\theta) \Leftarrow R_R - C_P$

3.4. *TS specification*

The objective function q to be optimized represents the quality of the BDF h_θ (*c.f.* Section 3.1). Our problem is to choose an optimal model (good sub-optimal solution to be exact) θ^* for a function q when c_{p1} and c_{p2} are fixed. A model θ can be represented by a set of n' integer variables $\theta = (\theta_1, \ldots, \theta_{n'}) = (\beta_1, \ldots, \beta_n, k, C', \sigma')$. Notations k_θ, β_θ, C_θ and σ_θ used in Section 3.1 correspond respectively to k, $(\beta_1, \ldots, \beta_n)$, $\sqrt{2}^{C'}$ and $\sqrt{2}^{\sigma'}$ in that integer representation of θ model. One basic move in our TS method corresponds to adding $\delta \in [-1, 1]$ to the value of a θ_i, while preserving the constraints of the model which depend on it (*i.e.* $\forall i \in [1, \ldots, n'], \theta_i \in [min(\theta_i), \ldots, \max(\theta_i)]$ where $min(\theta_i)$ and $max(\theta_i)$ respectively denote lower and upper bound values of θ_i variable). From these constraints, the list of all possible neighborhood solutions is computed. From these possible solutions, the one which has the best DFQ and which is not *tabu* is chosen. The set of all Θ_{tabu}^{it} solutions θ which are *tabu* at the *it* iteration step of TS is defined as follows: $\Theta_{tabu}^{it} = \{\theta \in \Omega \mid \exists\, i, t' : t' \in [1, \ldots, t],\ \theta_i \neq \theta_i^{it-1} \wedge \theta_i = \theta_i^{it-t'}\}$ with Ω the set of all solutions and t an adjustable parameter for the short memory used by TS (for experimental results $t = \sum_{i=1}^{n'} \max(\theta_i) - \min(\theta_i)$). The idea is that a variable θ_i could be changed only if its new value is not present in the short memory. Then, our TS method does not go back to a value of θ_i previously changed in short time, avoiding by that mechanism undesirable oscillation effects. Tabu status of solutions Θ_{tabu}^{it} may prohibit some attractive moves at iteration *it*. Therefore, our TS uses an aspiration criterion which consists in allowing a move (even if it is *tabu*) if it results in a solution with an objective value better than that of the current best-known solution.

The initialisation of model θ with our TS model selection is the following:

- $k_\theta = \lfloor \log_2 (\max(m_{+1}, m_{-1}))/3 \rfloor$,
- $C_\theta = 1$ and $\sigma_\theta = 1$,
- $\forall i : \beta_i = 1$.

In the expression of k_θ , m_{+1} and m_{-1} are respectively the number of examples of positive and negative classes in binary sub-problems. The value of k_θ allows to start with sufficiently simplified datasets in order to have low training times with SVM for the first intensification step.

Using intensification and diversification strategies can improve TS methods[40]. The model selection such as it was defined has to deal with two kinds of problems. First, testing all moves between two iterations with a great number of features can be time consuming. In particular, it is a waste of time to explore moves which are linked to features when the actual solution is not sufficiently promising. Therefore, focusing on moves which are only linked to SVM hyperparameters or simplification level is more efficient to discover new promising regions. Second, it is difficult for TS method to quickly escape from deep valleys or big clusters of poor solutions while only using the short memory and resulting in not tabu solutions. Using more diversified solutions can overcome this problem. This is dealt with by increasing step size ($\delta > 1$) of moves and by forcing the use of all types of moves (except feature selection moves for the reason stated above).

In our TS method, intensification and diversification strategies are used alternatively and begin with the intensification strategy. The next two subsections give details on these two strategies.

3.5. *Intensification strategy*

In the intensification algorithm sysnopsis shown in Table 4, `ExtensiveSearch` explores all eligible basic moves, whereas `FastExtensiveSearch` explores only eligible basic moves which are not related to feature selection (*i.e.* changing the value of β). $\eta_{\texttt{promising}}$ controls when the actual solution is considered as sufficiently promising and this one permits to switch between the two functions stated above. `BestNotTabu` corresponds to the move procedure selection described in the previous Section (the best *tabu* solution is chosen if all moves are tabu). In

this synopsis, $\theta_{\texttt{intensification}}$ corresponds to the best solution found into a same phase of intensification, although $\theta_{\texttt{best-known}}$ corresponds to the best solution found in all intensification and diversification steps. $n_{\texttt{max}}$ is the maximum number of intensification iterations for which no improvements of the last best intensification solution ($\theta_{\texttt{intensification}}$) are considered as failure of the intensification strategy. $n_{\texttt{failure}}$ counts the number of failures of intensification strategy. If $n_{\texttt{failure}}$ is higher than a fixed maximum number of failures `max`, our TS method stops and returns the solution $\theta_{\texttt{best-known}}$. If a solution in $\Theta_{\texttt{next}}$ has a QDF which is better than $\theta_{\texttt{best-known}}$, aspiration mechanism is used. That solution is selected as the new $\theta_{\texttt{best-known}}$ and $n_{\texttt{failure}}$ is reset to zero.

Table 4. Synopsis of TS intensification strategy.

Intensification(θ^{it})
IF $q(\theta^{it}) > \eta_{\texttt{promising}} \cdot q(\theta_{\texttt{best-known}})$
THEN $\Theta_{\texttt{next}} \Leftarrow \text{ExtensiveSearch}(\theta^{it})$
ELSE $\Theta_{\texttt{next}} \Leftarrow \text{FastExtensiveSearch}(\theta^{it})$
$\theta^{it+1} \Leftarrow \text{BestNotTabu}(\Theta_{\texttt{next}})$
IF $q(\theta^{it+1}) > q(\theta_{\texttt{intensification}})$
THEN
$\theta_{\texttt{intensification}} \Leftarrow \theta^{it+1}$
$n_{\texttt{WithoutImprove}} \Leftarrow 0$
ELSE
$n_{\texttt{WithoutImprove}} \Leftarrow n_{\texttt{WithoutImprove}} + 1$
IF $n_{\texttt{WithoutImprove}} > n_{\texttt{max}}$
THEN
$n_{\texttt{failure}} \Leftarrow n_{\texttt{failure}} + 1$
`stategy` $\Leftarrow$ `Diversification`
IF $n_{\texttt{failure}} > n^{\texttt{max}}_{\texttt{failure}}$ **THEN** STOP

3.6. *Diversification strategy*

In the diversification algorithm sysnopsis shown in Table 5, an eligible variable (one which does not have a link with features) is selected (`SelectEligibleVariable`) by random and a jump of $\pm\delta$ is performed by modifying the selected variable in the actual solution. There are only two explored moves (`TwoMove`) to force the diversification of explored solutions. The jump size increases with the number of successive failures ($n_{\texttt{failure}}$) of the intensification strategy in order to explore more and more distant regions. During the diversification iterations, the best visited solution is stored ($\theta_{\texttt{diversification}}$) and selected as the starting solution for the next intensification step ($\theta^{it}_{\texttt{intensification}} = \theta^{it-1}_{\texttt{diversification}}$). At any time of TS exploration, if aspiration is involved, the strategy automatically switches to intensification and the number of failures is reset ($n_{\texttt{failure}} = 0$).

Table 5. Synopsis of TS diversification strategy.

Diversification(θ^{it})
$\delta \Leftarrow n_{\texttt{failure}+1}$
$i \Leftarrow$ `SelectEligibleVariable`
$\Theta_{\texttt{next}} \Leftarrow \texttt{TwoMove}(\theta^{\texttt{it}}, \texttt{i}, \delta)$
$\theta^{it+1} \Leftarrow \text{BestNotTabu}(\Theta_{\texttt{next}})$
IF $q(\theta^{it+1}) > q(\theta_{\texttt{diversification}})$
THEN
$\theta_{\texttt{diversification}} \Leftarrow \theta^{it+1}$
$n_{\texttt{diversification}} \Leftarrow n_{\texttt{diversification}} + 1$
IF $n_{\texttt{diversification}} > n_{\texttt{max}} \cdot n_{\texttt{failure}}$
THEN
$\theta^{it+1} \Leftarrow \theta_{\texttt{diversification}}$
`stategy` $\Leftarrow$ `Intensification`

4. Experimental results

Two types of experiments were performed. In the first one, the abilities of our TS model selection were tested on well-known benchmark datasets[29]. In the second one, the goal was to produce a fast pixel classification by combining several SVM inducers of low complexities[28]. Pixel classifiers were used to define a fast and efficient segmentation scheme for cell microscopic image[30].

4.1. *Benchmark datasets*

For the experiments, datasets *Adults*, *OpticDigit*, *Letter* and *Shuttle* are from UCI repository[54]; dataset *Web* comes from Platt experiments[3]; and dataset *ClassPixel* comes from our works[30]. Table 6 provides statistics on those datasets with m, n_c and n_f respectively the number of examples, the number of classes and the number of features (learning and test sets respectively have 2/3 and 1/3 of datasets size. Test sets are used to estimate recognition rate (R_R) after TS model selection step).

Table 6. Datasets description.

bases	m	n_c	n_f
Adults	45222	2	103
OpticDigit	3823	10	64
Letter	20000	26	16
Shuttle	58000	6	9
Web	49749	2	300
ClassPixel	224636	3	27

4.2. *Influence of simplification level*

To illustrate the importance of simplification step in our model selection method, this one is tested when the value of k is fixed for the TS process (feature selection possibility of our TS method is not used by forcing FastExtensiveSearch in our intensification strategy). The idea is to observe the evolution of BDF recognition rate according to simplification intensity. Table 7 summaries results obtained with the *OpticDigit* dataset for $k \in \{0, 2, 4, 8\}$ (this corresponds to have respectively $\{1, 4, 16, 64\}$ prototype examples by class). Those results show that each individual binary sub-problem is more or less sensitive to simplification level in a same multi-class problem. For several binary sub-problems, few prototypes (*i.e.* low values of k) are sufficient to produce efficient BDF with SVM learners. The selection of the simplification level (k value) is then dependent on each binary sub-problem in a multi-class problem when QFD criterion must be optimized. The choice of complexity penalty coefficient (c_{p_1}) also has an impact on selection of the optimal simplification level. For example with Table 7 results, if c_{p_1} is fixed to 0.01, the k values which optimize DFQ criterion are respectively $\{2, 4, 4, 4, 8\}$. If c_{p_1} is changed to 0.002, the k optimal values become $\{4, 6, 4, 8, 8\}$.

Table 7. Results with ***OpticDigit*** dataset for different simplification levels (value of k) on some binary sub-problems involved in an *one-versus-all* decomposition. R_{R_i} is the recognition rate on test set for produced BDF in which i represents the digit in the binary sub-problem to identify for the others. C_{p_1} is fixed to 0.01.

k	R_{R_0}	R_{R_2}	R_{R_5}	R_{R_8}	R_{R_9}
0	98.4%	95.4%	92.8%	87.8%	88.5%
2	99.7%	97.3%	95.7%	91.5%	90.7%
4	100.0%	99.6%	99.1%	96.2%	92.6%
6	100.0%	99.8%	99.2%	96.4%	95.8%
8	100.0%	99.9%	99.2%	96.6%	97.7%

Similar results are obtained with other datasets[28,29]. From all those experiments one can conclude that increasing the training set size (number of prototypes) does not alway significantly improve the recognition rate for a specific binary problem. The main reason of that effect is that the level of redundancy in a dataset is variable. Moreover, complexity of a produced BDF is directly linked to training set size[28,29]. Those preliminary results show the importance of using DFQ criterion as an objective function because several θ models have very close recognition rates but great variations on their complexities.

4.3. *Tuning parameters of our TS method*

The objective of our model selection method is to automatically select efficiently all free parameters (θ model) involded in the construction process of SVM BDF in order to optimize DFQ, but our TS method also has to introduce other free parameters ($\eta_{\texttt{promising}}$, $n_{\texttt{max}}$, $n^{\texttt{max}}_{\texttt{failure}}$). Several experiments have been realized to determine efficient setting for them[29]. Results from: (1) $\eta_{\texttt{promising}} = 0.99$ is an efficient threshold value to determine if TS basic moves must incorporate feature selection possibility, (2) $n_{\texttt{max}} = 5$ and $n^{\texttt{max}}_{\texttt{failure}} = 5$ are a good compromise between learning time reduction and the importance of TS exploration for determining when TS must be terminated.

4.4. *Results with different datasets*

We applied the TS model selection described in Section 3 with settings given in Section 4.3. Two penalty configurations are used: (1) $c_{p_1} = c_{p_2} = 0.01$, (2) $c_{p_1} = 0.0001$ and $c_{p_2} = 0$. For the first configuration, doubling the number of support vectors or the number of features used is only profitable if recognition rate is increased of at least 1%. For the second configuration, the idea is to be very close to classical SVM training: SVM hyper-parameters selection corresponding to minimizing expected error rate (c_{p_1} is not equal to zero, but close to it, in order to avoid intractable SVM training time) with no feature selection possibility (*i.e* only FastExtensiveSearch with Intensification strategy). Each penalty configuration is used with datasets presented in Section 4.1.

Table 8. Total training time ($\sum TT$), mean of simplification level ($\bar{k}$), total number of support vectors ($\sum SV$), average number of features ($\bar{n}_f$) and recognition rate on test set ($\bar{R}_R$) for the n_c produced BDF (only one BDF is produced when $n_c = 2$) with our TS model selection method. Two penalty configurations are used with the benchmark datasets: $c_{p_1} = c_{p_2} = 0.01$ for config. 1 and $c_{p_1} = 0.0001$ and $c_{p_2} = 0$ for config. 2.

Config. 1	$\sum TT$	$\bar{k}$	$\sum SV$	$\bar{n}_f$	$\bar{R}_R$
Adult	5634	0	2	44	81.5%
OpticDigit	3569	3.7	143	20.6	97.4%
Letter	31478	4.7	237	9.3	92.8
Shuttle	628	2.3	27	1.3	99.9%
Web	25693	2	5	149	87.3%
ClassPixel	18557	3.3	33	6	84.8%
Config. 2	$\sum TT$	$\bar{k}$	$\sum SV$	$\bar{n}_f$	$\bar{R}_R$
Adult	21749	14	23698	103	84.8%
OpticDigit	134	8.6	134	64	99.0%
Letter	42127	9.3	5612	16	94.2%
Shuttle	128174	10.5	285	9	99.9%
Web	18127	11	730	300	90.4%
ClassPixel	31282	4.3	59	27	85.0%

Table 8 reports results obtained by each TS model selection. Those results show that recognition rate of BDF with configuration 1 are close to those with configuration 2. Although, for the first one, the model complexity is greatly decreased by reducing both the number of support vectors and features used by a BDF. More penalty configurations have been also tested on those classification problems[29]; results show that our TS model selection method can produce a range of BDF which have different trade-off between celerity and precision. Of course, the good penalty trade-off is application dependant, but results in Table 8 show that our method can produce, for a same penalty configuration, fast BDF which have globally efficient generalization capacities through datasets of different natures.

If training times are compared to the classical *grid-search* technique (a grid points in the kernel-parameter-and-C plane[19]) without simplification of training set[29], training time is greatly reduced (except with very low penalty values). Moreover, our method can also perform in addition feature and simplification level selections. Results in Table 8 show that the number of features used has been greatly reduced when configuration 1 and configuration 2 are compared. Time to proceed at model selection including feature selection (*i.e.* configuration 1) is tractable with our TS method if penalty coefficients are not too low (results, not presented here[29], show that model selection time increases quickly while penalty coefficient decreases). In the other case (*i.e.* configuration 2), feature selection possibility must be discarded. Let n_k be the number of solutions θ examined by TS for which simplification level is equal to k. Global SVM training time of our method is $O(\sum n_k(2^k)^\gamma)$ with $2 < \gamma < 3$. The examination of our TS method shows that n_k decreases while k increases. This effect increases when c_p values increases and explains the efficient training time of our TS model selection.

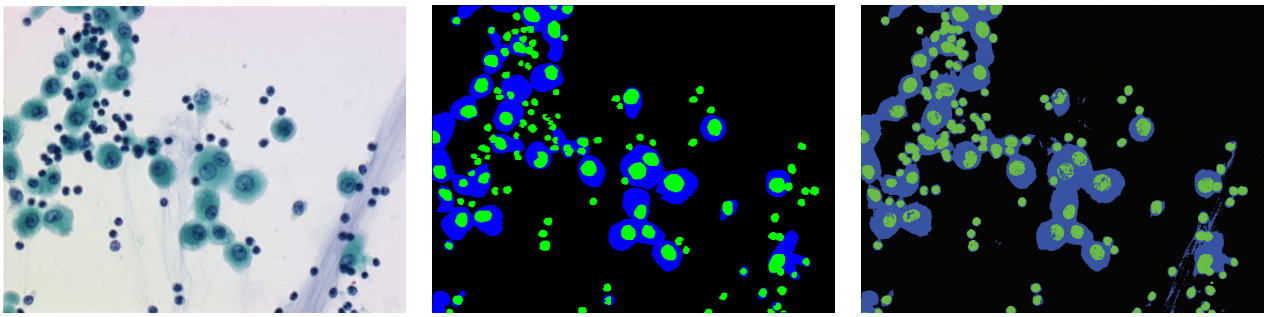
4.5. *Fast pixel classification problem*

Pixel classification is commonly used as an initial step in color image segmentation schemes[43,55,28] for the extraction of seeds. As for any classification problem, the choice of an inducer which produces efficient decision functions having good generalization performances is critical. Working with any machine learning algorithm for pixel classification involves to take into account not only the recognition rate of the base inducer but also the processing time needed to perform pixel classification of all the pixels in an image. About millions of accesses to SVM inducers per image are necessary for that kind of application[30]. Therefore, our model selection method is adapted to produce fast and efficient pixel classification. Vector quantization is used in order to reduce the inherent redundancy present in huge pixel databases. Feature selection in our method is used to select an adapted hybrid color space[43,28].

The **ClassPixel** dataset is built from 8 microscopic images of bronchial tumors where ground truth is given by experts[30] (see fig. 1(a) and 1(b) which show respectively a cellular image and its corresponding expert segmentation). Results in Table 8 show that it is possible to produce efficient BDF with low complexity for that problem. If decomposition of combination scheme is changed to *one-versus-one* (c_p coefficients are also changed), it is possible to increase recognition rate ($R_R = 86.7\%$) and decrease complexity ($\sum SV = 10$, $\bar{n}_f = 3$) of pixel classifiers. Figure 1(c) illustrates pixel classification for a cellular image (fig. 1(a)) when *one-versus-one* combination scheme is used. Comparing the pixel classification result (fig. 1(c)) to the expert segmentation (fig. 1(b)) shows that shapes of nucleus and cytoplasm are well identified. Additional results show that pixel classification permits to produce a fast and efficient segmentation scheme for cell microscopic image[30].

5. Conclusion

A new learning method based on SVM inducers



(a) (b) (c) pixel classification.

Figure 1: (a) Microscopic cellular image (RGB, 752x574 pixels) stained with international coloration of Papanicolaou. (b) expert segmentation of the microscopic image: background (black), cytoplasm (blue) and nuclei (green). (c) pixel classification results with BDF produced by our TS model selection method.

is proposed to achieve a good compromise between fast decision process and precision of that decision. To that aim, we have proposed:

1. To use VQ technique, through the LBG algorithm, to produce prototypes which resume efficiently examples in a training dataset. Motivation was that training datasets generally have redundancy.

2. To include feature selection possibility to that learning method in order to deal with more or less correlation in the set of features describing training examples. The irrelevant features for binary sub-problems induced by multi-class decomposition can also be discarded by feature selection process.

3. To realize the selection of efficient SVM hyperparameters in order to increase generalization capacities of that type of inducers.

4. To define a quality criterion, called DFQ, which corresponds to a trade-off between low complexity and precision of a SVM decision process.

5. To define an adapted TS model selection which efficiently tunes parameters linked to the first three key points in order to optimize the DFQ criterion.

The objectives of the proposed learning method based on TS model selection is to produce BDF which have threefold advantages: high generalization abilities, low complexities and selection of an efficient features subsets. Experimental results on benchmark datasets illustrate that our TS selection method realizes those objectives. It also shows how to efficiently fix internal parameters of our TS method. Other experimental results for a cellular microscopic segmentation application shows that pixel classification can be fast and efficient. Resulting segmentations will be helpful for analysis, in particular for cancerous diagnostic-helping.

6. Future works

Future works have to deal with two topics. In the first one, we want to improve our model selection method with SVM inducers. In the second one, we have to compare the proposed method with other methodologies.
To improve the proposed TS model selection, several directions could be explored:

- Taking into account multi-class recognition rate and the total complexity of all BDF (induced by the binary decomposition) in the DFQ criterion. Recent results[27] show that optimizing individually each BDF implied in a combination scheme does not necessary produce the optimal multi-class scheme. Moreover, in our case, reducing the complexity of a specific BDF could have a significant decrease of the binary recognition rate for that BDF, but a lesser significant decrease of the multi-class recognition rate for the combination scheme in which that BDF is used. More generally, this problem refers to the question on how to combine several classifiers together in order to achieve improved performance[56].

- Extending TS model selection in order to have an efficient feature selection for the global multi-class problem. Indeed, feature selection is independently well tuned for a specific BDF, but the union of used features by all BDF in a multi-class decomposition is generally more important than the average of those ones. New type of TS moves must be defined to perform an efficient multi-class feature selection which preserves the global multi-class recognition rate for a given combination scheme, but makes sure that the union of every selected feature subset has the smallest possible size. Complexity term in the DFQ criterion must be also changed to favor that possibility.

- The influence of other simplification methods[4,12,11] has to be quantized. In particular, QV methods can be time consuming when datasets have million of examples. For instance, using hierarchical clustering tree algorithms can speed up this simplification step. Another way is to directly work with several pruned versions of hierarchical tree database representations[4]. New moves for our TS model selection must be defined in order to permit to un-prune or prune the tree according to the promising status of regions and the complexity variations of DFQ criterion.

Comparison with other methodologies could be devised in two ways:

- SVM algorithms have efficient generalization propriety in the machine learning framework but other learning algorithms like Neural Networks (NN) or decision trees also have that propriety[57,58,59,60,61]. It will be interesting to substitute SVM for one of them. The problem is how to compare complexity of those different inducers. For instance with NN, reducing the number of examples in training set has no impact on NN complexity, but reducing the number of neurons used with a NN has a great impact.

- Other meta-heuristic methods exist to optimize DFQ criterion like simulated annealing, evolutionary, particle swarm or ant colony algorithms[33,34]. It will be interesting to compare them on the model selection problem. In terms of process time. Another interesting possibility, with meta-heuristic algorithms that find multiple solution[34], is that a set of efficient models could then be produced in the framework of multi-objective optimization. Different realizations of the compromise between low complexity and precision could be produced without explicitly fixing c_p coefficient values in QDF criterion. The selection of the classifier which has the best compromise will be determined later in function of application constraints.

Acknowledgements

This work was supported by Université de Caen Basse-Normandie under grants of Coeur-Cancer association and the Low Normandy council funds. It was developed in the "Service d'Anatomie et de Cytologie Pathologiques de l'Hôpital Pasteur de Cherbourg". The authors thank their technical staff for time spent to produce manual segmentations of cell microscopic images.

References

1. V. N. Vapnik. *Statistical Learning Theory*. New York, wiley edition, 1998.
2. N. Cristianini and J. Shawe-Teylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.
3. J. Platt. Fast training of SVMs using sequential minimal optimization, advances in kernel methods-support vector learning. *MIT Press*, pages 185–208, 1999.
4. H. Yu, J. Yang, and J. Han. Classifying large data sets using SVM with hierarchical clusters. In *SIGKDD*, pages 306–315, 2003.
5. G. Lebrun, C. Charrier, and H. Cardot. SVM training time reduction using vector quantization. In *ICPR*, volume 1, pages 160–163, 2004.
6. I. Steinwart. Sparseness of support vector machines - some asymptotically sharp bounds. In *NIPS*, pages 169–184, 2004.
7. S. S. Keerthi, O. Chapelle, and D. DeCoste. Building support vector machines with reduced classifier complexity. *JMLR*, 7:1493–1515, 2006.
8. D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *TPAMI*, 26(5):530–549, 2004.
9. C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. Sofware Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

10. S. V. N. Vishwanathan, Alex J. Smola, and M. Narasimha Murty. SimpleSVM. In *ICML*, pages 760–767, 2003.
11. Y. Y. Ou, C. Y. Chen, S. C. Hwang, and Y. J. Oyang. Expediting model selection for SVMs based on data reduction. In *IEEE Proc. SMC*, pages 786–791, 2003.
12. I. W. Tsang, J. T. Kwok, and P.-M. Cheung. Core vector machines: Fast SVM training on very large data sets. *JMLR*, 6:363–392, 2005.
13. E. Parrado-Hernndez, I. Mora-Jimnez, J. Arenas-Garca, A. R. Figueiras-Vidal, and A. Navia-Vzquez. Growing support vector classifiers with controlled complexity. *Pattern Recognition*, 36(7):1479–1488, 2003.
14. J. Yang, Z.-W. Li, and J.-P. Zhang. A training algorithm of incremental support vector machine with recombining method. In *Machine Learning and Cybernetics*, volume 7, pages 4285–4288, 2005.
15. S. Katagiri and S. Abe. Incremental training of support vector machines using hyperspheres. *Pattern Recogn. Lett.*, 27(13):1495–1507, 2006.
16. R. Herbrich. *Learning Kernel Classifiers*. The MIT Press, 2002.
17. K. Lin and C. Lin. A study on reduced support vector machines. *Neural Networks*, 14(6):1449–1507, 2003.
18. T. Thies and F. Weber. Optimal reduced-set vectors for support vector machines with a quadratic kernel. *Neural Comput.*, 16(9):1769–1777, 2004.
19. S. Abe. *Support Vector Machines for Pattern Classification*. Springer, 2005.
20. N. Christianini. Dimension reduction in text classification with support vector machines. *JMLR*, 6:37–53, 2005.
21. O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.
22. H. Fröhlich, O. Chapelle, and B. Schölkopf. Feature selection for support vector machines using genetic algorithms. *International Journal on Artificial Intelligence Tools*, 13(4):791–800, 2004.
23. J. Bi. Multi-objective programming in SVMs. In *ICML*, pages 35–42, 2003.
24. O. Chapelle and V. Vapnik. Model selection for support vector machines. In *Advances in Neural Information Processing Systems*, volume 12, pages 230–236, 1999.
25. H. Nakayama, Y. Yun, T. Asada, and M. Yoon. Mop/gp models for machine learning. *European Journal of Operational Research*, 166(3):756–768, 2005.
26. R. Rifkin and A. Klautau. In defense of one-vs-all classification. *JMLR*, 5:101–141, 2004.
27. G. Lebrun, O. Lezoray, C. Charrier, and H. Cardot. *An EA Multi-model selection for SVM Multiclass schemes, Encyclopedia of Artificial Intelligence*. Information Science Reference, J. R. Rabual and J. Dorado and A. Pazos edition, 2008.
28. G. Lebrun, C. Charrier, O. Lezoray, C. Meurie, and H. Cardot. Fast pixel classification by SVM using vector quantization, tabu search and hybrid color space. In *CAIP*, pages 685–692, 2005.
29. G. Lebrun, O. Lezoray, C. Charrier, and H. Cardot. A new model selection method for SVM. In *IDEAL*, pages 99–107, 2006.
30. G. Lebrun, C. Charrier, O. Lezoray, C. Meurie, and H. Cardot. A fast and efficient segmentation scheme for cell microscopic image. *Cellular and Molecular Biology, special issue on signal and image processing*, 53(2):51–61, 2007.
31. A. Tikhonov and V. Arsenin. *Solution of Ill-posed Problems*. Winston & Sons, 1977.
32. A. Tikhonov and V. Arsenin. *Ill-Posed Pproblems: Theory and Applications*. Kluwer Academic Publishers, 1994.
33. C. A. C Coello and G. B. Lamont D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (volume 5)*. Kluwer Academic, 2002.
34. A. P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*. Wiley, 2006.
35. A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic, 1991.
36. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor Morgan Kaufmann Publishers, 2000.
37. T. Acharya S. Mitra. *Data mining: multimedia, soft computing and bioinformatics*. John Wiley and Sons, 2003.
38. P. N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2006.
39. J.-X. Dong, A. Krzyzak, and C. Y. Suen. An improved handwritten chinese character recognition system using support vector machine. *Pattern Recognition Letters*, 26(12):1849–1856, 2005.
40. F. Glover and M. Laguna. *Tabu search*. Kluwer Academic Publishers, 1997.
41. D. Korycinski, M. M. Crawford, and J. W. Barnes. Adaptive feature selection for hyperspectral data analysis. *SPIE*, 5238:213–225, 2004.
42. Ping-Feng Pai and Yu-Ying Huang. Using directed acyclic graph support vector machines with tabu search for classifying faulty product types. In *ISNN (2)*, pages 1117–1125, 2006.
43. N. Vandenbroucke, L. Macaire, and J.-G. Postaire. Color image segmentation by pixel classification in an adapted hybrid color space: application to soccer image analysis. *Comput. Vis. Image Underst.*, 90(2):190–216, 2003.
44. C. Meurie, G. Lebrun, O. Lezoray, and A. Elmoataz. A supervised segmentation scheme for cancerology color images. In *ISSPIT*, pages 664–667, 2003.
45. C. Meurie, O. Lezoray, C. Charrier, and A. Elmoataz. Combination of multiple pixel classifiers for microscopic image segmentation. *IJRA*, 20(2):63–69, 2005. Special issue on Colour Image Processing and Analy-

sis for Machine Vision, ISSN 0826-8185.
46. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-bases learning methods*. Cambridge University Press, 2000.
47. R. Collobert and S. Bengio. SVMTorch: Support vector machines for large-scale regression problems. In *Journal of Machine Learning Research*, volume 1, pages 143–160, 2001.
48. J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A. J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, 1999.
49. D. Price, S. Knerr, L. Personnaz, and G. Dreyfus. Pairwise neural network classifiers with probabilistic outputs. In *NIPS*, pages 1109–1116, 1994.
50. T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *NIPS*, pages 507–513, 1997.
51. T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *JAIR*, 2:263–286, 1995.
52. M. Moreira and E. Mayoraz. Improved pairwise coupling classification with correcting classifiers. In *ECML*, pages 160–171, 1998.
53. T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *JMLR*, 5:975–1005, 2004.
54. C. Blake and C. Merz. UCI repository of machine learning databases. advances in kernel methods, support vector learning., 1998.
55. C. Meurie, G. Lebrun, O. Lezoray, and A. Elmoataz. A comparison of supervised pixels-based color image segmentation methods. application in cancerology. In *WSEAS Transactions on Computers*, volume 2, pages 739–744, 2003.
56. L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, 2004.
57. C. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.
58. S. Haykin. *Neural Networks: a comprehensive foundation*. Tom Robbins, 1999.
59. L. Breiman, J. Freidman, R. Olshen, and C. Stone. *Classification And Regression Trees*. Wadsworth and Brooks, 1984.
60. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
61. V. Kecman. *Learning and soft computing*. MIT Press, 2001.