SEQUENTIAL MAXIMUM GRADIENT OPTIMIZATION FOR SUPPORT VECTOR DETECTION

Mireille Tohmé[1,2], *Régis Lengellé*[2]

[1] FORENAP Frp, 27, Rue du 4eme RSM , 68250 Rouffach, France
[2] ICD-LM2S (FRE CNRS 2848), Université de Technologie de Troyes, BP 2060, 10010 Troyes cedex, France
mireille.tohme@utt.fr ; regis.lengelle@utt.fr

ABSTRACT

Support Vector Machines (SVM) are playing an increasing role for detection problems in various engineering domains, notably in statistical signal processing, pattern recognition, image analysis, and communication systems. In this paper, we present a new method for optimizing Support Vector Machines for classification problems. An implicit reformulation of the optimization problem is proposed. The bias term is added to the primal problem formulation, which leads to eliminating the equality constraint. In order to deal with large data set problems, we propose a decomposition method, Sequential Maximum Gradient Optimization (SMGO), that relies on the selection of the working set via the search of the highest absolute values of the gradient. Furthermore, considering the quadratic nature of the dual problem, the optimum step-size is analytically determined. Moreover the solution, the gradient and the objective function are recursively calculated. The Gram matrix has not to be stored. SMGO is easy to implement and able to perform on large data sets.

1. INTRODUCTION

The last few years have seen the rise of Support Vector Machines (SVM) as powerful tools in machine learning for classification and regression problems [3] and they have much success in detection problems [14] [6]. The complexity of SVM only depends on the number of observations which is equal to the sample size, thus the need for developing efficient learning algorithms to solve the optimization problem for large data sets.
Recently, fast iterative algorithms have been suggested. Among them, decomposition methods, initially proposed by Joachims [8], put a subset of data into a working set and solves the Quadratic Programming (QP) problem by optimizing the corresponding Lagrange multipliers while keeping the other unchanged. In this way, a large QP problem is decomposed into a series of smaller QP problems, thus making it possible to train a SVM on large data sets. Using an idea similar to decomposition, Chunking [23], solves a QP problem on a subset of data. Chunking trains a SVM using a set of training data and only keeps the support vectors into the subset. These steps are repeated until all training data are used and they all satisfy the Karush-Kuhn-Tucker (KKT) conditions. Collobert et al [4] presented SVM-Torch, a decomposition algorithm similar to the one proposed by Joachims [8]. Among all decomposition algorithms, Sequential Minimum Optimization (SMO) introduced by Platt [15] takes decomposition methods to the extreme: the working set is only composed of two observations for which the optimization process is performed analytically at every iteration. Mangasarian and his colleagues [11] [9] proposed several variations of standard support vector machines by modifying the objective function, together with several very efficient training algorithms. For more detailed survey, see [2]. Another way to deal with large scale data sets is to use the geometric properties of SVM. As in [16], they developed learning algorithms based on reduced convex hull. Their algorithms have been shown to be efficient and accurate.
In this paper, we present a new learning algorithm in the framework of the decomposition methods. We propose a reorganization of Platt's algorithm SMO, using Joachims heuristic. Inspired by Mangasarian et al [7] [10], we add the bias term in the primal optimization problem. Therefore, we combine all these ideas and we extend our previous work [12] to propose a new learning algorithm for solving detection problems: Sequential Maximum Gradient Optimization (SMGO). In SMGO, the bias term added to the objective function turns out to eliminate the equality constraint, consequently the number of Lagrange multipliers to be modified at each iteration is no more lower bounded by 2. The working set is selected following an efficient heuristic that relies on the search for the maximum absolute values of the gradient ensuring feasible directions. The optimal step-size of the gradient algorithm is calculated analytically The solution, the gradient and the objective function are recursively computed. All these steps, coupled with the previous heuristic, ensure a fast and efficient training algorithm and are new within the context of SVM training.
We begin this paper with a brief review of the basic technique used for implementing Support Vector Machines for detection problems. Then we give an overview of SMGO by introducing in details its different steps. Section 4 presents the experimental results and we finally conclude with some perspectives.

2. SUPPORT VECTOR DETECTION

The goal of this section is to give a brief review of the fundamentals of Support Vector in the non linear case. Given a training data set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$, $\mathbf{x}_i \in \mathscr{X} \subset \mathbb{R}^p, y_i \in \{-1, +1\}$. The problem is to find a hyperplane in $\Phi(\mathscr{X})$ defined by:

$$\left\langle \mathbf{w}, \Phi(\mathbf{x}) \right\rangle + b = 0 \tag{1}$$

Assuming that, without loss of generality, $\Phi(\mathbf{x})$ belongs to a finite dimensional space, so the dot product $\left\langle \mathbf{w}, \Phi(\mathbf{x}) \right\rangle$ can be written $\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{x})$, and assuming that the data are linearly separable in $\Phi(\mathscr{X})$, this hyperplane must verify

$$\mathrm{sign}(\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{x}_i) + b) = y_i, \forall i$$

In the case of SVM, the separating hyperplane is defined by:

$$\mathbf{w}, b : \min_i \left| \mathbf{w}^T \Phi(\mathbf{x}_i) + b \right| = 1 \tag{2}$$

It is easy to prove that the distance from the closest training point $\Phi(\mathbf{x}_i)$ to the hyperplane defined by (2) is $1/\parallel \mathbf{w} \parallel$. So we aim to maximize the margin given by $1/\parallel \mathbf{w} \parallel$. Allowing some relaxation to deal with overlapping data distributions, the optimization problem is then:

$$\begin{array}{l} \underset{\mathbf{w},\xi}{\text{minimize}} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \left\{ \begin{array}{l} y_i\left(\mathbf{w}^T\Phi(\mathbf{x}_i)+b\right) \geq 1-\xi_i \\ \xi_i \geq 0, \quad i=1,....,N \end{array} \right. \end{array} \tag{3}$$

Here, $\mathbf{w}$ is a vector in $\Phi(\mathscr{X})$, and $\Phi(\mathbf{x})$ maps the input $\mathbf{x}$ to a vector in $\Phi(\mathscr{X})$. $C > 0$ is a regularization constant that controls the trade-off between the empirical loss and the margin width, the slack variables ξ_i represent the empirical loss associated with $\mathbf{x}_i$ and b is calculated using the Karush-Kuhn-Tucker (KKT) conditions.
The hyperplane given by (1) can be written as:

$$\begin{pmatrix} \mathbf{w} \\ b \end{pmatrix}^T \begin{pmatrix} \Phi(\mathbf{x}) \\ 1 \end{pmatrix} = 0 \tag{4}$$

The hyperplanes defined by (1) and (4) are strictly equivalent. Consequently, within the context of SVM, and as in [7] [10], we introduce the term $b^2/2$ in the objective function, so the primal SVM problem (3) becomes:

$$\begin{array}{l} \underset{\mathbf{w},\xi,b}{\text{minimize}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{b^2}{2} + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \left\{ \begin{array}{l} y_i\left(\mathbf{w}^T\Phi(\mathbf{x}_i)+b\right) \geq 1-\xi_i \\ \xi_i \geq 0, \quad i=1,....,N \end{array} \right. \end{array} \tag{5}$$

By introducing the Lagrange multipliers and minimizing the Lagrangian with respect to the primal variables, the problem (5) can be written as follows:

$$\begin{array}{ll} \max\limits_{\alpha} & W(\alpha) = -\frac{1}{2}\sum\limits_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i,\mathbf{x}_j) + \sum\limits_{i=1}^{N}\alpha_i \\ & \quad -\frac{1}{2}\sum\limits_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j \\ \text{subject to} & \alpha_i \in [0,C] \end{array} \tag{6}$$

where $k(\mathbf{x}_i,\mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$. The introduction of $b^2/2$ turns out that:

- the equality constraint in the dual problem, i.e., $\sum_i \alpha_i y_i = 0$, gets eliminated
- the bias term is given by: $b = \sum_i \alpha_i y_i$

In matrix form, this problem can be written:

$$\begin{array}{ll} \max\limits_{\alpha} & W(\alpha) = \frac{1}{2}\alpha^{\mathbf{T}}\mathbf{H}\alpha + \alpha^T\mathbf{1}_N \\ \text{subject to} & \alpha \in [0,C]^N \end{array} \tag{7}$$

where $\mathrm{H}_{ij} = -y_i y_j (k(\mathbf{x}_i,\mathbf{x}_j)+1)$ and $\mathbf{1}_N$ is the N dimensional vector composed of 1.

3. ALGORITHM DESCRIPTION

In this section, we give an overview of our algorithm. Our main idea to iteratively solve problem (7) is to look for a working set composed of the q highest absolute values of the gradient that also define feasible directions. Once the working set is defined, we proceed on a optimal step gradient algorithm. Considering the quadratic nature of the optimization problem, the optimal step-size is analytically calculated. It is then clipped so that the updated solution α satisfies the inequality constraint.

3.1 Selecting a possible working set

All the previous work [8] [15] have searched for a direction $\mathbf{u}$ in such a way that $\mathbf{u}^{\mathbf{T}}\mathbf{g}$ is maximum, where $\mathbf{g}$ is the gradient of the objective function. In the case of a working set of size 2, $\mathbf{u}$ has only two non-null components equal to ± 1. However, in our case we do not have to satisfy any equality constraint as the result of our formulation of the quadratic optimization problem (7), so we are not restricted to the choice of $\mathbf{u}$ defined previously. Consequently, the direction move could be collinear to the *partial gradient* i.e. the gradient calculated only with respect to the working set. Accordingly, we define the working set by a set of indices I_{WS} given by:

$$I_{WS} = \left\{ i : |g_i| \in \left\{ \begin{array}{l} q \text{ largest absolute values} \\ \text{of } g_n, n = 1,...N \end{array} \right\} \right\} \tag{8}$$

But an update of α in the direction of the *partial gradient* can be allowed if and only if the *partial gradient* defines a feasible direction. This will be ensured if it is not located on the boundaries of the domain $[0,C]^N$ when the *partial gradient* leads us outside the domain. Consequently, the search for the working set is now defined by:

$$I_{WS} = \left\{ i : |g_i| \in \left\{ \begin{array}{l} q \text{ largest absolute values} \\ \text{of } g_n, n = 1,...N; \text{ with} \\ \alpha_i < C \text{ if } g_i > 0 \quad \text{and} \\ \alpha_i > 0 \text{ if } g_i < 0 \end{array} \right\} \right\} \tag{9}$$

The *partial gradient* $\tilde{\mathbf{g}}$ that will be our direction move is defined by:

$$\tilde{g}_i = \left\{ \begin{array}{ll} g_i & \text{if } i \in I_{WS} \\ 0 & \text{elsewhere} \end{array} \right. \tag{10}$$

In some cases, we cannot obtain a working set of size q since the number of indices verifying (9) is smaller than q. The size of the working set, in this case, is reduced to the number of indices verifying (9).

3.2 Calculation of the optimal step-size

The update of the solution is:

$$\alpha \leftarrow \alpha + \lambda_{opt}\tilde{\mathbf{g}} \tag{11}$$

where λ_{opt} is the optimum step-size that verifies:

$$\lambda_{opt} = \arg\max_{\lambda} \left(W(\alpha + \lambda\tilde{\mathbf{g}})\right) \tag{12}$$

In the quadratic case, it is easy to prove that λ_{opt} is given by:

$$\lambda_{opt} = \frac{\tilde{\mathbf{g}}^{\mathbf{T}}\tilde{\mathbf{g}}}{\tilde{\mathbf{g}}^{\mathbf{T}}\mathbf{H}\tilde{\mathbf{g}}} \tag{13}$$

Note that, to calculate λ_{opt}, we do not need to compute the whole matrix $\mathbf{H}$. We only need to determine the submatrix of $\mathbf{H}$ of size $(q \times q)$ associated to the elements of the working set. Moreover, $\mathbf{H}$ has not to be stored.
Before any modification, we need to verify that the updated point remains feasible (it must satisfy all the constraints):

$$\begin{aligned} & 0 \leq \alpha_i + \lambda \tilde{g}_i \leq C, i \in I_{WS} \\ & \text{if } \tilde{g}_i > 0 \quad \tfrac{-\alpha_i}{\tilde{g}_i} \leq \lambda \leq \tfrac{C-\alpha_i}{\tilde{g}_i} \\ & \text{if } \tilde{g}_i < 0 \quad \tfrac{C-\alpha_i}{\tilde{g}_i} \leq \lambda \leq \tfrac{\alpha_i}{\tilde{g}_i} \end{aligned} \tag{14}$$

from which we can deduce:

$$\lambda \leq \min\left(\min_i(\frac{C-\alpha_i}{\tilde{g}_i}), \min_i(\frac{\alpha_i}{\tilde{g}_i})\right) \tag{15}$$

$$\lambda \geq \max\left(\max_i(\frac{C-\alpha_i}{\tilde{g}_i}), \max_i(\frac{-\alpha_i}{\tilde{g}_i}), 0\right) \tag{16}$$

If λ does not verify equation (15) then

$$\begin{aligned} & \lambda > \min\left(\min_i(\frac{C-\alpha_i}{\tilde{g}_i}), \min_i(\frac{\alpha_i}{\tilde{g}_i})\right) \Rightarrow \\ & \lambda_{opt} \longleftarrow \min\left(\min_i(\frac{C-\alpha_i}{\tilde{g}_i}), \min_i(\frac{\alpha_i}{\tilde{g}_i})\right) \end{aligned} \tag{17}$$

likewise if λ does not verify (16) then

$$\begin{aligned} & \lambda < \max\left(\max_i(\frac{C-\alpha_i}{\tilde{g}_i}), \max_i(\frac{-\alpha_i}{\tilde{g}_i}), 0\right) \Rightarrow \\ & \lambda_{opt} \longleftarrow \max\left(\max_i(\frac{C-\alpha_i}{\tilde{g}_i}), \max_i(\frac{-\alpha_i}{\tilde{g}_i}), 0\right) \end{aligned} \tag{18}$$

3.3 Calculation of the increments of the objective function and gradient

The increment of the objective function can be calculated by:

$$\Delta W = W(\alpha + \lambda_{opt}\tilde{\mathbf{g}}) - W(\alpha)$$

In matrix form ΔW can be written as:

$$\Delta W = \lambda_{opt}\tilde{\mathbf{g}}^T\mathbf{H}\alpha + \frac{1}{2}\lambda_{opt}^2\tilde{\mathbf{g}}^T\mathbf{H}\tilde{\mathbf{g}} + \lambda_{opt}\tilde{\mathbf{g}}^{\mathbf{T}}\mathbf{1_N} \tag{19}$$

Taking into account that $\tilde{\mathbf{g}}$ has $N-q$ null components, we do not have to calculate the whole matrix $\mathbf{H}$ to evaluate ΔW.
The gradient of the objective function W is given by:

$$\mathbf{g} = \mathbf{H}\alpha + \mathbf{1_N}$$

An update of the gradient is also necessary:

$$\Delta\mathbf{g} = \lambda_{opt}\mathbf{H}\tilde{\mathbf{g}} \tag{20}$$

Here again, the update of the gradient does not exhibit the calculation of $\mathbf{H}$, but only the $q \ll N$ rows corresponding to the working set.

3.4 Initialization

The initial values of the Lagrange multipliers are, as usual, set to 0 (feasible solution) so we have:

$$\alpha = \mathbf{0} \Rightarrow \mathbf{g} = \mathbf{1}_N,\ W(\mathbf{0}) = 0 \tag{21}$$

3.5 Algorithm

The SMGO algorithm can be resumed as follows:

1. Initialization, equation (21)
2. Direction search, $\tilde{\mathbf{g}}$, equations (9), (10)
3. Calculation of the optimal step-size λ_{opt}, equations (13), (15), (16) (17) and (18)
4. Update of the solution, $\alpha \longleftarrow \alpha + \lambda_{opt}\tilde{\mathbf{g}}$
5. Update of the gradient $\mathbf{g} \longleftarrow \mathbf{g} + \mathbf{\Delta g}$, where $\Delta\mathbf{g}$ is given by (20)
6. Update of the objective function $W \longleftarrow W + \Delta W$ where ΔW is given by (19)
7. Test for convergence, if not return to step 2

Because the constraints are satisfied by construction, our stopping criterion is based on the increment of the objective function. It has been shown satisfactory.
It is easy to show that the bias term is calculated from the annulation of the derivative of the primal objective function with respect to b:

$$b = \sum_i \alpha_i y_i \tag{22}$$

4. EXPERIMENTAL RESULTS

We compare now our SMGO algorithm with an implementation of SVM-Light. In a first step, we are only concerned with training time comparison, so we do not use any test set. However, in a second step, test data are used to compare the performance of SMGO with SVM-Light, here used as a reference. We consider different sizes of the working set. Both algorithms are written in MATLAB and the kernel used is gaussian $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$. The data sets used are real word applications[1] and simulated data. The parameters used σ and C are the ones used in the literature.
The first data set is the Spam emails aiming to classify emails between spam or non-spam. The size $N = 4601$ and 57 attributes are used, $\sigma = 5$ and $C = 10$.

Working Set size q	SMGO	SVM-Light
1	198.73	-
2	143.56	200.66
3	99.94	199.016
4	86.38	152.79
6	167.75	141.83

Table 1: Training time (s) for Spam emails database

Working Set size q	SMGO	SVM-Light
1	575.21	-
2	837.40	1633.96
3	1143.70	1636.54
4	718.51	2282.71
6	535.78	2906.02

Table 2: Training time (s) for gaussian simulated data set

The second data set used is artificially generated and consists of two 10-dimensional gaussian distributions with

[1] archive.ics.uci.edu/ml/datasets

Working Set size q	SMGO	SVM-Light
1	171.54	-
2	300.75	850.60
3	350.64	825.88
4	402.16	1131.67
6	350.65	1287.34

Table 3: Training time (s) for Forest

Working Set size q	SMGO	SVM-Light
1	1.715	-
2	1.874	7.102
3	3.448	6.362
4	3.567	3.519
6	2.782	2.811

Table 4: Training time (s) for Diabetes database

Working Set size q	SMGO	SVM-Light
1	0.621	-
2	0.478	1.029
3	0.509	1.042
4	0.463	0.681
6	0.437	0.587

Table 5: Training time (s) for Spectf database

q	2	3	4	6
Spam	0.0002	0.0004	0.0006	0.0016
Simulated	0.0012	0.0012	0.0012	0.0015
Forest	0.0024	0.0011	0.0014	0.0028
Diabetes	0.0150	0.0177	0.0174	0.0185
Spectf	0.0039	0.0019	0.0023	0.0032

Table 6: RMSE for different test data sets

identity covariance matrices and mean vectors equal to $\mathbf{0_{10}}$ and $\mathbf{1_{10}}$ respectively. We consider here $N = 40000$ observations. The gaussian kernel parameter σ is 3 and $C = 10$. Forest[2] dataset has been used for large scale SVM training (e.g.,Collobert et al., 2002). Following [5], we aim at separating class 2 from the other classes. $N = 20000$, 10 attributes, $\sigma = 100$ and $C = 1000$. For the Diabetes[3] dataset, the diagnostic, binary-valued variable investigated, is taken whether the patient shows signs of diabetes according to World Health Organization criteria. $N = 763$, 8 attributes, $\sigma = 8$ and $C = 10$. We use also the dataset Spectf heart[4], the data recorded using cardiac Single Proton Emission Computed Tomography (SPECT) images. Each patient is classified into one of two categories: normal and abnormal. ($N = 187, p = 44$, $\sigma = 10$ and $C = 50$).
A close look to the training time between SMGO and SVM-Light (Table 1, 2, 3, 4, 5) on all the experiments run shows that the gain in training time generally increases when the data set size increases.

[2]http://kdd.ics.uci.edu/databases/covertype/covertype.data.html
[3]archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes
[4]http://archive.ics.uci.edu/ml/datasets/SPECTF+Heart

In order to evaluate the performance degradation, we partitioned randomly the previous data sets in training and test data sets and we evaluated the Root Mean Square Error (RMSE)[5] on test data, for each experiment, between our results and the solution obtained with SVM-Light for different working set sizes (Table 6). The comparison has been done between SMGO and SVMLight. For $q = 1$, RMSE was not calculated since SVM-Light does not allow such a working set size. Note that, even if we obtain a very similar solution between SMGO and SVM-Light, we are not solving the same optimization problem.

5. CONCLUSION

In this paper, we have introduced a new and fast training algorithm for support vector detection. This algorithm ingeniously exploits the inequality constraint and the quadratic nature of the objective function. At each iteration, all the computations are done analytically and the solution, gradient and objective function are recursively determined. SMGO is easy to implement and does not require the storage of the Gram matrix.
Our objective was not to improve the performance of the detector obtained but to propose a faster training algorithm (also able to deal with large data sets) presenting no performance degradation compared to usual SVM. The interest of SMGO results from the formulation of the primal problem. This formulation allows the suppression of the equality constraint usually appearing in the dual formulation of the optimization problem. It is important to note that the primal is not the same as usual. Therefore, the solution obtained using the classical formulation of Support Vector Classifiers and our formulation might be different.
A comparison between SMGO and SVM-Light has been performed on different data sets. An advantage of our algorithm is that the minimum working set size equals 1, which, coupled with an analytical expression of the bias term, allows us to extend SMGO to an online version. The results obtained have shown that SMGO generally advantageously compares with SVM-Light. We have experienced that our algorithm is faster and faster compared to SVM-Light since data set size increases, so it is well adapted to large scale problems.
In our future work, we aim to extend SMGO to an online version and use kernel cache to improve training time execution.

REFERENCES

[1] A. Bordes, S. Ertekin, J. Weston, L. Bottou, Fast Kernel Classifiers with Online and Active Learning. *Journal of Machine Learning Research (JMLR)*, vol 6, pp.1579–1619, 2005.

[2] L. Bottou and O. Chapelle and D. Dennis and J. Weston, Large Scale Kernel Machines. *MIT Press*, Cambridge, MA, 2007.

[3] J. C. Burges, A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, vol 2, pp.121–167,1998.

[5]RMSE $= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_{SMGO}(\mathbf{x}_i) - f_{svmlight}(\mathbf{x}_i))^2}$

[4] R. Collobert, S. Bengio, SVMTorch: Support Vector Machines for large-scale regression problems. *Journal of Machine Learning Research*, vol 1, pp.143–160,2001.

[5] R. Collobert, S. Bengio, Y. Bengio, A parallel mixture of svms for very large scale problems. *Neural Computation*, vol 14, pp.1105–1114, 2002.

[6] M. Davy, F. Desobry, A. Gretton, C. Doncarli, An online support vector machine for abnormal events detection. *Signal processing* , Elsevier Science, ISSN 0165-1684, vol. 86, pp.2009-2025, 1997.

[7] T. Friess, N. Cristianini, C. Campbell, The Kernel-Adatron Algorithm: A Fast and Simple Learning Procedure for Support Vector Machines. *International Conference of Machine Learning Research (ICML)*, pp.188–196, 1998.

[8] T. Joachims, Making large-scale support vector machine learning practical. *Advances in Kernel Methods: Support Vector Machines*. B. Schölkopf, C. Burges, A. Smola, MIT Press, pp.69-184, 1999.

[9] Y. Lee and O. L. Mangasarian, RSVM: Reduced support vector machines. *Proc. First SIAM International Conference on Data Mining*, 2001.

[10] O. L. Mangasarian and D. R. Musicant, Successive overrelaxation for support vector machines. *Technical report*, University of Wisconsin, Computer Sciences Department, Madison, WI, USA, 1998.

[11] O. L. Mangasarian and G. Fung, Proximal support vector machine classifiers. *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, pp.77-86, 2001.

[12] M. Tohmé and R. Lengellé, F-SVC: a simple and fast training algorithm for soft margin support vector classification. *MLSP, Proc. IEEE international workshop on machine learning for signal processing*, 2008.

[13] M. Tohmé and R. Lengellé, F-SVR: a new learning algorithm for support vector regression. *ICASSP, Proc. IEEE international conference on acoustics speech and signal processing*, 2008.

[14] E. Osuna, R. Freund, F. Girosi, Training support vector machines: An application to face detection. *In: Proc Computer Vision and Pattern Recognition*, Puerto Rico, pp.130-136, 1997.

[15] J.C. Platt, Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods: Support Vector Learning*, MIT Press, pp.185-208, 1999.

[16] M. E. Mavroforakis and S. Theodoridis, A geometric approach to Support Vector Machine (SVM) classification. *IEEE Transactions on Neural Networks*, vol 17, pp.671-682, 2006.

[17] B. Schölkopf and K. R. Müller and A. J. Smola, Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, vol 10, pp.1299-1319, 1998.

[18] B. Schölkopf and J. C. Burges and A. J. Smola, Advances in kernel methods. *MIT Press*, Cambridge, MA, 1999.

[19] A.J. Smola, Learning with Kernels. *PhD thesis, Techische Universität Berlin* GMD, Birlinghoven, Germany, 1998.

[20] A. J. Smola and B. Schölkopf, A tutorial on support vector regression. *NeuroCOLT*, Royal Holloway College, University of London, UK, NC-TR-98-030, 1998.

[21] V. N. Vapnik, The nature of statistical learning theory. *Springer*, 1995.

[22] V. N.Vapnik, Statistical Learning Theory. *Wiley*, New York, 1998.

[23] V. N.Vapnik, Estimation of Dependences Based on Empirical Data. *Springer*, 2001.