Expert Systems with Applications 38 (2011) 4052–4057

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Short term wind speed prediction based on evolutionary support vector regression algorithms

Sancho Salcedo-Sanz [a,*], Emilio G. Ortiz-García [a], Ángel M. Pérez-Bellido [a], Antonio Portilla-Figueras [a], Luis Prieto [b]

[a] Department of Signal Theory and Communications, Universidad de Alcalá, Madrid, Spain
[b] Energy Resource Department, Iberdrola Renovables, Madrid, Spain

ARTICLE INFO

Keywords:
Support vector regression algorithms
Evolutionary algorithms
Hyper-parameters estimation
Short-term wind speed forecasting

ABSTRACT

Hyper-parameters estimation in regression Support Vector Machines (SVMr) is one of the main problems in the application of this type of algorithms to learning problems. This is a hot topic in which very recent approaches have shown very good results in different applications in fields such as bio-medicine, manufacturing, control, etc. Different evolutionary approaches have been tested to be hybridized with SVMr, though the most used are evolutionary approaches for continuous problems, such as evolutionary strategies or particle swarm optimization algorithms. In this paper we discuss the application of two different evolutionary computation techniques to tackle the hyper-parameters estimation problem in SVMrs. Specifically we test an Evolutionary Programming algorithm (EP) and a Particle Swarm Optimization approach (PSO). We focus the paper on the discussion of the application of the complete evolutionary-SVMr algorithm to a real problem of wind speed prediction in wind turbines of a Spanish wind farm.

1. Introduction

The Support Vector Machine (SVM) (Vapnik, 1998) is a powerful and robust methodology in statistical machine learning, that has been successfully applied to regression problems (SVMr) (Akay, 2009; Cherkassky & Ma, 2004; He, Wang, & Jiang, 2008; Lázaro, Santamaría, Pérez-Cruz, & Artés-Rodrıguez, 2005; Wu, Chau, & Li, 2008), including problems of wind speed prediction (Mohandes, Halawani, Rehman, & Hussain, 2004; Salcedo-Sanz et al., 2009). The SVMr is considered a good methodology because it allows the use of the kernel theory to increase the quality of regression models and also because, in most cases, it can be solved as a convex optimization problem. Several fast algorithms there exist to carry out the SVMr training, such as the sequential minimal optimization algorithm (Smola & Schölkopf, 1998). In spite of this, the SVMr performance heavily depends on the choice of several hyper-parameters, necessary to define the optimization problem and the final SVMr model. Unfortunately, there is not an exact method to obtain the optimal set of SVMr hyper-parameters, so that a search algorithm must be applied to obtain the best possible set of hyper-parameters which guarantees the maximum possible quality of the final model.

In general, the search algorithms used to obtain SVMr hyper-parameters can be divided in three groups. The first group of algorithms for SVMr hyper-parameters search is based on *grid searches* (Akay, 2009; Mohandes et al., 2004), where the search space of parameters is divided into groups of possible parameters to be tested, usually in an uniform fashion. The second group of search algorithms is formed by local search type approaches, such as *pattern search* proposed in Momma and Bennett (2002). Finally, the third group of algorithms applied to obtain SVMr hyper-parameters is based on metaheuristics, or global optimization algorithms, such as evolutionary computation (Hou & Li, 2009; Wang, Yang, Qin, & Gui, 2005; Wu, Tzeng, & Lin, 2009). This latter set of methodologies include several approaches, such as genetic algorithms, evolutionary algorithms, particle swarm or differential evolution, etc., which are applied to implement a robust research on the hyper-parameters search space. This work is focused on this type of methodologies, because they have shown very good performance in previous applications. For example Friedrichs and Igel (2005) was one of the first approaches where the evolutionary optimization of SVM parameters was proposed. More recently, similar approaches have also been successfully applied to different problems, as in Hou and Li (2009), where an evolutionary strategy algorithm has been applied to obtain the SVM parameters in a problem of short-term fault prediction, or in Cheng and Wu (2009), where an evolutionary SVM approach has been proposed to a problem of construction management. Novel evolutionary computation algorithms have also been tested, for example a quan-

* Corresponding author. Address: Departamento de Teoría de la Señal y Comunicaciones, Universidad de Alcalá de Henares, Campus Universitario, 28871 Alcalá de Henares, Madrid, Spain. Tel.: +34 91 885 6731; fax: +34 91 624 8749.
E-mail address: sancho.salcedo@uah.es (S. Salcedo-Sanz).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.09.067

tum-inspired evolutionary approach has been proposed in Luo et al. (2008), and a novel hybrid genetic algorithm for both selecting the optimal kernel function and optimizing its corresponding hyper-parameters has been proposed in Wu et al. (2009). A hybrid approach involving a genetic algorithm for selecting SVM hyper-parameters has been recently applied to a problem of mitochondrial toxicity prediction (Zhang et al., 2009). In Wu, Tzeng, Goo, and Fang (2007) a real-coded genetic algorithm has been successfully applied to a problem of bankruptcy prediction. Also, different approaches based on Particle Swarm Optimization (PSO) algorithms can be found in the literature for the SVM hyper-parameters tuning. In Lin, Ying, Chen, and Lee (2008) a PSO has been successfully applied to the optimization of the SVM hyper-parameters together with a feature selection problem. A similar approach also including feature selection has been presented in Huang and Dun (2008). Novel PSO approaches have been proposed and tested in specific applications within the frame of SVM hyper-parameters tuning. For example, in Wang et al. (2009), an immune PSO approach is proposed to optimize the SVM parameters in an interesting problem of forest fire prediction, and in Wu (2010) and PSO with Gaussian mutation hybridized with a SVM is applied to a problem of power load forecast. It is easy to see that the evolutionary-based optimization of SVM hyper-parameters is really a hot topic, not only because the large amount of works applying these techniques, but also because the majority of works are very recent.

In this paper we deal with evolutionary-based approaches for SVMr hyper-parameters tuning in a real problem of wind speed prediction. We compare the performance of an Evolutionary Programming algorithm (Yao, Liu, & Lin, 1999) and a PSO approach (Eberhart & Shi, 2001) for SVMr parameter tuning, and how the resulting algorithm can be integrated in a complete system for wind speed prediction in wind farms. The specific model for SVMr parameters estimation described in the paper starts from a coarse grid search to reduce the SVMr hyper-parameters search space size. Then, in a second step, the evolutionary-based algorithms are applied to refine the search, improving the performance of the SVMr. Note that the regression SVMr algorithm has been previously applied to a problem of wind speed in Mohandes et al. (2004). In that paper, the authors successfully applied the SVMr standard algorithm with grid search to obtain the best set of kernel parameters. In this paper we show that the evolutionary estimation of hyper-parameters performs better than the grid search. The complete Evolutionary SVMr algorithm has been tested integrated into the forecasting model presented in Salcedo-Sanz et al. (2009), in a wind farm located at the south of Spain.

The rest of this paper is structured as follows: next section describes the main characteristics of the regression SVMr algorithm used in this paper. Section 3 presents the SVMr hyper-parameters model considered in this paper, consisting of a first coarse grid search, that will be refined using the evolutionary computation algorithms. Section 4 presents the basic prediction system where the proposed evolutionary SVMr approach will be integrated, and shows the good performance of this model in a problem of wind speed forecast in several turbines of a Spanish wind farm. Section 5 closes the paper with some final considerations and remarks.

2. ϵ-SVMr formulation

The ϵ-SVMr method for regression (Smola, Murata, Scholkopf, & Muller, 1998) consists of, given a set of training vectors $C = \{(\mathbf{x_i}, y_i), i = 1,\ldots,l\}$, obtaining a model of the form $y(\mathbf{x}) = f(\mathbf{x}) + b = \mathbf{w}^T\phi(\mathbf{x}) + b$, to minimize the following general risk function:

$$R[f] = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}C\sum_{i=1}^{l} L(y_i, f(\mathbf{x})), \tag{1}$$

where $\mathbf{w}$ controls the smoothness of the model, $\phi(\mathbf{x})$ is a function of projection of the input space to the feature space, b is a parameter of bias, $\mathbf{x_i}$ is a feature vector of the input space with dimension N, y_i is the output value to be estimated and $L(y_i,f(\mathbf{x}))$ is the loss function selected. In this paper, we use the L1-SVMr (L1 support vector regression), characterized by an ϵ-insensitive loss function (Smola & Schölkopf, 1998)

$$L(y_i, f(\mathbf{x})) = |y_i - f(\mathbf{x_i})|_\epsilon. \tag{2}$$

In order to train this model, it is necessary to solve the following optimization problem (Smola & Schölkopf, 1998):

$$\min\left(\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*)\right) \tag{3}$$

subject to

$$y_i - \mathbf{w}^T\phi(\mathbf{x_i}) - b \leqslant \epsilon + \xi_i, \quad i = 1,\ldots,l, \tag{4}$$

$$-y_i + \mathbf{w}^T\phi(\mathbf{x_i}) + b \leqslant \epsilon + \xi_i^*, \quad i = 1,\ldots,l, \tag{5}$$

$$\xi_i, \xi_i^* \geqslant 0, \quad i = 1,\ldots,l. \tag{6}$$

The dual form of this optimization problem is usually obtained through the minimization of the Lagrange function, constructed from the objective function and the problem constraints. In this case, the dual form of the optimization problem is the following:

$$\begin{aligned} \max \quad & \left(-\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\mathbf{x_i}, \mathbf{x_j}) \right. \\ & \left. -\epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*)\right) \end{aligned} \tag{7}$$

$$\text{subject to} \quad \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \tag{8}$$

$$\alpha_i, \alpha_i^* \in [0, C]. \tag{9}$$

In addition to these constraints, the Karush–Kuhn–Tucker conditions must be fulfilled, and also the bias variable, b, must be obtained. We do not detail this process for simplicity, the interested reader can consult (Smola & Schölkopf, 1998) for reference. In the dual formulation of the problem the function $K(\mathbf{x_i},\mathbf{x_j})$ is the kernel matrix, which is formed by the evaluation of a kernel function, equivalent to the dot product $\langle\phi(\mathbf{x_i}), \phi(\mathbf{x_j})\rangle$. An usual election for this kernel function is a Gaussian function, as follows:

$$K(\mathbf{x_i}, \mathbf{x_j}) = e^{-\gamma\cdot\|\mathbf{x_i}-\mathbf{x_j}\|^2}. \tag{10}$$

The final form of function $f(\mathbf{x})$ depends on the Lagrange multipliers $\alpha_i,\ \alpha_i^*$, as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)K(\mathbf{x_i}, \mathbf{x}). \tag{11}$$

Thus, the ϵ-SVMr model depends on parameters ϵ, which controls the width of the error margin allowed (see Eqs. (4) and (5)). Parameter C, which controls the regularization of the model and parameter γ, which determines the final width of the Gaussians in the final model. Note that all these parameters have influence in the set of support vectors, and their effect within the final regression model, which also influences the accuracy and robustness of the final model.

3. Proposed model for SVMr hyper-parameters estimation

This section describes the proposal of this technical note about the estimation of SVMr hyper-parameters estimation. The idea is to start with a coarse-grain grid search of parameters, and then carry out a refinement of the search using evolutionary computation algorithms.

3.1. Initial coarse-grain search

In order to initialize an evolutionary algorithm to search for SVMr hyper-parameters C, ϵ and γ, it is necessary to set a range of possible values for these parameters. If the selected range is too large, the algorithm's convergence will decrease, since the search space is also large. In addition, in large hyper-parameters search spaces, the training time of the ϵ-SVMr considered may be quite different depending on specific values of the hyper-parameters. For example, in general, large values of C and small values of ϵ imply a larger training time, due to a large number of support vectors appear in the system. Thus, it is important to start the evolutionary algorithm's search in a good point of the search space, in which the training time of the ϵ-SVMr is reasonable. This is carried out by means of the previous coarse-grain search. For this, the implemented coarse-grain search starts from given intervals of possible values for each SVMr parameter. In this intervals (parameters ranges), the search samples the space in such a way that distances between different points tested is large (only a few points are evaluated, over the total large amount of possible points in the range). The best range of parameters located by this coarse-grain procedure is considered as the range in which the evolutionary computation techniques must perform the search refinement.

Evolutionary algorithms (EAs) (Bäck, Rudolph, & Schwefel, 1993; Fogel, 1994; Goldberg, 1989; Lee & Yao, 2004; Yao et al., 1999), are robust problem's solving techniques based on natural evolution processes. They are population-based techniques which codify a set of possible solutions to the problem, and evolve it through the application of the so called *evolutionary operators* (Goldberg, 1989). In this paper we test two different algorithms of the evolutionary computation family: Evolutionary Programming (EP) and Particle Swarm Optimization (PSO), which will be described in the following subsections.

3.2. Evolutionary programming algorithm

The Classical Evolutionary Programming algorithm (CEP) was first described in the work by Bäck et al. (1993), and analyzed later by Yao et al. (1999) and Lee and Yao (2004). It is used to optimize a given function $f(\mathbf{x})$ (the validation error in our case), i.e. obtaining $\mathbf{x}_o$ such that $f(\mathbf{x}_o) < f(\mathbf{x})$, with $\mathbf{x} \in [lim\_inf, lim\_sup]$ (note that in this case, the values of $[lim\_inf, lim\_sup]$ are set by the initial coarse grain). Then, the CEP algorithm performs as follows:

(1) Generate an initial population of μ individuals (solutions). Let t be a counter for the number of generations, set it to $t = 1$. Each individual is taken as a pair of real-valued vectors $(\mathbf{x}_i, \boldsymbol{\sigma}_i)$, $\forall i \in \{1,\ldots,\mu\}$, where $\mathbf{x}_i$'s are objective variables (C, ϵ and γ), and $\boldsymbol{\sigma}_i$'s are standard deviations for Gaussian mutations.
(2) Evaluate the fitness value for each individual $(\mathbf{x}_i, \boldsymbol{\sigma}_i)$ (using the problem's objective function, the error obtained with the SVM using the parameters $\mathbf{x_i}$, in a validation set).
(3) Each parent $(\mathbf{x}_i, \boldsymbol{\sigma}_i)$, $\{i = 1,\ldots,\mu\}$ then creates a single offspring $(\mathbf{x}'_i, \boldsymbol{\sigma}'_i)$ as follows:

$$\mathbf{x}'_i = \mathbf{x}_i + \boldsymbol{\sigma}_i \cdot \mathbf{N_1}(\mathbf{0}, \mathbf{1}), \tag{12}$$
$$\boldsymbol{\sigma}'_i = \boldsymbol{\sigma}_i \cdot \exp\left(\tau' \cdot N(0,1) + \tau \cdot \mathbf{N}(\mathbf{0}, \mathbf{1})\right), \tag{13}$$

where $N(0,1)$ denotes a normally distributed one-dimensional random number with mean zero and standard deviation one, and $\mathbf{N}(\mathbf{0},\mathbf{1})$ and $\mathbf{N_1}(\mathbf{0},\mathbf{1})$ are vectors containing random numbers of mean zero and standard deviation one, generated anew for each value of i. The parameters τ and τ' are commonly set to $\left(\sqrt{2\sqrt{n}}\right)^{-1}$ and $\left(\sqrt{2n}\right)^{-1}$, respectively (Yao et al., 1999), where n is the length of the individuals.
(4) If $x_i(j) > lim\_sup$ then $x_i(j) = lim\_sup$ and if $x_i(j) < lim\_inf$ then $x_i(j) = lim\_inf$.
(5) Calculate the fitness values associated with each offspring $(\mathbf{x}'_i, \boldsymbol{\sigma}'_i)$, $\forall i \in \{1,\ldots,\mu\}$.
(6) Conduct pairwise comparison over the union of parents and offspring: for each individual, p opponents are chosen uniformly at random from all the parents and offspring. For each comparison, if the individual's fitness is better than the opponent's, it receives a "win".
(7) Select the μ individuals out of the union of parents and offspring that have the most "wins" to be parents of the next generation.
(8) Stop if the halting criterion is satisfied, and if not, set $t = t + 1$ and go to step 3.

A second version of the algorithm is the so called Fast Evolutionary Programming (FEP). The FEP was described and compared with the CEP in Yao et al. (1999). The FEP is similar to the CEP algorithm, but it performs a mutation following a Cauchy probability density function, instead of a Gaussian based mutation. The one-dimensional Cauchy density function centered at the origin is defined by

$$f_t(x) = \frac{1}{\pi} \frac{t}{t^2 + x^2}, \tag{14}$$

where $t > 0$ is a scale parameter. See Yao et al. (1999) for further information about this topic. Using this probability density function, we obtain the FEP algorithm by substituting step 3 of the CEP, by the following equation:

$$\mathbf{x}'_i = \mathbf{x}_i + \boldsymbol{\sigma}_i \cdot \boldsymbol{\delta}, \tag{15}$$

where $\boldsymbol{\delta}$ is a Cauchy random variable vector with the scale parameter set to $t = 1$.

Finally, in Yao et al. (1999) the *Improved FEP* (IFEP) is also proposed, where the best result obtained between the Gaussian mutation and the Cauchy mutation is selected to complete the process.

3.3. Particle swarm optimization algorithm

Particle Swarm Optimization (PSO) is a population-based stochastic optimization technique developed by Eberhart and Shi (2001), inspired by social behavior of bird flocking and fish schooling. A PSO system is initialized with a population of random solutions, and searches for the optimal one by updating the population over several generations. PSO has no evolution operators, such as crossover and mutation as genetic algorithms do, but potential solutions instead, called *particles*, which fly through the problem search space to look for promising regions according to its own experiences and experiences of the whole group. Thus, social information is shared, and also individuals profit from the discoveries and previous experiences of other particles in the search. The PSO is considered a global search algorithm.

Mathematically, given a swarm of N particles, each particle $i \in \{1,2,\ldots,N\}$ is associated with a position vector $\mathbf{x}_i = (x_1^i, x_2^i, \ldots, x_K^i)$, with K the number of parameters to be optimized in the problem (note that in this case $K = 3$, and $x_1^i = C_i$ – parameter C of the ith particle, $x_2^i = \epsilon_i$ – parameter ϵ of the ith particle and finally $x_3^i = \gamma_i$ – parameter γ of the ith particle). Let $\mathbf{p}_i$ be the best previous position that particle i has ever found, i.e. $\mathbf{p}_i = (p_1^i, p_2^i, \ldots, p_K^i)$, and $\mathbf{g}$ be the group's best position ever found by the algorithm, i.e. $\mathbf{g} = (g_1, g_2, \ldots, g_K)$. At each iteration step $k + 1$, the position vector of the ith particle is updated by adding an increment vector $\Delta \mathbf{x}_i(k+1)$, called velocity $\mathbf{v}_i(k+1)$, as follows:

$$v_d^i(k+1) = v_d^i(k) + c_1 r_1 \left(p_d^i - x_d^i(k)\right) + c_2 r_2 \left(g_d - x_d^i(k)\right), \qquad (16)$$

$$v_d^i(k+1) = \frac{v_d^i(k+1) \cdot V_d^{max}}{\left|v_d^i(k+1)\right|}, \quad \text{if } \left|v_d^i(k+1)\right| > v_d^{max}, \qquad (17)$$

$$x_d^i(k+1) = x_d^i(k) + v_d^i(k+1), \qquad (18)$$

where c_1 and c_2 are two positive constants, r_1 and r_2 are two random parameters which are found uniformly within the interval $[0,1]$, and v_d^{max} is a parameter that limits the velocity of the particle in the dth coordinate direction. This iterative process will continue until a stop criterion is satisfied, and this forms the basic iterative process of a standard PSO algorithm (Eberhart & Shi, 2001).

4. Experimental part: application to a wind speed forecasting problem in Spain

This section describes a real application of the proposed technique in a Spanish wind farm. First, the forecasting model used, previously presented in Salcedo-Sanz et al. (2009), will be described, and how to include the SVM model in it. Several tests have been carried out using this forecasting model with the evolutionary SVMr algorithm, including test with different global forecasting models as initial wind predictors. The results obtained are given in Section 4.2.

4.1. Hybrid model considered for short-term wind speed prediction using evolutionary SVMrs

Wind power is nowadays one of the predominant alternative sources of energy, representing about 10% of the energy consumption in Europe, and over 15% in countries such as Spain, Germany or the USA (Mohandes et al., 2004). One of the main problems of wind power generation are the continuous fluctuations of the wind speed. This point makes very difficult to forecast the power which will be injected in the distribution network, which can cause difficulties in the energy transportation. A good forecast of the produced power is, therefore, very important for the management of wind farms. It is common that, in wind farms, the prediction of produced power is estimated from the prediction of wind speed in the turbines of the farm. This methodology allows a better management of the wind farm, so several works dealing with wind speed prediction problems in wind farms have been can be found in the literature (Alexiadis, Dokopoulos, Sahsamanoglou, & Manousaridis, 1998; Landberg, 1999; Mohandes et al., 2004; Salcedo-Sanz et al., 2009).

A system of short-term wind speed forecast based on global and mesoscale models has been recently presented in Salcedo-Sanz et al. (2009) (Fig. 1). The original system starts from a given global weather forecast model, and two different processes of down-scaling are considered, the first one a physical down-scaling using a mesoscale forecasting model, and the second one an statistical downscaling processing, using a neural network. This last process of down-scaling can be replaced by a SVMr, obtaining the system considered in this paper.

The system proposed in Salcedo-Sanz et al. (2009) uses a global prediction system (specifically the Global Forecasting System (GFS) (Kanamitsu et al., 1991)) to obtain a first prediction of meteorological variable for future times at given positions and altitudes, considering as horizontal domain the entire Earth. The global prediction system integrates the Navier–Stokes equations, to provide a set of atmospheric variables which can be useful for different applications, such as pressure (P), temperature (T), geopotential height (gph) and also wind speed and direction ($\mathbf{v}$). In general,

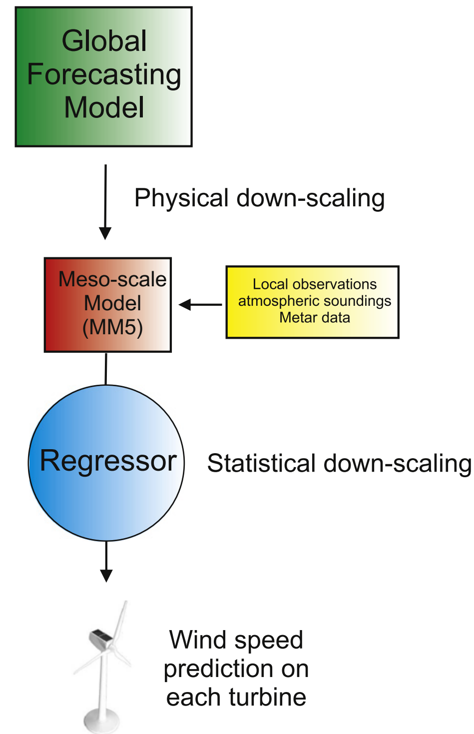

Fig. 1. Outline of the general form of a hybrid system for short-term wind speed forecasting.

these variables are solved for a given number of levels in height, which usually vary between 1000 hPa and 10 hPa. In addition, the majority of the global models also provide these basic variables at a level of 10 m over the ground. Regarding the spatial resolution of the model proposed in Salcedo-Sanz et al. (2009) was $1° \times 1°$. Note that the forecasts from global models do not include local characteristics, so a down-scaling process is carried out.

Two processes of down-scaling are considered in Salcedo-Sanz et al. (2009). First, a process of down-scaling (physical down-scaling) using the Fifth generation Mesoscale Model (MM5 model) (Dudhia, 1993). The MM5 is a *limited area* model, which solves the Navier–Stokes equations which modeled the behavior of the atmosphere (similar to the global models), but without including ocean–land interactions and other important variables of the global forecasting models. The MM5 initialization is carried out by using the data from the global prediction systems considered, and also data from atmospheric soundings, using aerostatic balloons at the Iberian Peninsula in the following sites: Gibraltar, Madrid, Murcia, Palma de Mallorca, Santander and Zaragoza. In addition, also *metar* data are included in the MM5 initialization. Metar are surface data measured in the 39 airports of the Iberian Peninsula and the Balear Islands, each 30 min. Variables included in metar data are pressure, temperature, wind speed and direction, among others. The result of this physical down-scaling carried out using the MM5 model is a forecast of the wind speed and direction in a more realistic orography than the one giving by the global forecasting models.

Finally, the data from the MM5 models are processed with a neural system in order to obtain the wind speed prediction in each turbine of the wind farm. In this case, as has been mentioned before, we substitute the neural network used in Salcedo-Sanz et al. (2009) by a Support Vector regression algorithm. The problem is, in this case, to obtain the optimal parameters of the SVM for the system to obtain the best quality wind speed prediction. For this we apply the evolutionary computation algorithms described above.

4.2. Results

Data from January to June 2006 in the wind farm "La Fuensanta", Albacete, Spain, have been used in the study presented in this paper. In this case, we consider three different global models as input data for the MM5: the GFS (Global Forecasting System, from the National Center for Environmental Prediction, USA) (Kanamitsu et al., 1991), also used in Salcedo-Sanz et al. (2009). The NOGAPS (Navy Operational Global Prediction System, USA Navy) (Toll & Clune, 1995) and finally the CMC (del Canadian Meteorological Center) (Coté et al., 1998). Note that we obtain a different wind forecast for each of the global models considered. For each of the three global forecasting model considered in this paper, the wind speed forecast in two wind turbines of the farm will be obtained using the proposed evolutionary SVMr. The solver of the well known LIBSVM library (Chang & Lin, 2001) has been used to solve the different SVMr optimization problems. The first step for the application of the evolutionary SVMr algorithm proposed consists of generating the training and test sets, using 80% of the data for the training, and the rest for testing the results obtained. In order to estimate the objective function in the evolutionary algorithms, it is necessary a measure of how good the training of the SVMr with a set of hyper-parameters is. For this, we have divided the training set in 10 subsets, and a measure of cross-validation has been obtained. The mean error in this cross-validation process will be used then to guide the evolutionary search of SVMr hyper-parameters.

Regarding the first step of coarse-grain search, the following ranges have been used as initial step of the algorithm, previously to the application of the evolutionary search: for $C[10^{-2}, 10^{0}]$ $[10^{0}10^{2}][10^{2}5\cdot10^{2}][5\cdot10^{2}10^{3}]$, for ϵ, $[10^{-4}10^{-3}][10^{3}10^{-2}][10^{-2}10^{-1}][10^{-1}10^{0}]$ and for $\gamma[10^{-3}10^{-2}][10^{-2}10^{-1}][10^{-1}10^{0}]$ $[10^{0}10^{2}]$. Starting with these ranges, we study the performance of the SVMr in the mean point of each interval, and then the evolutionary search is applied. Both algorithms (EP and PSO) have been run with the same number of function evaluations (50 generations and 20 individuals in the population), in order to make a fair comparison of their performance.

Once completed the SVMr hyper-parameters search using the coarse-grain search and the corresponding evolutionary approaches, the test error provided by the final model was analyzed and compared. The error function used to this end is the Mean Absolute Error (MAE) of the difference between the real wind speed and the predictions by our trained model. The results of MAE for the different global forecasting models used and the two evolutionary algorithms considered is displayed in Tables 1 and 2, for the two wind turbines considered, respectively (Turbines 15 and 21 of the wind farm). We can see how in the wind Turbine 15 the best results are obtained using the GFS global forecasting system, and the PSO for the calculation of the SVMr hyper-parameters. Note that the NOGAPS and CMC global models produce worse results than the GFS. In the case of Turbine 21, the results obtained with the GFS model are still better, but in this case the results using the EP are slightly better than the ones with the PSO algorithm. A further comparison can be made with the original system presented in Salcedo-Sanz et al. (2009), where a multilayer perceptron (MLP) was used as final regressor. The comparison can be only carried with the data of GFS, since it was the global system used in Salcedo-Sanz et al. (2009). For Turbine 15, the best MAE obtained using the MLP was 1.8005, and for the Turbine 21, 1.7914. Note that the proposed system using the evolutionary SVMr obtains better results in terms of MAE in these turbines, specifically, for Turbine 15, the EP-SVMr approach obtains a MAE of 1.7921 and the PSO-SVMr 1.7823, and for Turbine 21, the EP-SVMr approach obtains a MAE of 1.7847 and the PSO-SVMr, 1.7854.

Table 1
Results (MAE) in the test set using the EP-SVMr approach.

Global model	Turbine 15	Turbine 21
GFS	1.792161	1.784791
NOGAPS	1.836075	1.782673
CMC	1.928163	1.898526

Table 2
Results (MAE) in the test set using the PSO-SVMr approach.

Global model	Turbine 15	Turbine 21
GFS	1.782335	1.785467
NOGAPS	1.854671	1.788077
CMC	1.934112	1.900042

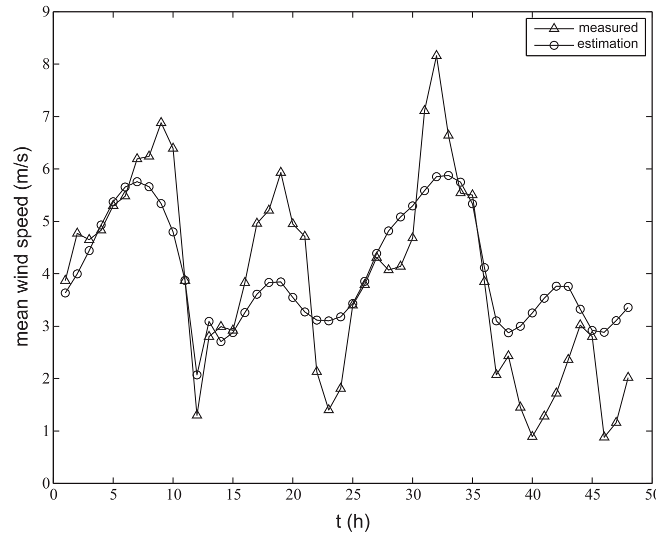

Fig. 2. Wind speed estimation using the EP-SVMr algorithm, and the real measured wind speed for the wind Turbine 21 of the considered wind farm.

Once obtained a trained SVMr system, we can apply it to the wind speed forecasting in a specific wind turbine of the farm. An example of the wind speed forecast for 48 h in the test set, and the comparison with the real wind speed measured for Turbine 21 is shown in Fig. 2. The forecasting presented has been obtained using the prediction system with the EP-SVMr approach to carry out the final down-scaling process. Note how the SVMr produces soft curves for the prediction, which follow quite well the wind speed trend in the turbine.

5. Conclusions

In this paper we have presented a model of hyper-parameters search in Support Vector Machines for regression (SVMr) based on evolutionary computation, and how to incorporate this approach to a system of wind speed prediction. Specifically, two different evolutionary algorithms, Evolutionary Programming and Particle Swarm Optimization have been considered, and their performance in the problem of SVMr parameters tuning, has been tested in the frame of the wind speed prediction problem tackled. The complete system for prediction, including the evolutionary SVMr algorithm has shown very good performance in the wind speed forecast at an Spanish wind farm, outperforming a previous model using a multi-layer perceptron as final regression algorithm.

Acknowledgements

This work has been partially supported by Universidad de Alcalá and Comunidad de Madrid with a Grant No. CCG08-UAH/AMB-3993. E.G. Ortiz-García is supported by the FPU Predoctoral Program (Spanish Ministry of Innovation and Science) Grant reference AP2008-00248. Á.M. Pérez-Bellido is supported by a doctoral fellowship by the European Social Fund and Junta de Comunidades de Castilla la Mancha, in the frame of the Operating Programme ESF 2007-2013.

References

Akay, M. F. (2009). Support vector machines combined with feature selection for breast cancer diagnosis. *Expert Systems with Applications, 36*(2), 3240–3247.

Alexiadis, M. C., Dokopoulos, P. S., Sahsamanoglou, H., & Manousaridis, I. M. (1998). Short-term forecasting of wind speed and related electrical power. *Solar Energy, 63*(1), 61–68.

Bäck, T., Rudolph, G., & Schwefel, H. P. (1993). Evolutionary programming and evolution strategies: Similarities and differences. In *Proceedings of the second annual conference on evolutionary programming* (pp. 11–22).

Chang, C.-C., & Lin, C.-J. (2001). LIBSVM: A library for support vector machines. Software available from http://www.csie.ntu.edu.tw/cjlin/libsvm.

Cheng, M.-Y., & Wu, Y.-W. (2009). Evolutionary support vector machine inference system for construction management. *Automation in Construction, 18*(5), 597–604.

Cherkassky, V., & Ma, Y. (2004). Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks, 17*(1), 113–126.

Coté, J., Gravel, S., Méthot, A., Patoine, A., Roch, M., & Staniforth, A. (1998). The operational CMC-MRB global environmental multiscale (GEM) model. Part I: Design considerations and formulation. *Monthly Weather Review, 126*(6), 1373–1395.

Dudhia, J. (1993). A nonhydrostatic version of the penn state-NCAR mesoscale model: Validation, tests and simulation of an atlantic cyclone and cold front. *Monthly Weather Review, 121*, 1493–1513.

Eberhart, R., & Shi, Y. (2001). Particle swarm optimization: Developments, applications and resources. In *Proceedings of the IEEE congress on evolutionary computation*.

Fogel, D. B. (1994). An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks, 5*(1), 3–14.

Friedrichs, F., & Igel, C. (2005). Evolutionary tuning of multiple SVM parameters. *Neurocomputing, 64*, 107–117.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-Wesley.

He, W., Wang, Z., & Jiang, H. (2008). Model optimizing and feature selecting for support vector regression in time series forecasting. *Neurocomputing, 72*(1–3), 600–611.

Hou, S., & Li, Y. (2009). Short-term fault prediction based on support vector machines with parameter optimization by evolution strategy. *Expert Systems with Applications, 36*(10), 12383–12391.

Huang, C.-L., & Dun, J.-F. (2008). A distributed PSO-SVM hybrid system with feature selection and parameter optimization. *Applied Soft Computing, 8*(4), 1381–1391.

Kanamitsu, M., Alpert, J. C., Campana, K. A., Caplan, P. M., Deaven, D. G., Iredell, M., et al. (1991). Recent changes implemented into the global forecast system at NMC. *Weather and Forecasting, 6*(3), 425–435.

Landberg, L. (1999). Short-term prediction of the power production from wind farms. *Journal of Wind Engineering and Industrial Aerodynamics, 80*, 207–220.

Lázaro, M., Santamaría, I., Pérez-Cruz, F., & Artés-Rodríguez, A. (2005). Support vector regression for the simultaneous learning of a multivariate function and its derivatives. *Neurocomputing, 69*(1–3), 42–61.

Lee, C. Y., & Yao, X. (2004). Evolutionary programming using mutations based on the levy probability distribution. *IEEE Transactions on Evolutionary Computation, 8*(1), 1–13.

Lin, S. W., Ying, K. C., Chen, S. C., & Lee, Z. J. (2008). Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications, 35*, 1817–1824.

Luo, Z., Wang, P., Li, Y., Zhang, W., Tang, W., & Xiang, M. (2008). Quantum-inspired evolutionary tuning of SVM parameters. *Progress in Natural Science, 18*(4), 475–480.

Mohandes, M. A., Halawani, T. O., Rehman, S., & Hussain, A. A. (2004). Support vector machines for wind speed prediction. *Renewable Energy, 29*(6), 939–947.

Momma, M., & Bennett, K. P. (2002). A pattern search method for model selection of support vector regression. In *Proceedings of the SIAM international conference on data mining*.

Salcedo-Sanz, S., Ortiz-García, E. G., Pérez-Bellido, A. M., Portilla-Figueras, J. A., Prieto, L., Paredes, D., et al. (2009). Performance comparison of multilayer perceptrons and support vector machines in a short-term wind speed prediction problem. *Neural Network World, 19*(1), 37–51.

Salcedo-Sanz, S., Pérez-Bellido, A. M., Ortiz-García, E. G., Portilla-Figueras, A., Prieto, L., & Paredes, D. (2009). Hybridizing the fifth generation mesoscale model with artificial neural networks for short-term wind speed prediction. *Renewable Energy, 34*, 1451–1457.

Smola, A. J., Murata, N., Scholkopf, B., & Muller, K. (1998). Asymptotically optimal choice of ϵ-loss for support vector machines. In *Proceedings of the eighth international conference on artificial neural networks, perspectives in neural computing*.

Smola, A. J., & Schölkopf, B. (1998). A tutorial on support vector regression. *Statistics and Computing, 14*(3), 199–222.

Toll, R. F., Jr., & Clune, W. M. (1995). An operational evaluation of the navy operational global atmospheric prediction system (NOGAPS): 48-hour surface pressure forecasts. *Monthly Weather Review, 113*(9), 1433–1440.

Vapnik, V. N. (1998). Statistical learning theory. In S. Haykin (Ed.), *Adaptive and learning systems for signal processing, communications and control*. John Wiley and Sons.

Wang, Y., Wang, J., Du, W., Wang, C., Liang, Y., Zhou, C., et al. (2009). Immune particle swarm optimization for support vector regression on forest fire prediction. *Lecture Notes in Computer Science, 5552*, 382–390.

Wang, X., Yang, C., Qin, B., & Gui, W. (2005). Parameter selection of support vector regression based on hybrid optimization algorithm and its application. *Journal of Control Theory and Applications, 3*(4), 371–376.

Wu, C. L., Chau, K. W., & Li, Y. S. (2008). River stage prediction based on a distributed support vector regression. *Journal of Hydrology, 358*(1–2), 96–111.

Wu, Q. (2010). Power load forecasts based on hybrid PSO with Gaussian and adaptive mutation and Wv-SVM. *Expert Systems with Applications, 37*(1), 194–201.

Wu, C.-H., Tzeng, G.-H., Goo, Y.-J., & Fang, W.-C. (2007). A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy. *Expert Systems with Applications, 32*(2), 397–408.

Wu, G. H., Tzeng, G. H., & Lin, R. H. (2009). A novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression. *Expert Systems with Applications, 36*(3), 4725–4735.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*(2), 82–102.

Zhang, H., Chen, Q.-Y., Xiang, M.-L., Ma, C.-Y., Huang, Q., & Yang, S.-Y. (2009). In silico prediction of mitochondrial toxicity by using GA-CG-SVM approach. *Toxicology in Vitro, 23*(1), 134–140.