Research Article

A Faster Gradient Ascent Learning Algorithm for Nonlinear SVM

Catalina-Lucia Cocianu,¹ Luminita State,² Marinela Mircea,¹ and Panayiotis Vlamos³

¹ Department of Informatics and Cybernetics, Bucharest University of Economic Studies, Dorobanți 15-17, 010552 Bucharest, Romania

² Department of Mathematics and Informatics, University of Pitesti, Târgu din Vale No. 1, 110040 Pitesti, Romania

³ Department of Informatics, Ionian University, Palaia Anaktora 49100 Corfu, Greece

Correspondence should be addressed to Luminita State; lstate@clicknet.ro

Received 27 May 2013; Accepted 19 July 2013

Academic Editors: S. He and L. Wu

Copyright © 2013 Catalina-Lucia Cocianu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We propose a refined gradient ascent method including heuristic parameters for solving the dual problem of nonlinear SVM. Aiming to get better tuning to the particular training sequence, the proposed refinement consists of the use of heuristically established weights in correcting the search direction at each step of the learning algorithm that evolves in the feature space. We propose three variants for computing the correcting weights, their effectiveness being analyzed on experimental basis in the final part of the paper. The tests pointed out good convergence properties, and moreover, the proposed modified variants proved higher convergence rates as compared to Platt's SMO algorithm. The experimental analysis aimed to derive conclusions on the recognition rate as well as on the generalization capacities. The learning phase of the SVM involved linearly separable samples randomly generated from Gaussian repartitions and the WINE and WDBC datasets. The generalization capacities in case of artificial data were evaluated by several tests performed on new linearly/nonlinearly separable data coming from the same classes. The tests pointed out high recognition rates (about 97%) on artificial datasets and even higher recognition rates in case of the WDBC dataset.

1. Introduction

According to the theory of SVMs, while traditional techniques for pattern recognition are based on the attempt to optimize the performance in terms of the empirical risk, SVMs minimize the structural risk, that is, the probability of misclassifying yet-to-be-seen patterns for a fixed but unknown probability distribution of data [1–4]. The most distinguished and attractive features of this classification paradigm are the ability to condense the information contained by the training set and the use of families of decision surfaces of the relatively low Vapnik-Chervonenkis dimension.

SVM approaches to classification lead to convex optimization problems, typically quadratic problems in a number of variables equal to the number of examples, and these optimization problems become challenging when the number of data points exceeds few thousands.

For making SVM more practical, several algorithms have been developed such as Vapnik's chunking and Osuna's

decompositions [1, 5]. They make the training of SVM possible by breaking the large QP problem into a series of smaller QP problems and optimizing only a subset of training data patterns at each step. Because the subset of training data patterns optimized at each step is called the working set, these approaches are referred to as the working set methods.

Recently, a series of works on developing specializations as, for instance, reduced support vector machines (RSVM) [6] and smooth support vector machines (SSVM) [7] as well as parallel implementations of training SVM's have been proposed [8]. Also, there have been proposed methods to solve the least squares SVM formulations [7–10] as well as software packages as SVM^{light} [11], *mysvm* [12], and many others [3, 11, 13–15]. It is worth to mention that a series of developments aimed to improve the accuracy of the resulted SVM classifier by combining it with boosting-type techniques [16, 17].

Assume that the data is represented by a finite set of labeled examples $S_N = \{(x_i, y_i), x_i \in \mathbf{R}^n, y_i \in \{-1, 1\}, 1 \leq i \leq N\}$ coming from two pattern classes h_1, h_2 , and $g: \mathbf{R}^n \rightarrow \mathbf{R}^m$

is a vector-valued function representing the filter that extracts information from the current input. The function g is usually referred to as the feature extractor, and $\mathbf{R}^m$ is thought as the feature space. Briefly, from mathematical point of view, the problem of determining the parameters of an optimal margin classifier reduces to the quadratic programming (QP) problem [3]:

$$\text{minimize } \frac{1}{2} \|w\|^2, \quad (1)$$

$$y_i (w^T g(x_i) + b) \geq 1, \quad 1 \leq i \leq N.$$

If (w^*, b^*) is a solution of (1), then the SVM classifier corresponds to the decision rule $D : \mathbf{R}^n \rightarrow \{h_1, h_2, U\}$, where

$$\text{IF } w^{*T} g(x) + b^* > 0 \text{ THEN } D(x) = h_1,$$

$$\text{IF } w^{*T} g(x) + b^* < 0 \text{ THEN } D(x) = h_2,$$

$$\text{IF } w^{*T} g(x) + b^* = 0 \text{ THEN } D(x) = U,$$

and U means “unclassifiable.”

Using the Lagrange multiplier method, the QP problem (1) reduces to the QP problem

$$\begin{aligned} \text{maximize } Q(\alpha) &= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j g(x_i)^T g(x_j), \\ \sum_{i=1}^N \alpha_i y_i &= 0, \\ \alpha_i &\geq 0, \quad 1 \leq i \leq N. \end{aligned} \quad (2)$$

The parameter b cannot be explicitly computed by solving the SVM problem, a convenient choice of b being derived in terms of the support vectors. Usually, a suitable value of b should be selected such that $1 - \min_{i, y_i=1} w^{*T} g(x_i) \leq b^* \leq -1 - \max_{i, y_i=-1} w^{*T} g(x_i)$ holds, for instance [3],

$$b^* = -\frac{1}{2} \left\{ \max_{i, y_i=-1} w^{*T} g(x_i) + \min_{i, y_i=1} w^{*T} g(x_i) \right\}. \quad (3)$$

In our work, we prefer to use a value of the parameter b^* computed on a heuristic basis aiming to take into account the available information about the variability of the subsamples coming from the classes [18].

The performance of the resulted classifier is essentially determined by the quality of the feature extractor g , the main problem becoming the design of a particular informational feature extractor. One way in over passing this difficulty is the “kernel trick.” Basically, the method consists in selecting a suitable kernel that, on one hand, “hides” the explicit expression of g and, on the other hand, allows working in a feature space of possibly very high dimension without increasing the computational complexity [3]. Usually, we assume a certain particular functional expression for the kernel that “hides” both the dimension of the feature space and the explicit expression of the feature extractor g .

If we assume that the kernel $K : \mathbf{R}^n \times \mathbf{R}^n \rightarrow [0, \infty)$ is given by $K(x, x') = g(x)^T g(x')$ for a certain feature extractor g , then K is a semidefined Mercer kernel [4]. Then, the problem of determining the parameters of an optimal margin classifier corresponds to solving the optimization problem (2), where the objective function is

$$Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(x_i, x_j). \quad (4)$$

In our work, we use exponential type kernels (RBF) and develop a modified gradient ascent method for solving the QP problem (2). The refinement considered in our developments comes from the use of weights in determining the direction of the search displacement at each step in order to get a better tuning to the particular training sequence. We propose three attempts in determining the weights, partially heuristically, and their corresponding performance is experimentally analyzed in the final section of the paper. In our developments, we implemented an SVM classifier of SVM^{light} type [11].

2. Modified Gradient Ascent Method for Learning Nonlinear SVM Purposes

In [19], we proposed a modified learning rule of gradient ascent type for linear SVM that can be extended to the nonlinear case as follows. Assume that $S_N = \{(x_i, y_i), x_i \in \mathbf{R}^n, y_i \in \{-1, 1\}, 1 \leq i \leq N\}$ is a set of labeled examples coming from the classes h_1, h_2 . Conventionally, the first m examples come from the class labeled by 1, the rest of examples being labeled by -1 . By straightforward computations, the expressions of the gradient $\nabla_\alpha Q(\alpha)$ and the Hessian matrix $H(Q(\alpha)) = \|\partial^2 Q(\alpha) / \partial \alpha_k \partial \alpha_p\|_{k,p}$ are

$$\begin{aligned} \frac{\partial Q(\alpha)}{\partial \alpha_k} &= 1 - y_k \sum_{i=1}^N \alpha_i y_i K(x_k, x_i), \quad 1 \leq i \leq N, \\ \frac{\partial^2 Q(\alpha)}{\partial \alpha_k \partial \alpha_p} &= -y_p y_k K(x_k, x_p). \end{aligned} \quad (5)$$

Note that $H(Q(\alpha))$ is a negative semidefined matrix.

For a given learning rate $\rho > 0$, if α^{old} is the current value of the parameter α , the updating rule of a gradient type learning algorithm is

$$\alpha = \alpha^{\text{old}} + \rho \nabla_\alpha Q(\alpha) \Big|_{\alpha=\alpha^{\text{old}}}. \quad (6)$$

However, being given that $\sum_{i=1}^N \alpha_i y_i = 0$, the updating rule (6) should be modified such that to assure that the new parameter still belongs to the space of the feasible solutions. Our method can be briefly described as follows. Assume that p_1, p_2 are the components of the current parameter vector α^{old} selected for being updated, $1 \leq p_1 \leq m, m+1 \leq p_2 \leq N$. If ρ_1 is a weighting parameter expressing the relative “influence” of $\partial Q(\alpha) / \partial \alpha_{p_1} \Big|_{\alpha=\alpha^{\text{old}}}$ and $\partial Q(\alpha) / \partial \alpha_{p_2} \Big|_{\alpha=\alpha^{\text{old}}}$ to the direction of the updating displacement, then the updated

parameter satisfying the constraint $\sum_{i=1}^N \alpha_i y_i = 0$ is $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_N)^T$, where

$$\begin{aligned} \alpha_i &= \alpha_i^{\text{old}}, \quad 1 \leq i \leq N, \quad i \neq p_1, p_2, \\ \alpha_{p_i} &= \alpha_{p_i}^{\text{old}} + \rho \left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}} \Big|_{\alpha=\alpha^{\text{old}}} + (1 - \rho_1) \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}} \Big|_{\alpha=\alpha^{\text{old}}} \right], \\ 0 &\leq \rho_1 \leq 1, \quad i = 1, 2. \end{aligned} \quad (7)$$

The indices p_1, p_2 involved in the updating step should be selected such that to assure the local maximization of $Q(\alpha) - Q(\alpha^{\text{old}})$. Using first-order approximations, the pair of indices (p_1, p_2) should satisfy the conditions [19]

$$\begin{aligned} 1 &\leq p_1 \leq m, \quad m+1 \leq p_2 \leq N, \\ \rho &\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}} \Big|_{\alpha=\alpha^{\text{old}}} + (1 - \rho_1) \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}} \Big|_{\alpha=\alpha^{\text{old}}} \right] \\ &\times \left(\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}} \Big|_{\alpha=\alpha^{\text{old}}} + \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}} \Big|_{\alpha=\alpha^{\text{old}}} \right) > 0. \end{aligned} \quad (8)$$

Therefore, one has to pick up a pair satisfying (8) for which $Q(\alpha) - Q(\alpha^{\text{old}})$ is maximized.

The stopping condition for the search process is controlled by a given threshold $\varepsilon > 0$, and it holds when at least one of the conditions (9) is satisfied:

$$\begin{aligned} \rho &\left[\rho_1 \frac{\partial Q(\alpha)}{\partial \alpha_{p_1}} \Big|_{\alpha=\alpha^{\text{old}}} + (1 - \rho_1) \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}} \Big|_{\alpha=\alpha^{\text{old}}} \right] \\ &\times \left(\frac{\partial Q(\alpha)}{\partial \alpha_{p_1}} \Big|_{\alpha=\alpha^{\text{old}}} + \frac{\partial Q(\alpha)}{\partial \alpha_{p_2}} \Big|_{\alpha=\alpha^{\text{old}}} \right) \leq 0, \\ \forall 1 &\leq p_1 \leq m, \quad m+1 \leq p_2 \leq N, \\ |Q(\alpha) - Q(\alpha^{\text{old}})| &< \varepsilon. \end{aligned} \quad (9)$$

3. Strategies for Determining the Influence Weight in Implementing the Gradient Ascent Search

The option on the values of the learning rate ρ and the weight parameter ρ_1 should be such that to assure good performance from both point of views, accuracy and efficiency. In our tests, we used $\rho \in [10^{-4}, 0.8]$.

Our research focused on several ways to compute the weight ρ_1 , all of them being expressed in terms of the first- and second- order statistics computed in the feature space. Let $\hat{\mu}_{g,1}, \hat{\mu}_{g,2}, \hat{\Sigma}_{g,1}, \hat{\Sigma}_{g,2}$ be the sample means and sample covariance matrices computed on the basis of the samples labeled by 1 and -1 in the feature space, where g is a particular feature extractor. We denote by K the kernel generated by g , that is, $K(x, x') = g(x)^T g(x')$. Since we assumed the

first m examples as coming from the first class and the next $N - m$ examples as coming from the second class, we get

$$\begin{aligned} \hat{\mu}_{g,1} &= \frac{1}{m} \sum_{i=1}^m g(x_i), \quad \hat{\mu}_{g,2} = \frac{1}{N-m} \sum_{i=1}^{N-m} g(x_{m+i}), \\ \hat{\Sigma}_{g,1} &= \frac{1}{m-1} \sum_{i=1}^m (g(x_i) - \hat{\mu}_{g,1})(g(x_i) - \hat{\mu}_{g,1})^T, \\ \hat{\Sigma}_{g,2} &= \frac{1}{N-m-1} \sum_{i=1}^{N-m} (g(x_{m+i}) - \hat{\mu}_{g,2})(g(x_{m+i}) - \hat{\mu}_{g,2})^T. \end{aligned} \quad (10)$$

Concerning the options on the choice of the weight parameter ρ_1 , we have to take into account that its particular expression should be justified by evidence or by mathematical arguments, and moreover its value should be computable in the feature space without increasing the computational complexity. We propose three variants for the expression of the weight parameter ρ_1 estimated exclusively from data in terms of first- and second- order sample statistics, namely,

$$\rho_1^{(1)} = \frac{\|\hat{\mu}_{g,2}\|^2}{\|\hat{\mu}_{g,1}\|^2 + \|\hat{\mu}_{g,2}\|^2}, \quad (11)$$

$$\rho_1^{(2)} = \frac{\text{tr}\{\hat{\Sigma}_{g,1}\}}{\text{tr}\{\hat{\Sigma}_{g,1}\} + \text{tr}\{\hat{\Sigma}_{g,2}\}}, \quad (12)$$

$$\rho_1^{(3)} = \frac{\theta_{g,2}}{\theta_{g,1} + \theta_{g,2}}, \quad (13)$$

where

$$\theta_{g,i} = \frac{(\hat{\mu}_{g,i}^T \hat{\mu}_{g,i})^2}{\hat{\mu}_{g,i}^T \hat{\Sigma}_{g,i} \hat{\mu}_{g,i}}, \quad i = 1, 2. \quad (14)$$

The expression (11) is mostly heuristic, justified by geometric reasons, while the significance of (12) and (13) is supported by standard arguments coming from mathematical statistics (in terms of eigenvalues of sample covariance matrices and Fisher information, resp.). Note that the weight coefficients (11), (12), and (13) can be evaluated in the feature space using exclusively the values of the kernel on the available sample. Indeed, using straightforward computation, the coefficient $\rho_1^{(1)}$ can be expressed in terms of the kernel K as follows:

$$\begin{aligned} \rho_1^{(1)} &= \left(\frac{1}{(N-m)^2} \sum_{k=1}^{N-m} \sum_{l=1}^{N-m} K(x_{k+m}, x_{l+m}) \right) \\ &\times \left(\frac{1}{m^2} \sum_{k=1}^m \sum_{l=1}^m K(x_k, x_l) \right) \\ &+ \frac{1}{(N-m)^2} \sum_{k=1}^{N-m} \sum_{l=1}^{N-m} K(x_{k+m}, x_{l+m}) \Big)^{-1}, \end{aligned} \quad (15)$$

where the norms $\|\hat{\mu}_{g,1}\|^2$ and $\|\hat{\mu}_{g,2}\|^2$ are evaluated as

$$\begin{aligned}\|\hat{\mu}_{g,1}\|^2 &= \hat{\mu}_{g,1}^T \hat{\mu}_{g,1} = \frac{1}{m^2} \sum_{k=1}^m (g(x_k))^T \sum_{l=1}^m (g(x_l)) \\ &= \frac{1}{m^2} \sum_{k=1}^m \sum_{l=1}^m (g(x_k))^T (g(x_l)) \\ &= \frac{1}{m^2} \sum_{k=1}^m \sum_{l=1}^m K(x_k, x_l),\end{aligned}\quad (16)$$

$$\|\hat{\mu}_{g,2}\|^2 = \frac{1}{(N-m)^2} \sum_{k=1}^{N-m} \sum_{l=1}^{N-m} K(x_{k+m}, x_{l+m}). \quad (17)$$

Usually, the kernels are normalized, that is, $\|g(x)\|^2 = K(x, x) = 1$ for any $x \in \mathbf{R}^n$. In case of a normalized kernel, using straightforward computations, we get

$$\begin{aligned}\text{tr} \{\hat{\Sigma}_{g,1}\} &= \frac{1}{m-1} \text{tr} \left\{ \sum_{i=1}^m (g(x_i) - \hat{\mu}_{g,1}) (g(x_i) - \hat{\mu}_{g,1})^T \right\} \\ &= \frac{1}{m-1} \sum_{i=1}^m (g(x_i) - \hat{\mu}_{g,1})^T (g(x_i) - \hat{\mu}_{g,1}) \\ &= \frac{1}{m-1} \sum_{i=1}^m [(g(x_i))^T g(x_i) - 2(g(x_i))^T \hat{\mu}_{g,1} + \hat{\mu}_{g,1}^T \hat{\mu}_{g,1}] \\ &= \frac{1}{m-1} \sum_{i=1}^m \left[K(x_i, x_i) - \frac{2}{m} \sum_{k=1}^m (g(x_i))^T g(x_k) + \|\hat{\mu}_{g,1}\|^2 \right] \\ &= \frac{1}{m-1} \sum_{i=1}^m K(x_i, x_i) - \frac{2}{m(m-1)} \\ &\quad \times \sum_{i=1}^m \sum_{k=1}^m K(x_i, x_k) + \frac{m}{m-1} \|\hat{\mu}_{g,1}\|^2 \\ &= \frac{1}{m-1} \sum_{i=1}^m K(x_i, x_i) - \frac{m}{m-1} \|\hat{\mu}_{g,1}\|^2 \\ &= \frac{m}{m-1} [1 - \|\hat{\mu}_{g,1}\|^2]\end{aligned}\quad (18)$$

and similarly

$$\begin{aligned}\text{tr} \{\hat{\Sigma}_{g,2}\} &= \frac{1}{N-m-1} \sum_{i=m+1}^N K(x_i, x_i) - \frac{N-m}{N-m-1} \|\hat{\mu}_{g,2}\|^2 \\ &= \frac{N-m}{N-m-1} [1 - \|\hat{\mu}_{g,2}\|^2].\end{aligned}\quad (19)$$

Consequently, the expression of the weight coefficient $\rho_1^{(2)}$ is

$$\begin{aligned}\rho_1^{(2)} &= \left(\frac{m}{m-1} [1 - \|\hat{\mu}_{g,1}\|^2] \right) \\ &\quad \times \left(\frac{m}{m-1} [1 - \|\hat{\mu}_{g,1}\|^2] \right. \\ &\quad \left. + \frac{N-m}{N-m-1} [1 - \|\hat{\mu}_{g,2}\|^2] \right)^{-1}.\end{aligned}\quad (20)$$

The evaluation of $\rho_1^{(3)}$ can be carried out as follows. Since

$$\begin{aligned}\hat{\mu}_{g,1}^T \hat{\Sigma}_{g,1} &= \frac{1}{m(m-1)} \\ &\quad \times \sum_{k=1}^m \left(\sum_{i=1}^m \left(K(x_i, x_k) - \frac{1}{m} \sum_{p=1}^m K(x_i, x_p) \right) \right) \\ &\quad \times (g(x_k) - \hat{\mu}_{g,1})^T,\end{aligned}\quad (21)$$

by denoting

$$\begin{aligned}T(k) &= \sum_{i=1}^m \left(K(x_i, x_k) - \frac{1}{m} \sum_{p=1}^m K(x_i, x_p) \right), \\ S(k) &= \sum_{i=1}^{N-m} \left(K(x_{m+i}, x_{m+k}) - \frac{1}{N-m} \sum_{p=1}^{N-m} K(x_{m+i}, x_{m+p}) \right),\end{aligned}\quad (22)$$

we get

$$\begin{aligned}\hat{\mu}_{g,1}^T \hat{\Sigma}_{g,1} \hat{\mu}_{g,1} &= \frac{1}{m(m-1)} \sum_{k=1}^m T(k) ((g(x_k))^T \hat{\mu}_{g,1} - \|\hat{\mu}_{g,1}\|^2), \\ \hat{\mu}_{g,2}^T \hat{\Sigma}_{g,2} \hat{\mu}_{g,2} &= \frac{1}{(N-m)^2 (N-m-1)} \sum_{k=1}^{N-m} (S(k))^2,\end{aligned}\quad (23)$$

that is,

$$\begin{aligned}\rho_1^{(3)} &= \left((N-m)^2 (N-m-1) \frac{\|\hat{\mu}_{g,2}\|^4}{\sum_{k=1}^{N-m} (S(k))^2} \right) \\ &\quad \times \left(m^2 (m-1) \frac{\|\hat{\mu}_{g,1}\|^4}{\sum_{k=1}^m (T(k))^2} \right. \\ &\quad \left. + (N-m)^2 (N-m-1) \frac{\|\hat{\mu}_{g,2}\|^4}{\sum_{k=1}^{N-m} (S(k))^2} \right)^{-1}.\end{aligned}\quad (24)$$

Note that the weight coefficient $\rho_1^{(3)}$ is the extension of the Fisher coefficient to the multidimensional repartitions.

TABLE 1: The comparative analysis in terms of the number of iterations and the values of γ .

γ	Number of iterations/the estimation of the maximum value of (4)			
	SMO	The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.15	712/139.9	947/139.84	994/139.82	901/139.85
0.3	661/71.37	458/71.12	494/71.11	437/71.12
0.5	322/43.96	269/43.72	300/43.72	262/43.72
0.8	312/28.64	180/28.38	195/28.38	180/28.38
1	319/23.63	143/23.43	157/23.42	143/23.43
1.2	311/20.34	113/20.17	133/20.17	117/20.17
1.4	279/18.05	88/17.9	116/17.9	104/17.9

4. Experimental Performance Analysis of the Proposed Variants of the Weight Coefficient ρ_1

In order to develop a comparative analysis on the proposed variants of the modified gradient-like algorithm in solving the QP problem (2), we performed a long series of tests on simulated data coming from Gaussian repartitions and on the public databases WINE and Wisconsin Diagnostic Breast Cancer (WDBC) [20], all tests involving the feature extractor g corresponding to the RBF kernel $K(x, x') = \exp\{-\gamma\|x - x'\|^2\}$, $\gamma > 0$. Note that the generated data used for training were linearly separable. The comparative analysis aimed to establish conclusions about the performance of the modified gradient ascent algorithm in the feature space using the weight coefficients $\rho_1^{(i)}$, $i = 1, 2, 3$ against Platt's SMO method and the standard gradient ascent algorithm in the initial space.

Also, we aimed to evaluate

- (i) the influence of different values of the parameter γ on the number of iterations needed to obtain significant accuracy,
- (ii) the dependency of the number of iterations required to obtain significant accuracy on the distance between the classes the samples come from and on the samples variability,
- (iii) the influence of different values of the parameter γ on class separability index.

The variability of a sample coming from a certain class can be expressed by many ways. We considered the indicator given by the mean distance between the feature vectors representing examples coming from that class to quantitatively express the variability within the sample. If C is the subset of S_N containing the examples coming from one of the labeled classes, then the measure of the variability of C is

$$\text{var}(C) = \frac{1}{2|C|^2} \sum_{x_i, x_j \in C} \|g(x_i) - g(x_j)\| \quad (25)$$

that can be expressed in terms of the kernel function as [18]

$$\text{var}(C) = \frac{1}{\sqrt{2}|C|^2} \sum_{x_i, x_j \in C} \sqrt{1 - K(x_i, x_j)}. \quad (26)$$

The class separability index is evaluated by

$$s(h_1, h_2) = \min_{\substack{x_i \in h_1 \\ x_j \in h_2}} \|g(x_i) - g(x_j)\|^2 = \min_{\substack{x_i \in h_1 \\ x_j \in h_2}} K^2(x_i, x_j). \quad (27)$$

Test 1. We aimed to derive conclusions of previously mentioned types in case of simulated data from normal multidimensional classes $N(\mu_i, \Sigma_i)$, $i = 1, 2$. The closeness degree between the resulted datasets $S^{(1)}$, $S^{(2)}$ of sizes N_1 , N_2 can be evaluated by many ways. One way was to express it in terms of the Mahalanobis distance, $d(h_1, h_2) = (\mu_1 - \mu_2)^T (\Sigma_1 + \Sigma_2)^{-1} (\mu_1 - \mu_2)$. We consider two model-free indices to express the closeness degree using only the datasets, given by the sample Mahalanobis distance, $\tilde{d}_{N_1, N_2}(h_1, h_2) = (\hat{\mu}_1 - \hat{\mu}_2)^T (\hat{\Sigma}_1 + \hat{\Sigma}_2)^{-1} (\hat{\mu}_1 - \hat{\mu}_2)$ and $\tilde{d}_{N_1, N_2}(h_1, h_2) = \min_{x \in S^{(1)}, y \in S^{(2)}} \|x - y\|$, respectively, where $\hat{\mu}_1$, $\hat{\mu}_2$, $\hat{\Sigma}_1$, $\hat{\Sigma}_2$ are the sample means and sample covariance matrices corresponding to $S^{(1)}$, $S^{(2)}$.

For instance, the conclusions derived on the basis of the samples $S^{(1)}$, $S^{(2)}$ generated from Gaussian repartitions, where $N_1 = 45$, $N_2 = 45$, $d(h_1, h_2) = 6.6301$, $\tilde{d}_{N_1, N_2}(h_1, h_2) = 9.8424$, and $\tilde{d}_{N_1, N_2}(h_1, h_2) = 0.0776$, are summarized in Figures 1–3 and Table 1.

The samples prove to be comparable values of the variability index, for each $\gamma \in (0, 1]$, and the variability indices depend increasingly on γ (Figure 1).

The results obtained in solving the QP problem (2) using Platt's SMO algorithm and the variants of the gradient algorithm using the weight parameters given by (15), (20), and (24), respectively, are summarized in Table 1 and shown in Figure 2. Each entry a/b of Table 1 represents a = number of iterations and b = the resulted estimation of the maximum value of (4) corresponding to a particular value of γ and resulted by applying one of the proposed variants of algorithms.

According to the experimental results, we can conclude that in order to obtain the same accuracy, the variants of

FIGURE 1: The dependency on γ of the variability index (26).

FIGURE 2: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms.

the gradient ascent algorithm using the weight parameter ρ_1 given by (15), (20), and (24) require a far less number of iterations. Moreover, as γ increases, the number of iterations dramatically decreases as compared to Platt's SMO algorithm.

In order to evaluate the recognition rate of the resulted SVM classifier, we used linearly and nonlinearly datasets coming from the same distributions. In the case of this example, the mean recognition rate was around 94%.

Similar results were obtained in case "closer" or "farther" normal distributions were used to generate datasets.

Concerning the variants (15), (20), and (24) of weighting coefficient ρ_1 , the tests proved almost the same efficiency,

FIGURE 3: The dependency on γ of the class separability index (27).

their mean values being around 0.62, 0.54, and 0.64, respectively. It seems that the variant (24) behaves better than (15) and (20) for $\gamma < 1$.

Test 2. We aim to develop a comparative analysis among the performance corresponding to Platt's SMO and the variants of gradient ascent algorithms presented in Section 2 on the WINE dataset [20]. The data in WINE dataset are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars, the analysis being determined by the quantities of 13 constituents found in each of the three types of wines. The dataset consists of labeled examples of sizes 59, 71, and 48 coming from the pairwise linearly separable classes $S^{(1)}$, $S^{(2)}$, $S^{(3)}$, respectively. In processing the WINE dataset, we used the kernel $K(x, x') = \exp\{-\gamma\|x - x'\|\}$, for different values of $\gamma > 0$.

In order to implement the previously mentioned algorithms, we used the variability and separability indices (26) and (27) to find out a suitable criterion for planning a two-class classification of these three subsamples. Unfortunately, the three subsamples proved very close values of the interclass separability index (27), for all values of γ ; therefore, from this point of view, all two-class classifications seemed to be almost equivalent. The variation with respect to γ of the interclass separability index (27) is presented in Figure 4.

Hopefully, the three subsamples proved quite different variability in the sense of the variability index (26), for all values of γ , enabling us to formulate the plan: discriminate first between $S^{(3)}$ and $S^{(1)} \cup S^{(2)}$ and then discriminate between $S^{(1)}$ and $S^{(2)}$.

The results obtained in solving the QP problem (2) using Platt's SMO algorithm, and the variants of the gradient algorithm using the weight parameters given by (15), (20), and (24), respectively, are summarized in Tables 2 and 3 and shown in Figures 5 and 6. Each entry a/b of the tables represents a = number of iterations and b = the resulted estimation of the maximum value of (4) corresponding to a particular value of γ and resulted by applying one of the proposed variants of algorithms.

TABLE 2: The comparative analysis in terms of the number of iterations and the values of γ in discriminating between $S^{(3)}$ and $S^{(1)} \cup S^{(2)}$.

γ	Number of iterations/the estimation of the maximum value of (4)			
	SMO	The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.15	1768/61.64	214/61.18	249/61.50	193/61.58
0.3	806/64.41	149/63.10	144/63.12	130/63.27
0.5	357/64.45	157/65.59	157/65.59	143/65.39
0.75	355/65.74	159/66.84	159/66.84	165/66.83
1	354/66.42	159/67.26	160/67.25	172/66.90
1.2	354/66.57	160/67.41	160/67.41	173/66.51

TABLE 3: The comparative analysis in terms of the number of iterations and the values of γ in discriminating between $S^{(1)}$ and $S^{(2)}$.

γ	Number of iterations/the estimation of the maximum value of (4)			
	SMO	The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.15	202/44.20	73/44.05	72/44.08	73/44.10
0.3	434/56.65	90/56.35	88/56.29	84/56.07
0.5	357/60.37	96/60.91	96/60.88	102/60.96
0.75	330/60.96	96/62.31	96/62.30	104/61.96
1	382/62.83	96/62.71	96/62.70	96/62.36
1.2	382/62.90	96/63.04	96/63.04	97/62.98

FIGURE 4: The dependency on γ of the interclass separability index (27).

In case of this example, the recognition rate was 100% for all values of γ . Being given the missing information about the measured features and the sizes relative small sizes of the subsamples coming from the three categories of wines, we had no possibility to test the generalization capacities of the resulted classifiers either on simulated data or by splitting the subsamples in design and test data, respectively.

Test 3. We performed a series of tests on the dataset Wisconsin Diagnostic Breast Cancer (WDBC) [20] aiming to develop a comparative analysis on the performance of the variants of gradient ascent algorithms presented in Section 2 and Platt's SMO. We used the kernel $K(x, x') = \exp\{-\gamma\|x - x'\|\}$, for different values of $\gamma > 0$, and the weighting coefficients given by (15), (20), and (24).

The examples in WDBC dataset 30-dimensional vectors representing the features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass describing characteristics of the cell nuclei present in the image, a confirmed diagnosis either benign (B) or malignant (M) being supplied. It is stated that the dataset is linearly separable and relevant for the design of classifiers having good generalization capacities. The sizes of subsamples labeled by Band M are 357 and 212, respectively.

The variation with respect to γ of the values of the variability and separability indices (26) and (27) is presented in Figures 7 and 8. The separability index (27) proves to be quite sensitive to the variation of the values of the parameter γ pointing out an increasing dependency on γ . Moreover, for $\gamma \geq 0.5$, the values of the separability index (27) stabilize around 1.41. Using the variability index (26), the tests revealed that the variability of the M-class seems to be insensitive at the variations of γ , while the variability index of the B-class depends increasingly on γ , the values stabilizing around the value 0.705 for $\gamma \geq 0.5$.

In the first series of tests, all examples were used in both design and test phases. The results are summarized in Table 4 and Figure 9. Concerning the use of the weight coefficients

FIGURE 5: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms in discriminating between $S^{(3)}$ and $S^{(1)} \cup S^{(2)}$.

FIGURE 6: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms in discriminating between $S^{(1)}$ and $S^{(2)}$.

(15), (20), and (24), the accurate estimates of the maximum value of (4) were computed in a far less number of iterations than Platt's SMO algorithm. A slight better performance resulted for all values of γ in case of using (24), the mean value of $\rho_1^{(3)}$ being around 0.99, while the mean values of $\rho_1^{(1)}$ and $\rho_1^{(2)}$ were around 0.5 and 0.4, respectively.

FIGURE 7: The dependency on γ of the variability index (26).

FIGURE 8: The dependency on γ of the separability index (27).

From the point of view of the performance in discriminating between the B-class and M-class, for all values of γ , the recognition rate was 100%.

Being given that the size of the WDBC dataset is relatively large, we used it in order to develop a comparative analysis from the point of view of the generalization capacity corresponding to the proposed variants. In order to derive a suitable splitting strategy of the available data into design and test datasets, we took into consideration the relative relevance of the examples with respect to the class from which they come. We computed a prototype (barycenter) for each class by averaging the examples belonging to it, and the relative relevance of each example coming from each class is expressed in terms of the Euclidian distance to the corresponding prototype.

In order to establish suitable partitions into design and test subsets for each class, we used several strategies, the differences between them being given by the sizes and the

TABLE 4: The comparative analysis in terms of the number of iterations and the values of γ .

γ	SMO	Number of iterations/the estimation of the maximum value of (4)		
		The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.15	2710/214.95	537/214.99	501/214.57	523/215.88
0.2	2730/238.19	500/237.20	513/239.01	493/237.39
0.3	2969/255.09	497/255.91	493/255.83	492/254.93
0.5	2958/261.88	501/260.26	501/260.20	493/260.85
0.75	2936/262.10	503/261.94	503/261.13	492/261.67
1	2712/260.64	504/261.26	503/261.27	492/261.79
1.2	2865/260.57	503/261.29	502/261.29	492/261.81

FIGURE 9: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms. FIGURE 10: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms.

way in which the available examples were allotted to the design and test samples. On one hand, in order to extract the most information concerning the classes, the design sample should include some of the most representative examples. On the other hand, the less representative examples give us information concerning the class variability. Following this idea, we arrived at the conclusion that the design and test samples should contain both kinds of examples in different proportions. We aimed to derive conclusions on experimental basis concerning the effects of design and test samples on the recognition rate and generalization capacity. In a series of tests, we considered the following experimental plan.

- (1) Compute the barycenter of each class (given by the mean of the examples belonging to the class), and sort the examples in each class according to the increasing order of the values of the Euclidian distance to the barycenter.

- (2) Select the first 10 examples from each class and include them in the test sample. Include the 10 examples from each class corresponding to the largest distances to the barycenter in the design sample. The rest of the examples are placed alternatively in the design and test datasets.

By applying this plan, the whole dataset was split into design dataset consisting of 179 and 106 examples and test dataset including 178 and 106 examples coming from the B-class and the M-class, respectively.

The results are summarized in Table 5 and Figure 10. Concerning the usefulness of the proposed weight coefficients, the variants (15), (20), and (24) proved almost equal efficiency, the number of iterations required to obtain an accurate estimate of the maximum value of (4) being far less than in case of Platt's SMO algorithm. By submitting the test sample to the resulted classifiers, we obtained correct recognition rates in the range [94.01%, 95.07%], the maximum value 95.07%

TABLE 5: The comparative analysis in terms of the number of iterations and the values of γ .

γ	SMO	Number of iterations/the estimation of the maximum value of (4)		
		The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.1	1351/106.32	294/106.60	230/106.06	272/106.78
0.15	1495/120.49	314/120.54	251/120.49	254/120.45
0.2	1485/126.39	324/127.19	250/127.22	298/127.04
0.3	1497/130.52	322/131.84	252/131.82	288/131.01
0.5	1461/131.77	252/131.46	252/131.43	285/132.31

TABLE 6: The comparative analysis in terms of the number of iterations and the values of γ .

γ	SMO	Number of iterations/the estimation of the maximum value of (4)		
		The variant of the gradient algorithm using the weight parameter (15)	The variant of the gradient algorithm using the weight parameter (20)	The variant of the gradient algorithm using the weight parameter (24)
0.1	2120/114.85	312/114.44	300/114.57	330/114.69
0.15	2320/129.18	278/129.17	273/129.23	280/129.19
0.2	2223/135.88	278/136.44	278/136.57	278/136.01
0.3	2285/141.10	277/141.52	276/141.50	323/140.72
0.5	3073/142.89	301/142.99	297/142.98	379/142.84

FIGURE 11: The dependency on γ of the number of iterations required by Platt's SMO and the variants of the gradient ascent algorithms.

being obtained for $\gamma \in [0.1, 0.15]$ and the weight coefficient (20).

Several tests were performed using the same strategy for different sizes of the design and test datasets. For instance, in Table 6 and Figure 11, the results of a test that used the design and test datasets obtained by including from each class the most relevant 10 examples in the test set and the least relevant 30 examples in the design sample are represented. This way, the learning phase was developed on a design dataset containing 189 and 116 examples coming from the B-class and the M-class, respectively. The test phase was performed on a dataset

containing 168 and 96 examples coming from the B-class and the M-class, respectively, the resulted recognition rates being in the range [95.08%, 96.59%], where the maximum value 96.59% was obtained for $\gamma = 0.1$.

As it is expected, higher recognition rates are obtained in case the design dataset is enlarged to contain more of the less relevant examples. For instance,

- in case the design dataset consists of the least relevant 60 examples and 40 examples from B-class and M-class, respectively, the rest of examples being allotted alternatively to the design and test samples; the sizes of the test and design datasets are 41.12% and 58.88%; the resulted recognition rate is 96.15% when $\gamma = 0.15$;
- the test based on the design dataset consisting of the least relevant 75 examples and 60 examples from B-class and M-class, respectively, the rest of examples being allotted alternatively to design and test samples (38.14% examples in the test set); the recognition rate increases to 98.18% when $\gamma = 0.15$;
- finally, 100% recognition rate results using $\gamma = 0.15$ and the design dataset consisting of the least relevant 100 examples and 70 examples from B-class and M-class, respectively (33.56% examples in the test set).

5. Conclusions and Suggestion for Further Work

In the paper, we propose a modified gradient ascent method for solving the dual problem of nonlinear SVM. Basically, the refinement proposed here consists in using weight parameters to tune the direction of the search to the particular training sequence. The work was based on the use of variability and separability indices expressed in terms of the

exponential RBF kernel. A part of the comparative analysis aimed to evaluate the dependency of the expected number of iterations required to obtain reasonable accuracy of the criterion function on the kernel parameter.

The proposed variants of the gradient ascent learning algorithm are somehow heuristically justified in the sense that there is no mathematically founded proof of the convergence properties. Therefore, several tests were performed in order to derive conclusions on experimental basis. The tests pointed out good convergence properties of the modified variants, and their convergence rates were significantly higher as compared to Platt's SMO algorithm. The experimental analysis aimed to derive conclusions on the recognition rate as well as on the generalization capacities. All linear classifiers proved almost equal recognition rate and generalization capacities, the difference being given by the number of iterations required for learning the separating hyperplanes.

The learning phase of the SVM involved linearly separable samples randomly generated from Gaussian repartitions and the WINE and WDBC datasets. In order to evaluate the generalization capacities, several tests were also performed on new linearly/nonlinearly separable data coming from the same classes in case of samples randomly generated from Gaussian classes. In case of the WINE and WDBC datasets, both of them are linearly separable and no information concerning the generative model is supplied; therefore an additional strategy for splitting them into design and test samples was required. In case of the tests performed on the WINE dataset, being given its relatively small size, the performance was analyzed using all samples in the design and test phases, while the size of the WDBC dataset, being significantly larger, allowed us to develop different experimental plans by splitting the available data into design and test samples of different sizes.

Being given the optimality of SVMs from the point of view of generalization capacities, as expected, we obtained high recognition rates on new test data in most of cases (around 97%). In case of the WDBC dataset, higher recognition rates were obtained in case the design dataset was enlarged to contain more of the less relevant examples. For instance, a 100% recognition rate resulted using $\gamma = 0.15$ and 66.44% examples in the design set.

The tests pointed out that the variation of the recognition rates depends also on the inner structure of the classes from which the learning data come as well as on interclass separability degree. Consequently, we estimate that the results are encouraging and entail future work toward extending these refinements to multiclass classification problems and approaches in a fuzzy-based framework.

References

- [1] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.
- [2] V. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.
- [3] S. Abe, "Support vector machines for pattern classification," in *Advances in Pattern Recognition*, Springer, London, UK, 2010.
- [4] J. Shawe-Taylor and N. Cristianini, *Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, UK, 2000.
- [5] E. Osuna, R. Freund, and F. Girosi, "Improved training algorithm for support vector machines," in *Proceedings of the 7th IEEE Workshop on Neural Networks for Signal Processing (NNSP '97)*, pp. 276–285, September 1997.
- [6] L. I.-J. Chien, C.-C. Chang, and Y.-J. Lee, "Variant methods of reduced set selection for reduced support vector machines," *Journal of Information Science and Engineering*, vol. 26, no. 1, pp. 183–196, 2010.
- [7] Y.-J. Lee and O. L. Mangasarian, "SSVM: a smooth support vector machine for classification," *Computational Optimization and Applications*, vol. 20, no. 1, pp. 5–22, 2001.
- [8] L. J. Cao, S. S. Keerthi, C. J. Ong, P. Uvaraj, X. J. Fu, and H. P. Lee, "Developing parallel sequential minimal optimization for fast training support vector machine," *Neurocomputing*, vol. 70, no. 1–3, pp. 93–104, 2006.
- [9] G. C. Cawley and N. L. C. Talbot, "Improved sparse least-squares support vector machines," *Neurocomputing*, vol. 48, pp. 1025–1031, 2002.
- [10] C.-H. Li, H.-H. Ho, Y.-L. Liu, C.-T. Lin, B.-C. Kuo, and J.-S. Taur, "An automatic method for selecting the parameter of the normalized kernel function to support vector machines," *Journal of Information Science and Engineering*, vol. 28, no. 1, pp. 1–15, 2012.
- [11] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods—Support Vector Learning*, pp. 41–56, 1998.
- [12] J. A. K. Suykens, J. de Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: robustness and sparse approximation," *Neurocomputing*, vol. 48, pp. 85–105, 2002.
- [13] S. Rueping, "mySVM: another one of those support vector machines," 2003, <http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM>.
- [14] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, Cambridge, Mass, USA, 2004.
- [15] P. Laskov, "Feasible direction decomposition algorithms for training support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 315–349, 2002.
- [16] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: primal estimated sub-Gradient Solver for SVM," in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, pp. 807–814, June 2007.
- [17] V. Yugov and I. Kumazava, "Online boosting algorithm based on two-phase SVM training," *ISRN Signal Processing*, vol. 12, 2012.
- [18] L. State, C. Cocianu, and M. Mircea, "Heuristic attempts to improve the generalization capacities in learning SVMs," in *Proceedings of the 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 108–116, 2012.
- [19] C.-L. Cocianu, L. State, and P. Vlamos, "A new method for learning the support vector machines," in *Proceedings of the 6th International Conference on Software and Database Technologies (ICSOT '11)*, pp. 365–370, July 2011.
- [20] <http://archive.ics.uci.edu/ml/index.html>.