International Journal of Intelligence Science, **2013, 3, 5-14**
http://dx.doi.org/10.4236/ijis.2013.31002 Published Online January 2013 (http://www.scirp.org/journal/ijis)

Support Vector Machines for Regression: A Succinct Review of Large-Scale and Linear Programming Formulations

Pablo Rivas-Perea[1], Juan Cota-Ruiz[2], David Garcia Chaparro[2], Jorge Arturo Perez Venzor[2], Abel Quezada Carreón[2], Jose Gerardo Rosiles[3]
[1]Department of Computer Science, School of Engineering & Computer Science, Baylor University, Waco, USA;
[2]Department of Electrical & Computer Engineering, Autonomous University of Ciudad Juárez, Ciudad Juárez, México
[3]Rosiles Consulting, El Paso, USA
Email: pablo_rivas_perea@baylor.edu, jcota@uacj.mx, dagarcia@uacj.mx, jorperez@uacj.mx, abquezad@uacj.mx, rosiles@ieee.org

Received October 5, 2012; revised November 13, 2012; accepted November 24, 2012

ABSTRACT

Support Vector-based learning methods are an important part of Computational Intelligence techniques. Recent efforts have been dealing with the problem of learning from very large datasets. This paper reviews the most commonly used formulations of support vector machines for regression (SVRs) aiming to emphasize its usability on large-scale applications. We review the general concept of support vector machines (SVMs), address the state-of-the-art on training methods SVMs, and explain the fundamental principle of SVRs. The most common learning methods for SVRs are introduced and linear programming-based SVR formulations are explained emphasizing its suitability for large-scale learning. Finally, this paper also discusses some open problems and current trends.

Keywords: Support Vector Machines; Support Vector Regression; Linear Programming Support Vector Regression

1. Introduction

Statistical pattern recognition methods based on optimization methods have shifted the paradigm in the computational intelligence area. The reason is that most learning algorithms proposed in the past 20 years focused on heuristics or on analogies with natural learning systems, such as evolutionary algorithms (EA) or neural networks (NN). EAs and NNs were mostly the result of biological findings and experimental tuning of parameters. However, the underlying reasons for their performance was not fully understood. Most of the work was based on designing heuristics to avoid local minima trapping in the training (design) process. In contrast, recent statistical pattern recognition algorithms overcome many of these disadvantages by using known theories in the mathematical sciences, e.g., Mercer kernels [1], Hilbert spaces [2], and Kernel expansion [3].

In the last decade, a theoretical basis for learning machines has been developed to explicitly maximize their performance [4]. The combination of learning machines with kernel functions has led to novel pattern recognition algorithms characterized by their formulation as convex optimization problems. Convex optimization problems have the advantage of being free from local minima. In particular, it is now possible to model non- linear pattern recognition systems under the convenience of a linear representation using kernel functions [3].

Support vector machines (SVMs) are arguably the best known example among kernel learning algorithms. Since their introduction in 1992 [5], SVMs have been studied, generalized, and applied to a number of problems. Currently, SVMs hold records in performance benchmarks for handwritten digit recognition [6], text categorization [7], information retrieval [8], and time-series prediction [9] and are commonly used in the analysis of DNA micro-array data [10]. Due to its novelty and potential, there are many areas of improvement, particularly in the area of numerical methods that helps with the problem posing [11] (e.g., vectorial calculus and algebra) and optimization [12] (e.g., quadratic and linear programming).

In 1995, Vapnik *et al.* [13] and later Smola *et al.* [14] explored and developed the SVM approach for regretssion problems. This approach is commonly known as Support Vector Regression (SVR), which increased the application range of SVMs since SVRs can also perform multi-class pattern recognition [15]. This type of machine is typically formulated using quadratic optimization under the umbrella of convex optimization. In return, the solution is always global and is easy to compute for very small problems. This paper is devoted to introduce, familiarize, and motivate the reader to the field of SVR, ultimately, for large-scale formulations posed as linear programs.

The rest of the paper is organized as follows. The state-of-the-art on training SVMs is summarized in Section 2, emphasizing the number of the training samples used. Section 3 defines the fundamental principle of SVRs. The two major learning methods for SVRs are introduced in Section 4, while in Section 5 it is introduced the linear programming-based SVR formulations, aiming to emphasize formulations suitable for large-scale learning. Finally, open problems, current trends, and conclusions are drawn in Section 6.

2. Large-Scale Support Vector Regression Training

The original training of an SVM [16] was only valid for small data sets and was designed to find the solution of a QP problem using a constrained conjugate gradient algorithm. The size of the problem is directly proportional to the number of training vectors, making its solution computationally intractable even for moderate size training sets.

Later Vapnik *et al.* [16] improved this method by considering only those variables associated to support vectors lying on the boundaries, whose gradient take them out of the feasible region. This modification allowed computational tractability of a limited number of large-scale problems (less than 5000 samples).

In 1997, Osuna *et al.* [17] proposed a decomposition method that achieves optimality by solving a sequence of smaller sub-problems from randomly selected samples out of a training dataset. The algorithm selects the support vector coefficients that are active on either side of the optimal hyperplane (in a two class problem), then proceeds iteratively until the completion of the problem size. This algorithm performs satisfactory in applications of up to 110,000 data points.

Later in 1999, Joachims introduced the concept of “shrinking” under the SVM training context and named it “SVMlight” [18]. The idea is to get rid of points that have less probability of becoming support vectors, thereby saving time in the optimization problem solution. Joachims’ method demonstrated to be faster than Osuna’s method. The author reported results for as many as 49,749 data points.

Also in 1999, Platt extended Osuna’s work with an algorithm called “sequential minimal optimization” (SMO) [19]. This is a famous algorithm implemented in several commercial and open-source software applications. The key idea behind Platt’s method is to decompose the large QP problem into a series of very small QP sub-problems. These sub-problems are as small as they can be solvewithout a QP solver, thus much faster. Platt reported as many as 60,000 data points.

Collobert *et al.* [20] reported in 2001 an adaptation of Joachims’ method for SVR problems named “SVMTorch” (see Section 4). The authors show the implementation of reduction algorithms and give mathematical proof of convergence. The authors also reported using as many as 60,000 data points.

Rifkin presented the “SVMFu” algorithm in 2002 [21]. This work reviews and synthesizes Osuna’s, Platt’s, and Joachims’ work. The key idea is to divide the large problem into sub-problems such that their Hessian matrices fit within memory limitations. In QP, Hessian matrices are useful since they describe the second partial derivatives of an *n*-dimensional function. Rifkin reports as many as 60,000 data points.

Mangasarian *et al.* in 2002 [22], explored the very first linear programming (LP) approach to kernel-based regression. The authors proposed a linear programming chunking (LPC) algorithm with guaranteed termination. The method demonstrated efficiency, but slow convergence solving linear programs with commercial optimization software. The authors report solving linear programs with as many as 16,000 data points in a total of 19 days.

In 2005, Drineas *et al.* [23] developed an algorithm to approximate a low rank kernel matrix. The overall computational time is reduced. The authors report an interesting derivation of their ideas based on the Nystron method.

Continuing with QP-based methods, in 2006, Hush *et al.* [24] proposed an algorithm that produces approximate solutions without compromising accuracy in a QP SVM problem. The authors propose a first stage where the algorithm provides an approximate solution for the QP dual, and a second stage that maps the QP-dual problem to the QP-primal problem, based on the duality theorem. The authors report over a 100,000 samples data-set.

In 2006, Sra [25] proposed an algorithm that explicitly takes advantage of the LP formulation for SVMs to produce an efficient training method for large-scale problems. The method converts the LP problem into a QP problem, applies Bregman [26] and Hildreth [27] decomposition, and then the QP problem is solved. The author reports solving a problem with as many as 20,000,000 samples.

For some time, no new LP approaches for SVR were developed until 2009, when Lu *et al.* [28] reported an SVR method based on a novel LP formulation. The authors present an alternative of the typical kernels and use wavelet-based kernels instead. The paper shows an application to nonlinear dynamical systems identification, but the authors do not elaborate on the data-set size.

In 2010, by the time this paper was already in progress, Torii *et al.* [29] published work on a novel LP formulation along with three decomposition methods for training an LP-SVM approach. The authors report as many as 60,000 data points.

The development found up to date does not report any advances of sub-problem-based SVR approaches entirely motivated and aimed for large-scale LP-SVR formulations since Mangasarian *et al.* [22]. Even the idea from Sra [25] that seems to work for up to 20,000,000 samples, fails to take advantage of the efficiency of LP solvers and still works for only a limited number of samples.

In spite of all these advances in SVMs and SVRs, training schemes for large-scale applications are still required. It is of high importance to address the problem of large-scale training as technology advances and large amounts of data is being stored for further analysis. For now, let us review briefly the concept of SVR.

3. Support Vector Regression Fundamental Principle

To begin with, let us consider the linear regression case where the dependency of a scalar observable d on a regressor $\boldsymbol{x}$ is denoted as follows:

$$d = \boldsymbol{w}^T \boldsymbol{x} + b, \tag{1}$$

where the parameter vector $\boldsymbol{w}$ and the bias b are the unknowns. The problem is to estimate $\boldsymbol{w}$ and b given N training samples $\mathcal{T} = \left\{ \left(\boldsymbol{x}_i, d_i \right) \right\}_{i=1}^{N}$ where the vector elements $\boldsymbol{x}_i$ are assumed to be statistically independent and identically distributed (iid). The problem formulated by Vapnik is aimed to minimize, on the variables $\boldsymbol{w}$ and b, the structural risk functional

$$R = \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\left|d_i - y_i\right|_\epsilon , \tag{2}$$

where the variable y_i is the estimator output produced in response to the input $\boldsymbol{x}_i$, that is $f\left(\boldsymbol{x}_i\right) \equiv \boldsymbol{w}^T\boldsymbol{x} + b = y_i$; the function $|\cdot|_\epsilon$ describes the ϵ-insensitive loss function:

$$L_\epsilon\left(d, f\left(\boldsymbol{x}\right)\right) \equiv \left|d - f\left(\boldsymbol{x}\right)\right|_\epsilon = \begin{cases} \left|d - f\left(\boldsymbol{x}\right)\right| - \epsilon & \text{for } \left|d - f\left(\boldsymbol{x}\right)\right| \geq \epsilon \\ 0 & \text{otherwhise} \end{cases}, \tag{3}$$

where ϵ is a prescribed parameter (see **Figure 1**).

The structural functional (2) can be expressed as an optimization problem in its primal form as follows [30]:

$$\begin{aligned} &\min_{\boldsymbol{w},b,L_\epsilon} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\left|d_i - y_i\right|_\epsilon \\ &\text{s.t.} \begin{cases} d_i - y_i \leq \epsilon - \xi_i \\ y_i - d_i \leq \epsilon - \xi_i^* \end{cases} \quad \xi_i, \xi_i^* \geq 0 \\ &\text{for} \quad i = 1, 2, \cdots, N \end{aligned}$$

where the summation in the cost function accounts for the ϵ-insensitive training error, which forms a tube where the solution is allowed to be defined without penalization, as shown in **Figure 2** for the linear case, and in **Figure 3** for the non-linear regression case. The constant $C > 0$ describes the tradeoff between the training error and the penalizing term $\|\boldsymbol{w}\|_2^2$. The term $\|\boldsymbol{w}\|_2^2$ is penalized to enforce a sparse solution on $\boldsymbol{w}$. The variables ξ_i and ξ_i^* are two sets of nonnegative slack variables that describe the ϵ-insensitive loss function.

The objective function in the primal can be rewritten in terms of the slack variables ξ_i and ξ_i^*, by observing the restrictions of the primal and the definition of the ϵ-insensitive function, and thus defining $\xi = d_i - y_i - \epsilon$ and $\xi^* = y_i - d_i - \epsilon$. Then one obtains another common version of the primal problem as follows:

$$\begin{aligned} &\min_{\boldsymbol{w},b,\xi,\xi^*} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) \\ &\text{s.t.} \begin{cases} d_i - w^T x - b \leq \epsilon - \xi_i \\ w^T x + b - d_i \leq \epsilon - \xi_i^* \end{cases} \quad \xi, \xi^* \geq 0 \\ &\text{for} \quad i = 1, 2, \cdots, N. \end{aligned} \tag{4}$$

Next, we define the dual problem using the Lagrange multipliers method, where the following Lagrangian function is defined:

$$\begin{aligned} &L\left(\boldsymbol{w}, b, \xi, \xi^*, \alpha, \alpha^*, \gamma, \gamma^*\right) \\ &= \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{N}\left(\gamma_i\xi_i + \gamma_i^*\xi_i^*\right) \\ &\quad - \sum_{i=1}^{N}\alpha_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b - d_i + \epsilon + \xi_i\right) \\ &\quad - \sum_{i=1}^{N}\alpha_i^*\left(d_i - \boldsymbol{w}^T\boldsymbol{x}_i - b + \epsilon + \xi_i^*\right), \end{aligned} \tag{5}$$

where $\gamma_i, \gamma_i^*, \alpha_i,$ and α_i^* denote the Lagrange multipliers associated with the objective function and constraints respectively. The associated stationary points are

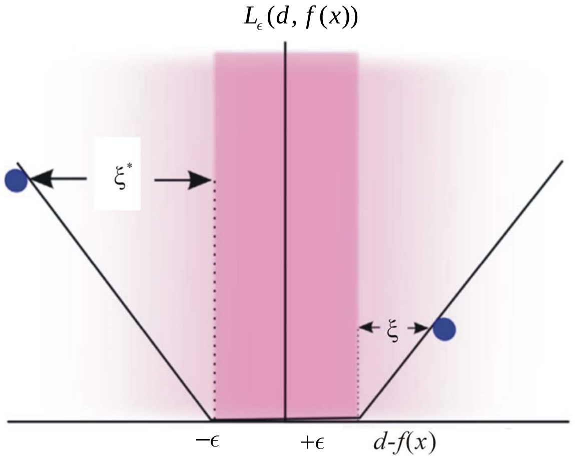

Figure 1. Illustration of the ϵ-insensitive loss function.

defined by the following partial derivatives:

$$\frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right)\boldsymbol{x}_i = 0, \quad \text{(6a)}$$

$$\frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial b} = \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right) = 0, \quad \text{(6b)}$$

$$\frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \xi} = C1 - \alpha - \gamma = 0, \quad \text{(6c)}$$

$$\frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \xi^*} = C1 - \alpha^* - \gamma^* = 0. \quad \text{(6d)}$$

Then, the dual problem using the method of Lagrange multipliers is stated as follows:

$$\max_{\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*} L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)$$
$$\text{s.t.} \begin{cases} \frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \boldsymbol{w}} = 0 \\ \frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial b} = 0 \\ \frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \xi} = 0 \\ \frac{\partial L\left(\boldsymbol{w},b,\xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^*\right)}{\partial \xi^*} = 0 \end{cases} \quad \xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^* \geq 0, \quad (7)$$

and the following problem is the expanded version of problem (7) above:

$$\max_{\substack{\boldsymbol{w},b,\xi,\xi^*,\\ \alpha,\alpha^*,\gamma,\gamma^*}} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{N}\left(\gamma_i\xi_i + \gamma_i^*\xi_i^*\right)$$
$$-\sum_{i=1}^{N}\alpha_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b - d_i + \epsilon + \xi_i\right)$$
$$-\sum_{i=1}^{N}\alpha_i^*\left(d_i - \boldsymbol{w}^T\boldsymbol{x}_i - b + \epsilon + \xi_i^*\right) \quad (8)$$
$$\text{s.t.} \begin{cases} \boldsymbol{w} - \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right)\boldsymbol{x}_i = 0 \\ \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right) = 0 \\ C1 - \alpha - \gamma = 0 \\ C1 - \alpha^* - \gamma^* = 0 \end{cases} \quad \xi,\xi^*,\alpha,\alpha^*,\gamma,\gamma^* \geq 0$$
$$\text{for} \quad i = 1,2,\cdots,N.$$

However, it is possible to remove three constraints by noticing from (6a) that one can solve for $\boldsymbol{w}$ as follows:

$$\boldsymbol{w} = \sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right)\boldsymbol{x}_i, \quad \text{(9a)}$$

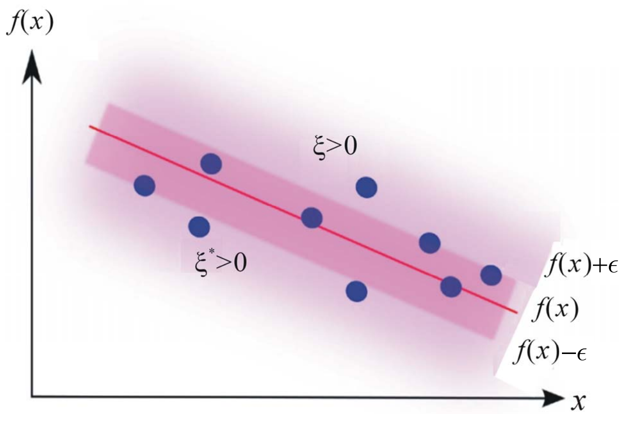

Figure 2. Linear regression: An ϵ -insensitive tube, fitted to the data points shown as circles.

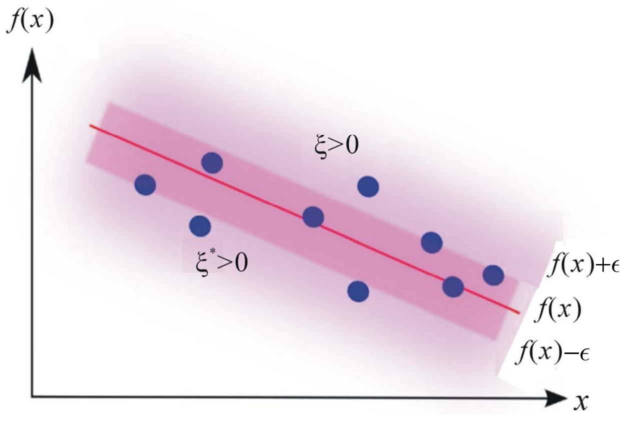

Figure 3. Non-linear regression: A regression curve along with the ϵ -insensitive tube.

and also one can solve for both γ and γ^* from (6c) and (6d) as

$$\gamma = C1 - \alpha, \quad \text{(9b)}$$

$$\gamma^* = C1 - \alpha^*, \quad \text{(9c)}$$

which also yields the boundary condition $0 \leq \alpha$, $\alpha^* \leq C1$. If we substitute the Equalities (9a)-(9c) into the objective function, and perform some analytic operations, we arrive to the following well known reduced dual problem [30]:

$$\max_{\alpha^*,\alpha} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)\boldsymbol{x}_i^T\boldsymbol{x}_j$$
$$-\epsilon\sum_{i=1}^{N}\left(\alpha_i + \alpha_i^*\right) + \sum_{i=1}^{N}d_i\left(\alpha_i - \alpha_i^*\right) \quad (10)$$
$$\text{s.t.} \quad \sum_{i=1}^{N}\left(\alpha_i - \alpha_i^*\right) = 0 \quad 0 \leq \alpha_i,\ \alpha_i^* \leq C,$$
$$\text{for} \quad i = 1,2,\cdots,N.$$

Problems (4) and (10) solve the linear regression problems, and for the non-linear regression case one just introduces a kernel function formulation. For the primal case, the sole modification is on the restrictions which are redefined as follows:

$$\boldsymbol{w}^T k(\boldsymbol{x}_i,\cdot) + b - d_i \le \epsilon - \xi_i, \tag{11a}$$

$$d_i - \boldsymbol{w}^T k(\boldsymbol{x}_i,\cdot) - b \le \epsilon - \xi_i^*, \tag{11b}$$

for $i, j = 1,2,\cdots,N$. For the dual problem the objective function is redefined as

$$\max_{\alpha^*,\alpha} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)k(\boldsymbol{x}_i,\boldsymbol{x}_j) -\epsilon\sum_{i=1}^{N}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{N}d_i(\alpha_i-\alpha_i^*). \tag{12}$$

This formulation is used to implement learning methods discussed next.

4. Support Vector Regression Learning Methods

Support Vector Machines for Regression (SVRs) share the same advantages and disadvantages as SVMs. That means, SVMs training methods can be extended to SVRs whenever they are not directly dependent on the particularities of the SVM problem. Since most SVM training methods depend on the particular mathematical formulation of SVM, very few of them can be extended to SVR.

Collobert *et al.* reported in 2001 an adaptation of Joachims' SVM method for SVR problems [20]. The authors start with the primal problem in (4), and then they reformulate the dual (10) with a vector-matrix notation in order to have the following Quadratic Programming (QP) minimization problem:

$$\begin{aligned}\min_{\alpha^*,\alpha} Q(\alpha,\alpha^*) &= \frac{1}{2}(\alpha^*-\alpha)^T \boldsymbol{K}(\alpha^*-\alpha) \\ &\quad -(\alpha^*-\alpha)^T\boldsymbol{d}+\epsilon(\alpha^*-\alpha)^T 1 \\ \text{s.t.}\quad & (\alpha-\alpha^*)^T 1 = 0 \\ & \alpha_i,\alpha_i^* \le C,\quad -\alpha_i,-\alpha_i^*\le 0 \\ \text{for}\quad & i=1,2,\cdots,N,\end{aligned} \tag{13}$$

where $\boldsymbol{K}$ is a kernel matrix.

Next, the authors perform the same decomposition proposed by Osuna [17] defining the working set $\mathcal{B}$ and the fixed set $\mathcal{M}$, where the size of the working set is $|\mathcal{B}|$. Also, the authors extend Joachims' idea of the steepest descent direction $\boldsymbol{p}$ to select the working set at each iteration of the SVR dual problem. That is, the problem described in [18] is reformulated as follows:

$$\begin{aligned}&\min \nabla Q(\alpha,\alpha^*)^T \boldsymbol{p} \\ &\text{s.t.}\ \begin{cases}\sum_{i=1}^{N} p_i - p_i^* = 0 \\ p_i \ge 0 \ \text{ for } i:\alpha_i = 0 \\ p_i^* \ge 0 \ \text{ for } i:\alpha_i^* = 0 \qquad \boldsymbol{p}\le 1,\ \boldsymbol{p}\ge -1, \\ p_i \le 0 \ \text{ for } i:\alpha_i = C \\ p_i^* \le 0 \ \text{ for } i:\alpha_i^* = C \end{cases}\end{aligned} \tag{14}$$

where $\boldsymbol{p} = (p_1,\cdots,p_N,p_1^*,\cdots,p_N^*)$, and the size of the working set is given the by $|\mathcal{B}| \equiv |p_i|$ for all $p_i \ne 0$.

Finally, the authors give proof of convergence following Platt;s SMO idea selecting a working set size of 2 samples. The problems with this implementation are inherited from Joachims' and augmented due to the complexity of the SVR definition. Particularly, the issues with this algorithm are the inherent QP solution process and the additional steepest descent process. Furthermore, they introduced the concept of shrinking, that is, to permanently remove from the working set the bounded support vectors after a number of iterations. Although the heuristic of shrinking dramatically speeds up the training process, convergence is not guaranteed anymore. However, Collobert's *et al*. shrinking method is still the most popular large-scale SVR training strategy.

5. Linear Programming Support Vector Regression

Linear Programing (LP) problems can be stated in the standard form

$$\begin{aligned}&\max_{z\in\mathbb{R}^n} \boldsymbol{c}^T\boldsymbol{z} \\ &\text{s.t.}\quad \boldsymbol{Az}\le\boldsymbol{b}\quad \boldsymbol{z}\ge\boldsymbol{0},\end{aligned} \tag{15}$$

where $\boldsymbol{z}\in\mathbb{R}^n$ is a vector containing the unknowns, $\boldsymbol{c}\in\mathbb{R}^n$ and $\boldsymbol{b}\in\mathbb{R}^m$ are vectors of known parameters, and $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ is a matrix of known coefficients associated with $\boldsymbol{z}$ through a linear relationship. However, there is also the canonical form

$$\begin{aligned}&\min_{z\in\mathbb{R}^n} \boldsymbol{c}^T\boldsymbol{z} \\ &\text{s.t.}\quad \boldsymbol{Az}=\boldsymbol{b}\quad \boldsymbol{z}\ge\boldsymbol{0},\end{aligned} \tag{16}$$

In fact, there is currently no agreement as to which of the previous problems is the canonical or standard form. In this paper, problem (16) will be regarded as the LP primal in its canonical form.

By introducing the Lagrange multipliers $(\boldsymbol{\lambda},\boldsymbol{s})$ into the primal problem above, one obtains the following dual

$$\begin{aligned}&\max_{\lambda} \boldsymbol{b}^T\boldsymbol{\lambda} \\ &\text{s.t.}\quad \boldsymbol{A}^T\boldsymbol{\lambda}+\boldsymbol{s}=\boldsymbol{c}\quad \boldsymbol{s}\ge\boldsymbol{0},\end{aligned} \tag{17}$$

where $\boldsymbol{\lambda}$ is a vector of dual variables defined over $\mathbb{R}^m$

and s is a slack vector variable in $\mathbb{R}^n$ .The solution to the primal problem is denoted as $\boldsymbol{z}^*$, and the solution to the dual problem is denoted as $(\boldsymbol{\lambda}^*, \boldsymbol{s}^*)$. The duality theorem states that $\boldsymbol{c}^T\boldsymbol{z}^* = \boldsymbol{b}^T\boldsymbol{\lambda}^*$, which means that the solution $\boldsymbol{z}^*$ also solves the dual, and the solution of the dual $(\boldsymbol{\lambda}^*, \boldsymbol{s}^*)$ also solves the primal [31].

The idea of Linear Programming Support Vector Regression (LP-SVR) is to pose the SVR problem (4) as a linear program either in its canonical (16) or standard (15) forms. Before we continue with linear programming and SVRs, we need to inform the reader of the complexity of the typical SVR problems to justify further implementations with interior point methods. The following paragraphs will inform the reader in regard to the computational expenses on SVR learning methods.

5.1. SVR Learning

Let n and m be the number of variables and constraints respectively in an SVR-based optimization problem. To solve such optimization problems, there are three known strategies. First, if the problem can fit within system memory, the strategy is to directly use interior point methods which have memory consumption of $\mathcal{O}(n^2)$. Second, if the problem is of medium size, active-set methods are proven to be more appropriate [32]. They need $\mathcal{O}(n_{SVs}^2)$ memory, where n_{SVs} is characterized by the number of support vectors of the problem. That is, active-set methods deal with the complete set of support vectors only. Third, for very large datasets, one currently uses working set methods (i.e., decomposition methods). Among these are Collobert's [20] or Mangasarian's [33]. These methods have memory consumption of $\mathcal{O}(n)$. The training strategies are summarized in **Figure 4**.

Now, let us review the most common LP-SVR formulations that will provide the reader with the general knowledge on LP-SVRs.

5.2. *ν*-LPR

Smolaetal in 1999 [34], first introduced a linear programming approach for SVR. Their formulation uses the following ν-SVR model:

$$\begin{aligned}
&\min_{\alpha^+,\alpha^-,b,\xi,\xi^*,\epsilon} \sum_{i=1}^{N} \frac{1}{N}\left(\alpha_i^+ + \alpha_i^-\right) + \frac{C}{N}\left(\xi_i + \xi_i^*\right) + C\nu\epsilon \\
&\text{s.t.} \quad \begin{cases} -d_i + \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) + b \le \epsilon + \xi_i \\ d_i - \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) - b \le \epsilon + \xi_i^* \end{cases} \qquad (18) \\
&\qquad \alpha_i^+, \alpha_i^-, \xi_i, \xi_i^*, \epsilon \ge 0 \\
&\text{for} \quad i = 1, 2, \cdots, N,
\end{aligned}$$

where $\nu \in [0,1]$ can be used to control the number of errors in ϵ . In this way, the value of ϵ is automatically estimated. The ν-SVR comes from a special type of SVM called ν-SVM [32]. The authors named their formulation as "ν-LPR", as in "nu-linear programming regression". Smola *et al.* poses problem (18) as a linear program in the following form:

$$\begin{aligned}
&\min_{z} \boldsymbol{c}^T\boldsymbol{z} \\
&\text{s.t.} \ \boldsymbol{A}\boldsymbol{z} + \boldsymbol{B}\boldsymbol{r} + \boldsymbol{b} - \boldsymbol{u} = 0 \quad \boldsymbol{z}, \boldsymbol{u} \ge 0,
\end{aligned} \qquad (19)$$

where $\in \mathbb{R}^{4N+1}$, $\boldsymbol{r} \in \mathbb{R}$, $\boldsymbol{u} \in \mathbb{R}^{2N}$, and the matrices $\boldsymbol{A} \in \mathbb{R}^{2N\times 4N+1}$, $\boldsymbol{B}, \boldsymbol{b} \in \mathbb{R}^{2N}$, $\boldsymbol{c} \in \mathbb{R}^{4N+1}$ are denoted as follows:

$$\boldsymbol{c} = \begin{pmatrix} 1 & 1 & C1 & C1 & C\nu N \end{pmatrix}, \qquad (20a)$$

$$\boldsymbol{c} = \begin{pmatrix} 1 & 1 & C1 & C1 & C\nu N \end{pmatrix}, \qquad (20b)$$

$$\mathbf{A} = \begin{pmatrix} -\boldsymbol{K} & \boldsymbol{K} & 1 & 0 & 1 \\ \boldsymbol{K} & -\boldsymbol{K} & 0 & 1 & 1 \end{pmatrix}, \qquad (20c)$$

$$\boldsymbol{B} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \text{and } \boldsymbol{b} = \begin{pmatrix} \boldsymbol{d} \\ -\boldsymbol{d} \end{pmatrix}. \qquad (20d)$$

The dual problem is derived in order to use a primal-dual interior point solver with a predictor-corrector strategy. In doing this, they also derive the corresponding Karush-Kuhn-Tucker (KKT) conditions.

Smola *et al.* [34] suggest that the parameter ν is proportional to the number of support vectors. Although the main contribution is to show the efficiency of an SVR using LP, a critique is that the authors eliminated one parameter by adding another, *i.e.*, ν instead of ϵ. Moreover, the problems with LP-SVR models is that the program dimension is very large, i.e., $4N+2$. As a consequence, the authors report performing regression with training sets consisting of only 50 and 80 data points.

5.3. LP-SVR with Modified LPC

Mangasarian *et al.* [22] in 2002 pioneered the linear programming formulations for large-scale SVRs. The authors analyzed the ν-LPR [34] method and provided means for training with large-scale datasets. The SVR formulation they use is the following:

$$\begin{aligned}
&\min_{\alpha^+,\alpha^-,b,t,\epsilon} \sum_{i=1}^{N} \frac{1}{N}\left(\alpha_i^+ + \alpha_i^-\right) + \frac{C}{N}t_i + C(1-\mu)\epsilon \\
&\text{s.t.} \quad \begin{cases} -d_i + \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) + b \le \epsilon + t_i \\ d_i - \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) - b \le \epsilon + t_i \end{cases} \qquad (21) \\
&\qquad \alpha_i^+, \alpha_i^-, t_i, \epsilon \ge 0 \\
&\text{for} \quad i = 1, 2, \cdots, N,
\end{aligned}$$

where $\mu \in [0,1]$ is a positive parameter that drives the tolerance error ϵ : if $\mu \to 1$ the errors are highly

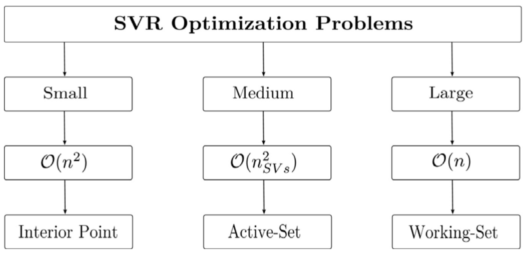

Figure 4. Depending on the problem size, one may find three different strategies for SVR training.

penalized. If one take a closer look to problems (18) and (21), one notices that $t_i = \left(\xi_i + \xi_i^*\right)$. In fact, the authors show that problems (18) and (21) are equivalent. The significance of this formulation is the use of a modified version of the Linear Programming Chunking (LPC) algorithm [33]. The LPC algorithm was originally utilized to solve the problem with a subset of the constraints. However, Mangasarian *et al.* also proposed a modified version that constraint the variables as well. This modification promotes a smaller version of the problem, hence faster computations in terms of memory allocation.

5.4. Wavelet Kernel-Based LP-SVR

In 2009, Lu *et al.* [28] reported an SVR method based entirely in the LP primal in its standard form. The authors present an alternative to the typical kernels and use wavelet-based kernels instead. The paper shows an application to nonlinear dynamical systems identification [28]. The authors start by changing the ℓ_2^2-norm by the ℓ_1-norm, and also use the notion of the kernel functions for nonlinear problems; and they reformulate problem (4) as follows:

$$\min \frac{1}{2}\|\alpha\|_1 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right)$$
$$\text{s.t.} \begin{cases} d_i - \sum_{j=1}^{N}\alpha_j k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) \le \epsilon + \xi_i \\ -d_i + \sum_{j=1}^{N}\alpha_j k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) \le \epsilon + \xi_i^* \end{cases} \quad \xi, \xi^* \ge 0 \qquad (22)$$
$$\text{for} \quad i = 1, 2, \cdots, N,$$

where α comes from the Representer Theorem notation [35]. Then, the authors arrive at the following LP problem in its primal standard form:

$$\min_{\alpha^+, \alpha^-, \xi} \begin{pmatrix} 1 & 1 & 2\mathbf{C} \end{pmatrix} \begin{pmatrix} \alpha^+ \\ \alpha^- \\ \xi \end{pmatrix}$$
$$\text{s.t.} \left\{ \begin{pmatrix} \boldsymbol{K} & -\boldsymbol{K} & -\boldsymbol{I} \\ -\boldsymbol{K} & \boldsymbol{K} & -\boldsymbol{I} \end{pmatrix} \begin{pmatrix} \alpha^+ \\ \alpha^- \\ \xi \end{pmatrix} \le \begin{pmatrix} \boldsymbol{d} + \epsilon 1 \\ \epsilon 1 - \boldsymbol{d} \end{pmatrix}, \right. \qquad (23)$$

where $\alpha_i = \alpha_i^+ - \alpha_i^-$; that is, the decomposition of the positive α_i^+ and the negative α_i^- part of α_i. Then, the authors show that this formulation performs around 25 times faster than the QP-SVR. No details were given with respect to the solver used, and no proof of convergence was addressed. The authors also show that in measuring the root mean squared error (RMSE), the QP-SVR performs better than their LP-SVR approach. However, they also report that this formulation requires fewer support vectors. Hence, a sparser solution is obtained, having the advantage of a reduced number of support vectors, which also represents computationally efficient solutions.

5.5. ℓ_1-Norm LP-SVR

In 2010, Zhang *et al.* [12] performed a study in regard to the sparseness of ℓ_1-norm-based SVMs and SVRs. The authors reported the following formulation of an ℓ_1-norm SVR:

$$\max_{\alpha^\pm, b^\pm, \xi^{(*)}, \boldsymbol{u}^\pm} -\sum_{i=1}^{N}\frac{1}{N}\left(\alpha_i^+ + \alpha_i^-\right) + \delta\left(b^+ + b^-\right) - C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right)$$
$$\text{s.t.} \begin{cases} d_i - \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) \\ -\left(b^+ - b^-\right) + u_i = \epsilon + \xi_i \\ -d_i + \sum_{j=1}^{N}\left(\alpha_j^+ - \alpha_j^-\right)k\left(\boldsymbol{x}_j, \boldsymbol{x}_i\right) \\ +\left(b^+ - b^-\right) + u_i = \epsilon + \xi_i^* \end{cases} \qquad (24)$$
$$\alpha_i^+, \alpha_i^-, \xi_i, \xi_i^*, \epsilon, b^+, b^-, u_i^+, u_i^- \ge 0$$
$$\text{for} \quad i = 1, 2, \cdots, N,$$

where $\delta \in \mathbb{R}$ (typically small: $\delta = 1\times10^{-11}$) is used to avoid possible infinite optimal solutions, and u_i is a slack variable. Then, defining

$$\boldsymbol{c} = \begin{pmatrix} -1 & -1 & -\delta & -\delta & -C1 & -C1 & 0 & 0 \end{pmatrix} \qquad (25a)$$

$$\boldsymbol{z} = \begin{pmatrix} \boldsymbol{\alpha}^+ & \boldsymbol{\alpha}^- & b^+ & b^- & \xi & \xi^* & \boldsymbol{u}^+ & \boldsymbol{u}^- \end{pmatrix}, \qquad (25b)$$

$$\boldsymbol{A} = \begin{pmatrix} -\boldsymbol{K} & \boldsymbol{K} & -1 & 1 & -\boldsymbol{I} & 0 & \boldsymbol{I} & 0 \\ \boldsymbol{K} & -\boldsymbol{K} & 1 & -1 & 0 & \boldsymbol{I} & 0 & \boldsymbol{I} \end{pmatrix} \qquad (25c)$$

$$\boldsymbol{b} = \begin{pmatrix} 1\epsilon - \boldsymbol{d} \\ 1\epsilon - \boldsymbol{d} \end{pmatrix}, \qquad (25d)$$

the authors proceeded to solve the following linear programming variation:

$$\max_{z} \boldsymbol{c}^T\boldsymbol{z}$$
$$\text{s.t.} \;\; \boldsymbol{Az} = \boldsymbol{b} \quad \boldsymbol{z} \ge \boldsymbol{0}. \qquad (26)$$

The evident problem with this formulation is the size of the problem: $6N+2$ variables and $2N$ constraints. Therefore, the author only reports experiments for small size problems; e.g., 351, 297, and 345 data points, using an RBF kernel. The poor formulation makes the size problem more evident since the authors used the simplex method. As a consequence very large training times will be experienced. One obvious critique is the inclusion of δ, making the set of hyper-parameters $[\epsilon, C, \sigma, \delta]$ to be chosen more heuristically.

5.6. Large-Scale LP-SVR

Recently, in 2011, Rivas *et al.* [36-37] attempted to solve the difficulties of previous formulations, particularly those that may hinder the learning phase. Similarly to previous formulations, the authors [36-37] assumed that the slack variables ξ_i, ξ_i^* can be expressed as simply $2\xi_i$ (since $\xi_i, \xi_i^* = 0$). Then, introduced a slack variable $\boldsymbol{u}$ to avoid inequalities in the SVR formulation. As a consequence of these assumptions, the following optimization problem was proposed:

$$\min_{\alpha^{\pm}, b^{\pm}, \xi^{(*)}, \boldsymbol{u}} \sum_{i=1}^{N} \left(\alpha_i^+ + \alpha_i^- + 2C\xi_i \right)$$

$$\text{s.t.} \quad \begin{cases} -\sum_{i=1}^{N} \left(\alpha_i^+ - \alpha_i^- \right) k\left(\boldsymbol{x}_j, \boldsymbol{x}_i \right) \\ -b^+ + b^- - \xi_j + u_i = \epsilon - d_j \\ \sum_{i=1}^{N} \left(\alpha_i^+ - \alpha_i^- \right) k\left(\boldsymbol{x}_j, \boldsymbol{x}_i \right) \\ +b^+ - b^- - \xi_j + u_i = \epsilon + d_j \end{cases} \tag{27}$$

$$\alpha_j^+, \alpha_j^-, b^+, b^-, \xi_j, u_j \ge 0$$

$$\text{for} \quad j = 1, 2, \cdots, N.$$

Then problem (27) was posed as a linear programming problem by defining the following equalities:

$$\boldsymbol{c} = \begin{pmatrix} 1 & 1 & 0 & 0 & 2\boldsymbol{C} & 0 \end{pmatrix}, \tag{28a}$$

$$\boldsymbol{z} = \begin{pmatrix} \boldsymbol{\alpha}^+ & \boldsymbol{\alpha}^- & b^+ & b^- & \xi & \boldsymbol{u} \end{pmatrix}, \tag{28a}$$

$$\boldsymbol{A} = \begin{pmatrix} -\boldsymbol{K} & \boldsymbol{K} & -1 & 1 & -\boldsymbol{I} & \boldsymbol{I} \\ \boldsymbol{K} & -\boldsymbol{K} & 1 & -1 & -\boldsymbol{I} & \boldsymbol{I} \end{pmatrix}, \tag{28c}$$

$$\boldsymbol{b} = \begin{pmatrix} 1\epsilon - \boldsymbol{d} \\ 1\epsilon + \boldsymbol{d} \end{pmatrix}, \tag{28d}$$

where $\boldsymbol{A} \in \mathbb{R}^{(2N)\times(4N+2)}$, $\boldsymbol{b} \in \mathbb{R}^{2N}$, $\boldsymbol{z}, \boldsymbol{c} \in \mathbb{R}^{4N+2}$.

In comparison with the ν-LPR formulation by Smola *et al.* [34], problem (27) 1) uses the canonical formulation, 2) computes b, and $\boldsymbol{u}$ implicitly, 3) does not compute ϵ implicitly, 4) does not require the parameter ν, 5) promotes efficiency in the sense of using only one ξ, and 6) is a lower dimensional problem.

In comparison with Mangasarian *et al.* [22], problem (27) 1) uses the canonical formulation, 2) computes b implicitly, 3) does not compute ϵ implicitly, and 4) does not require the parameter μ. By 3) and 4) the experimenter has more control of the sparseness of the solution [12]. In this case sparseness means fewer number of support vectors.

Similarly, problem (27) in comparison to Lu *et al.* [28] Rivas' *et al.* [36-37] LP-SVR formulation (27) 1) uses the canonical formulation and 2) computes b implicitly. By 2) the linear program (LP) size is reduced by a factor of N^2+N. In comparison with the ℓ_1-norm LP-SVR formulation by Zhang *et al.* [12] problem (27) does not require parameter δ and is more efficient in several ways: 1) uses only one ξ, 2) avoids penalization of b, 3) reduces computational efforts by forcing positivity in $\boldsymbol{u}$ which reduces the LP problem size by $2N^2+2N$, and 4) is a smaller problem. Finally, in order to find the solution to the problem, Rivas *et al.* [36-37] use an interior point methods-based solver to find the variables that satisfy the following KKT conditions:

$$\boldsymbol{A}^T \boldsymbol{\lambda} + \boldsymbol{s} = \boldsymbol{c}, \tag{29a}$$

$$Az = \boldsymbol{b}, \tag{29b}$$

$$z_i s_i = 0, \tag{29c}$$

$$\left(\boldsymbol{z}, \boldsymbol{s} \right) \ge 0, \tag{29d}$$

$$\text{for } i = 1, 2, \cdots, n,$$

where the equality $z_i s_i$ implies that one of both variables must be zero. This equality is typically referred to as the complementarity condition. Note that the KKT conditions depend on the variables $(\boldsymbol{z}, \boldsymbol{\lambda}, \boldsymbol{s})$, and if the set of solutions $(\boldsymbol{z}^*, \boldsymbol{\lambda}^*, \boldsymbol{s}^*)$ satisfy all the conditions, the problem is said to be solved. The set $(\boldsymbol{z}^*, \boldsymbol{\lambda}^*, \boldsymbol{s}^*)$ is known as a primal-dual solution.

6. Conclusions, Current Trends, and Open Problems

In this paper we have explained the efforts being made towards a solution of large-scale learning using SVRs. We have reviewed the most commonly used formulations of SVRs emphasizing its usability on large-scale applications. We reviewed the general concept of SVMs, addressed the state-of-the-art on training methods SVMs, and explained the fundamental principle of SVRs. The two most common learning methods for SVRs were introduced and linear programming-based SVR formulations are explained emphasizing its suitability for large-scale learning. The Support Vector learning field is still very active. As a consequence, the treatment of large-scale SVM remains being of interest to researchers in the field (see reference [38]). However, a number of open issues that have to be addressed still persist. Although recently, SVR algorithmic development seems to be at a more stable stage, one issue is to find whether linear programming SVR approaches will lead to more satisfactory results [12,14]. The problem of empirical tuning [3,14] of SVR hyper-parameters (e.g., C,σ), has to be devised to make SVRs less dependent on the skill of the experimenter [14,39]. Optimization techniques developed within the context of SVRs are also required in order to improve treatment of large datasets [14]. This may be done in combination with reduced set methods for speeding up the training phase for large datasets. This topic is of huge importance as machine learning applications demand algorithms that are capable of dealing with datasets that are at least larger than 1 million samples [14]. Other problems still considered open include the following: more data dependent generalization bounds, efficient training algorithms, and automatic kernel selection procedures. The authors would like to kindly invite highly motivated researchers to contribute in these areas as they can lead to a settlement on large-scale SV-based learning methods.

7. Acknowledgements

The author P. R. P. performed part of this work while at NASA Goddard Space Flight Center as part of the Graduate Student Summer Program (GSSP 2009) under the supervision of Dr. James C. Tilton. This work was supported in part by the National Council for Science and Technology (CONACyT), Mexico, under Grant 193324/303732 and mentored by Dr. Greg Hamerly who is with the department of Computer Science at Baylor University. Finally, the authors acknowledge the support of the Large-Scale Multispectral Multidimensional Analysis (LSMMA) Laboratory (www.lsmmalab.com).

REFERENCES

[1] J. Mercer, “Functions of Positive and Negative Type, and Their Connection with the Theory of Integral Equations,” *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, Vol. 209, 1909, pp. 415-446. doi:10.1098/rsta.1909.0016

[2] R. Courant and D. Hilbert, “Methods of Mathematical Physics,” Interscience, New York, 1966.

[3] J. Shawe-Taylor and N. Cristianini, “Kernel Methods for pattern Analysis,” Cambridge University Press, New York, 2004. doi:10.1017/CBO9780511809682.002

[4] N. Cristianini and B. Scholkopf, “Support Vector Machines and Kernel Methods: The New Generation of Learning Machines,” *Ai Magazine*, Vol. 23, No. 3, 2002, p. 31.

[5] B. E. Boser, I. M. Guyon and V. N. Vapnik, “A Training Algorithm for Optimal Margin Classifiers,” *Proceedings of the* 5*th Annual Workshop on Computational Learning Theory*, Pittsburgh, July 1992, pp. 144-152.

[6] K. Labusch, E. Barth and T. Martinetz, “Simple Method for High-Performance Digit Recognition Based on Sparse Coding,” *IEEE Transactions on Neural Networks*, Vol. 19, No. 11, 2008, pp. 1985-1989. doi:10.1109/TNN.2008.2005830

[7] H. Al-Mubaid and S. Umair, “A New Text Categorization Techniqueusing Distributional Clustering and Learning Logic,” *IEEE Transactions on Knowledge and Data Engineering*, Vol. 18, No. 9, 2006, pp. 1156-1165. doi:10.1109/TKDE.2006.135

[8] K. Wu and K.-H. Yap, “Fuzzy SVM for Content-Based Image Retrieval: A Pseudo-Label Support Vector Machine Framework,” *IEEE Computational Intelligence Magazine*, Vol. 1, No. 2, 2006, pp. 10-16. doi:10.1109/MCI.2006.1626490

[9] N. Sapankevych and R. Sankar, “Time Series Prediction Using Support Vector Machines: A Survey,” *IEEE Computational Intelligence Magazine*, Vol. 4, No. 2, 2009, pp. 24-38. doi:10.1109/MCI.2009.932254

[10] D. Peterson and M. Thaut, “Model and Feature Selection in Microarray Classification,” *Proceedings of the* 2004 *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, La Joll, 7-8 October 2004, pp. 56-60. doi:10.1109/CIBCB.2004.1393932

[11] A. Sanchez and V. David, “Advanced Support Vector Machines and Kernel Methods,” *Neurocomputing*, Vol. 55, No. 1-2, 2003, pp. 5-20. doi:10.1016/S0925-2312(03)00373-4

[12] L. Zhang and W. Zhou, “On the Sparseness of 1-Norm Support Vector Machines,” *Neural Networks*, Vol. 23, No. 3, 2010, pp. 373-385. doi:10.1016/j.neunet.2009.11.012

[13] V. N. Vapnik, “The Nature of Statistical Learning Theory,” Springer, New York, 1995.

[14] A. J. Smola and B. Scholkopf, “A Tutorial on Support Vector Regression,” *Statistics and Computing*, Vol. 14, No. 3, 2004, pp. 199-222. doi:10.1023/B:STCO.0000035301.49549.88

[15] B. Huang, Z. Cai, Q. Gu and C. Chen, “Using Support Vector Regression for Classification,” *Advanced Data Mining and Applications*, Vol. 5139, 2008, pp. 581-588.

[16] V. Vapnik, S. Golowich, and A. Smola, “Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing,” *Advances in Neural Information Processing Systems*, Vol. 9, 1997, pp. 281-287.

[17] E. Osuna, R. Freund and F. Girosi, “An Improved Training Algorithmfor Support Vector Machines,” *Neural Networks for Signal Processing VII. Proceedings of the* 1997 *IEEE Workshop*, Amelia Island, 24-26 September 1997, pp. 276-285. doi:10.1109/NNSP.1997.622408

[18] T. Joachims, “Making Large Scale SVM Learning Practical,” *Advances in Kernel Methods*, 1999, pp. 169-184.

[19] J. Platt, “Using Analytic QP and Sparseness to Speed Training of Support Vector Machines,” Advances in Neural Information Processing Systems, MIT Press,

Cambridge, 1999, pp. 557-563.

[20] R. Collobert and S. Bengio, "Svmtorch: Support Vector Machines Forlarge-Scale Regression Problems," *Journal of Machine Learning Research*, Vol. 1, 2001, pp. 143-160. doi:10.1162/15324430152733142

[21] R. Rifkin, "Everything Old Is New Again: A Fresh Look at Historical Approaches in Machine Learning," Ph.D. Dissertation, Massachusetts Institute of Technology, 2002.

[22] O. Mangasarian and D. Musicant, "Large Scale Kernel Regression via linear Programming," *Machine Learning*, Vol. 46, No. 1-3, 2002, pp. 255-269. doi:10.1023/A:1012422931930

[23] P. Drineas and M. W. Mahoney, "On the Nystrom Method for Approximating a Gram Matrix for Improved Kernel-Based Learning," *Journal of Machine Learning Research*, Vol. 6, 2005, pp. 2153-2175.

[24] D. Hush, P. Kelly, C. Scovel and I. Steinwart, "QP Algorithms with Guaranteed Accuracy and Run Time for Support Vector Machines," *The Journal of Machine Learning Research*, Vol. 7, 2006, p. 769.

[25] S. Sra, "Efficient Large Scale Linear Programming Support Vector Machines," *Lecture Notes in Computer Science*, Vol. 4212, 2006, pp. 767-774. doi:10.1007/11871842_78

[26] Y. Censor and S. Zenios, "Parallel Optimization: Theory, Algorithms, and Applications," Oxford University Press, Oxford, 1997.

[27] C. Hildreth, "A Quadratic Programming Procedure," *Naval Research Logistics Quarterly*, Vol. 4, No. 1, 1957, pp. 79-85. doi:10.1002/nav.3800040113

[28] Z. Lu, J. Sun and K. R. Butts, "Linear Programming Support Vector Regression with Wavelet Kernel: A New Approach to Nonlinear Dynamical Systems Identification," *Mathematics and Computers in Simulation*, Vol. 79, No. 7, 2009, pp. 2051-2063. doi:10.1016/j.matcom.2008.10.011

[29] Y. Torii and S. Abe, "Decomposition Techniques for Training Linear Programming Support Vector Machines," *Neurocomputing*, Vol. 72, No.4-6, 2009, pp. 973-984. doi:10.1016/j.neucom.2008.04.008

[30] S. S. Haykin, "Neural Networks and Learning Machine," Prentice Hall, Upper Saddle River, 2009.

[31] S. Wright, "Primal-Dual Interior-Point Methods," Society for Industrial Mathematics, 1987, p. 309. doi:10.1137/1.9781611971453

[32] L. Wang, "Support Vector Machines: Theory and Applications," Studies in Fuzziness and Soft Computing, Vol. 177, Springer-Verlag, Berlin, 2005.

[33] P. Bradley and O. Mangasarian, "Massive Data Discrimination via Linear Support Vector Machines," *Optimization Methods and Software*, Vol. 13, No. 1, 2000, pp. 1-10. doi:10.1080/10556780008805771

[34] A. Smola, B. Scholkopf and G. Ratsch, "Linear Programs for Automatic Accuracy Control in Regression," *Ninth International Conference on Artificial Neural Networks*, Edinburgh, 7-10 September 1999, pp. 575-580. doi:10.1049/cp:19991171

[35] B. Scholkopf and A. J. Smola, "Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond," The MIT Press, Cambridge, 2002.

[36] P. R. Perea, "Algorithms for Training Large-Scale Linear Programming Support Vector Regression and Classification," Ph.D. Dissertation, The University of Texas, El Paso, 2011.

[37] P. Rivas-Perea and J. Cota-Ruiz, "An Algorithm for Training a Large Scale Support Vector Machine for Regression Based on Linear Programming and Decomposition Methods," *Pattern Recognition Letters*, Vol. 34, No. 4, 2013, pp. 439-451. doi:10.1016/j.patrec.2012.10.026

[38] Y.-Z. Xu and H. Qin, "A New Optimization Method of Large-Scale svms Based on Kernel Distance Clustering," *International Conference on Computational Intelligence andSoftware Engineering*, Wuhan, 11-13 December 2009, pp. 1-4.

[39] C. Bishop, "Neural Networks for Pattern Recognition," Oxford University Press, Oxford, 1995.