Computers and Electronics in Agriculture 66 (2009) 46–52

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

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

Support vector machines regression and modeling of greenhouse environment

Dingcheng Wang[a,*], Maohua Wang[b], Xiaojun Qiao[c]

[a] Institute of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210044, China
[b] Key Laboratory of Modern Precision Agriculture System Integration Research, Ministry of Education, China Agricultural University, Beijing 100083, China
[c] National Engineering Research Center for Information Technology in Agriculture, Beijing 100083, China

ARTICLE INFO

Article history:
Received 5 August 2007
Received in revised form
15 November 2008
Accepted 2 December 2008

Keywords:
LSSVMR
Greenhouse
Online
Sparsity, Modeling

ABSTRACT

The greenhouse environment is an uncertain nonlinear system which classical modeling methods cannot solve. Support vector machines regression (SVMR) is well supported by mathematical theory and has a simple structure, good generalization ability, and nonlinear modeling properties. Therefore, SVMR offers a very competent method for modeling the greenhouse environment. However, to deal with uncertainty, the model must be rectified online, and Online Sparse Least-Squares Support Vector Machines Regression (OS_LSSVMR) was developed to solve this problem. OS_LSSVMR reduced the number of training samples through use of a sample dictionary, and consequently LSSVMR has sparse solutions; the training samples were added sequentially, so that OS_LSSVMR has online learning capability. A simplified greenhouse model, in which only greenhouse internal and external air temperatures were considered, was presented, after analyzing the factors in the greenhouse environment. Then the OS_LSSVMR greenhouse model was constructed using real-world data. The resulting model shows a promising performance in the greenhouse environment, with potential improvements if a more complete data setup is used.

1. Introduction

The greenhouse environment is a very complex dynamic system covered with thin and transparent materials. This system satisfies the conditions for plant growth, but creates difficulties in controlling the greenhouse environment because of time delays and intensive disturbances from the surroundings, such as global radiation, wind speed and direction, and external air temperature and humidity. In the view of classical feedback control, such a system is poorly controlled if disturbance monitors and model based feed forward control is not applied. For this reason, computer control technology for the greenhouse environment continues to require serious attention from researchers and engineers in various fields, although it has been studied since the 1980s.

The basic problem is modeling of the greenhouse environment, because many control methods, such as adaptive control, feedback control, and intelligent control, require a precise model. Therefore, many modeling methods have been proposed for this purpose, including mechanism modeling, transfer function modeling, and black-box modeling. The mechanism model provides a clear physical explanation of the greenhouse environment, such as the early static and dynamic model presented by Bot (1983) based on the principle of conservation of energy and the improved models presented by Van Henten and De Zwart (Van Henten, 1994; De Zwart, 1996). The transfer function model has a simple structure (Udink, 1985; Nielsen and Madsen, 1995) but is applicable only to linear systems. The black-box model is based on input and output data and is suitable for both linear and nonlinear modeling.

Neural networks can model nonlinear systems and have been applied to greenhouse environment modeling (Ferreira et al., 2002; Seginer, 1997; Caponetto et al., 2000; Morimoto and Hashimoto, 2000). However, the application of neural networks, based on the empirical risk minimum (ERM), is limited because of its disadvantages, including stopping at local minima, overfitting, and selection of types depending excessively on experience. SVMR, based on structural risk minimization (SRM), can model nonlinear systems without these disadvantages (Vapnik, 2000; Muller et al., 1997). However, to deal with uncertainty, the model must be rectified online, and currently SVMR does not offer this function. Compared to the existing work already reported in the literature, the contributions and novelty of this paper reside in the following three aspects: (1) novel method of modeling the greenhouse environment based on SVMR, (2) use of OS_LSSVMR(Online Sparse Least-Squares Support Vector Machines Regression) to solve the online learning problem, and (3) use of OS_LSSVMR to reduce the set of training samples using a sample dictionary (Engel et al., 2002), with the result that LSSVMR has sparse solutions and SVMR can be used to model the greenhouse environment. Different from on-line with a forgetting factor to take care of time drift (Albright et al., 1985), OS_LSSVMR need not considering time constant is simple, and nonlinear modeling.

* *Corresponding author.*
E-mail address: dcwang2005@126.com (D. Wang).

0168-1699/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.compag.2008.12.004

This paper presents the SVMR modeling method and online learning approach for the greenhouse environment and is structured as follows: Section 2 introduces the principles of SVMR and of OS_LSSVMR. Section 3 presents the principles of modeling the greenhouse environment using SVMR. Section 4 presents the experimental results from real-world data and discussion, and Section 5 presents conclusions.

2. Principles of SVMR

2.1. Support vector machines (SVM)

SVM were originally developed for pattern recognition. To find good decision rules for pattern recognition, the typically small subset of all training examples which defines the decision boundary is called the support vector. Separation of the support vector is equivalent to separation of all examples. The concept of SVM came from the principle of the optimal separating hyperplane, which could separate examples without error and maximize the distance between the closest vector and the hyperplane. The formulation of SVM is similar to a perceptron in neural networks, and the output is a linear combination of all middle nodes. Each of the middle nodes maps to a support vector. Further details about SVM and the optimal separating hyperplane can be found in the literature (Vapnik, 2000).

2.2. Support vector machines regression

The support vector method for estimating indicator functions is called SVM. Support vector machines regression (SVMR) is a generalization of support vector machines to estimate real-valued functions (Vapnik, 2000). SVMR is also called SVR to distinguish it from SVM. SVMR are gaining popularity due to many attractive features, and promising empirical performance. The formulation embodies the Structural Risk Minimization (SRM) principle, which has been shown to be superior to traditional Empirical Risk Minimization (ERM) principle employed by conventional neural networks. SRM minimizes an upper bound on the generalization error, as opposed to ERM which minimizes the error on the training data. It is this difference which equips SVMs with a greater ability to generalize (Steve, 1998).

The basic idea of SVMR is that the data vector x is mapped into a high-dimensional feature space F by a nonlinear mapping Φ, and then linear regression is performed as follows:

$$f(x) = (w \cdot \Phi(x)) + b \quad (\Phi : R^n \to F, w \in F), \tag{1}$$

where b is a threshold value. The resulting linear regression in a high-dimensional feature space corresponds to a nonlinear regression in the low-dimensional input space, so that the dot product computation of $w, \Phi(x)$, in high-dimensional space is avoided. Because Φ is a map, the value of w can be obtained from the data by minimizing the sum of the empirical risk R_{emp} and a complexity term $\|w\|^2$ that enforces flatness in feature space. That is,

$$R(w) = R_{\text{emp}} + \lambda\|w\|^2 = \sum_{i=1}^{l} e(f(x_i) - y_i) + \lambda\|w\|^2, \tag{2}$$

where l is the number of examples, λ is a regularization constant, and $e(.)$ is a cost function. According to the literature (Vapnik, 2000; Drucker et al., 1997; Vapnik et al., 1997), the cost function $e(.)$ in formulation (2) can be represented as follows:

(1) Linear ε-insensitive cost function:

$$e(f(x) - y) = \max(0, |f(x) - y| - \varepsilon) \tag{3}$$

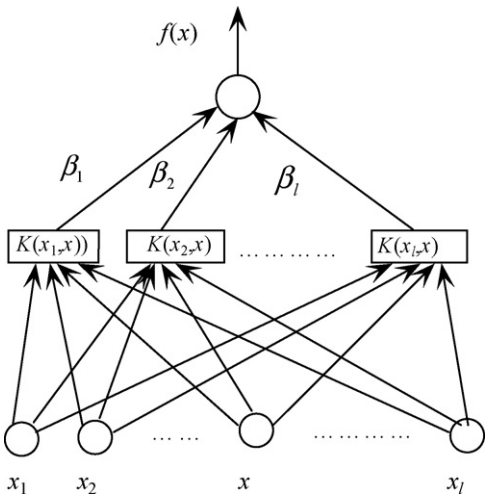

Fig. 1. The structure of SVMR.

(2) Quadratic cost function:

$$e(f(x) - y) = (f(x) - y)^2 \tag{4}$$

(3) Huber cost function:

$$e(f(x) - y) = \begin{cases} \mu\,|f(x) - y| - \dfrac{\mu 2}{2} & \text{if } |f(x) - y| > \mu \\ \dfrac{1}{2}|f(x) - y|^2 & \text{other} \end{cases} \tag{5}$$

To minimize $R(w)$, first obtain $\alpha_i - \alpha_i^*$, and then,

$$w = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*)\Phi(x_i), \tag{6}$$

where α_i, α_i^* is the solution which minimizes $R(w)$. For details, see (Steve, 1998). According to Eqs. (1) and (6), $f(x)$ can be rewritten as:

$$f(x) = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*)(\Phi(x_i).\Phi(x)) + b = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*)K(x_i, x) + b \tag{7}$$

where $K(x_i, x) = \Phi(x_i) \cdot \Phi(x)$ is called a kernel function, which is any symmetric kernel function satisfying Mercer's condition and corresponds to a dot product in some feature space (Boser et al., 1992). Therefore there are many types of kernel function, such as the polynomial kernel function $K(x_i, x) = [(x \cdot x_i) + 1]^q$, the RBF kernel function $K(x_i, x) = \exp\{-|x - x_i|^2/2\sigma^2\}$, and the sigmoid kernel function $K(x_i, x) = \tanh(\nu(x \cdot x_i) + c)$ (Vapnik, 2000; Drucker et al., 1997).

b can be computed by choosing a point on the margin using eq. (7). However, for stability purposes, it is recommended to take the average over all points on the margin,

$$b = \text{average}_k \left\{ \delta_k + y_k - \sum_{i=1}^{l} (\alpha_i - \alpha_{i_i}^*)K(x_i, x_k) \right\} \tag{8}$$

where δ_k is a prediction error for the ε-insensitive cost function, $\delta_k = \varepsilon\text{sign}(\alpha_k - \alpha_k^*)$; for the Huber cost function, $\delta_k = (1/C)(\alpha_k - \alpha_k^*)$ (Vapnik, 2000). The structure of SVMR (shown in Eq. (7)) is shown in Fig. 1.

In Fig. 1, $\beta_i = \alpha_i - \alpha_i^*, x_1, x_2, \ldots, x_3$ are $\beta_i \neq 0$ samples, are called support vectors, and responding to the dictionary samples (Section 2.4.1) selected for orthogonality for LSSVMR.