Constructive Transparent Radial Basis Function Network Learning for Non-linear Control

C. Pereira[1,2], A. Dourado[1]
[1]CISUC – Centro de Informática e Sistemas da Universidade de Coimbra
POLO II – Universidade de Coimbra, Pinhal de Marrocos, 3030 Coimbra
Phone: +351 39 790000, Fax: +351 39 701266
email: {cpereira,dourado}@dei.uc.pt
[2]ISEC – Instituto Superior de Engenharia de Coimbra
Quinta da Nora, 3030 Coimbra
Phone: +351 39 790200, Fax: +351 39 790270
email: cpereira@isec.pt

ABSTRACT: In this work a constructive radial basis function network (RBFN) learning method is applied. This approach uses the functional equivalence principles between RBFN and fuzzy systems in order to achieve a minimal structure network. Firstly, an initial network based on linguistic descriptions is constructed. Secondly, a constrained constructive adaptation law, based on a minimal resource allocating algorithm, is applied in order to on-line adjust the structure and parameters of the RBFN, keeping the transparency property and guaranteeing the linguistic interpretation. Thus, at any instant, knowledge from the network can be easily extracted, validating its structure. Experimental results on a benchmark process show the effectiveness of the presented approach.

KEYWORDS: Radial basis function, constructive learning, functional equivalence, non-linear system control.

1. INTRODUCTION

The radial basis function networks (RBFN) have become popular models for a broad variety of control problems. RBFN can be used as building blocks on adaptive controllers for unknown non-linear systems, Pereira *et al.* (1997). However, because the difficulties to extract knowledge from them in a comprehensible way, the RBFN are often considered as black-box models. Also, these neural networks (NN) adaptive controllers cannot incorporate linguistic system descriptions. On the other hand, fuzzy inference systems represent knowledge using linguistic labels and may be easily interpreted. Under some assumptions, a RBFN is functional equivalent to a fuzzy inference system, Jang and Sun (1993). Thus an obvious way to extract knowledge from it is to consider it as a fuzzy system. However, if we want to maintain, during the training, the interpretability of the NN, some restrictions must be applied. Although the view held by many NN researchers that explicit rules are not necessary, the transparency advantages are manifold. Firstly, as a means for better understanding the inner mechanisms of the network, obtaining an understandable interpretation of neural nets. Secondly, it can be a way to increase the learning speed. Thirdly, to validate the network. In this work we develop a RBFN training method in which real transparency is achieved, imposing learning constraints, limiting the number of membership functions and limiting the number of rules. The transparency property means the capacity of giving a linguistic interpretation to the NN. The following requirements should be satisfied:

(a) Minimal structure: implementation of growing and pruning techniques in order to maintain the minimum number of units at each instant.
(b) Transparency property: during the training, the functional equivalence to a fuzzy system is kept, assigning linguistic labels to each dimension of the activation function.
(c) The required knowledge about the process should be kept as little as possible.

Concerning control applications an initial reasonable structure is desired even without deep a-priori knowledge. Thus, the initial structure is based on a common sense fuzzy rule base.

The organisation of this paper is as follows. We first discuss the functional equivalence between a RBFNN and a fuzzy inference system and the restrictions that we must apply to the NN training if we want to keep transparency. Secondly the constructive training algorithm is described. Thirdly, simulations using one well-known non-linear system and experimental results on a laboratory scaled process are shown. Finally, some concluding remarks are given.

2. RESTRICTIONS FOR FUNCTIONAL EQUIVALENCE

In this section the functional equivalence between radial basis function networks and fuzzy inference systems is presented. In most generic sense, a RBFN is any network that has radial symmetric activation functions. The output of a hidden neuron is a function of the distance between an input vector and the centre of the activation function. Figure 1 shows the structure of this type of network.

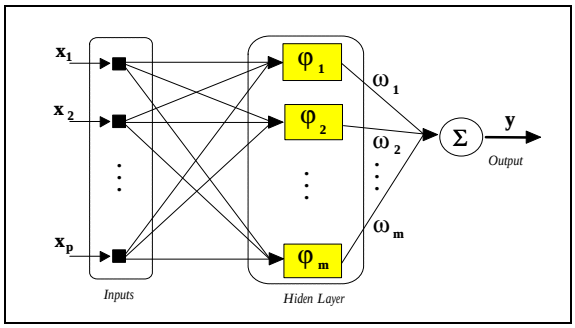


Figure1: RBFN structure.

Given an input vector of dimension *p*, $\mathbf{x} \in \Re^p$, the output of the network is described by:

$$y = f[\mathbf{x}] = \sum_{i=1}^{m} w_i j_i \left(\left\| \mathbf{x} - \mathbf{c}_i \right\| \right), \mathbf{c_i} \in \Re^p \tag{1}$$

in which w_i (i = 1,2,.., m) are the network weights, $\|.\|$ denotes de Euclidean norm, $\mathbf{c}_i$ are the basis function centres and j_i is the radial basis function. This function can be selected in one of many different ways. Typically the Gaussian exponential function is used according to (2), where s_i defines the width of the receptive field:

$$j_i : \Re^p \rightarrow \Re, \quad j_i = \exp\left(-\frac{\left\| \mathbf{x} - \mathbf{c}_i \right\|^2}{s_i^2} \right) \tag{2}$$

A fuzzy inference system is a class of expert systems that make decisions using built-in fuzzy rules. Generally, the fuzzy system maps an input fuzzy set into an output fuzzy set. A special case of fuzzy rules is the Takagi and Sugeno's type, in which the consequence part is a linear combination of inputs. The functional equivalence between an RBFN and a fuzzy inference system can be established under the following restrictions, Jang and Sun (1993):

a) The number of RBFN units is equal to the number of fuzzy if-then rules.
b) The output of each fuzzy rule is a constant (Zero-order Sugeno Fuzzy System).
c) The membership functions within each rule are chosen as Gaussian functions with the same variance.
d) The T-norm operator used to compute each rule's firing strength is multiplication.
e) Both the RBFN and the fuzzy inference system use the same method to derive their overall outputs (weighted sum is used in this case).

Additionally, it is worth noting that in order to keep the linguistic interpretation of the network, the centres must be positioned in a rectangular grid, which may be increased or reduced in size according to a constructive training procedure.

3. CONSTRUCTIVE LEARNING ALGORITHM

Several learning algorithms have been proposed to train the RBF networks, being well known the Orthogonal Least Squares, Chen *et al.* (1990), and the hybrid training proposed by Moddy and Darken (1989). In these classical approaches the number of hidden units is fixed a-priori. The disadvantage is that it may result usually in too many hidden neurons. However, how to determine this number effectively? Researchers have proposed constructive methods. The Resource Allocating Network (RAN) starts with no hidden units and allocates a new computational unit whenever an unusual pattern is presented to the network, Platt (1991). The network learns by allocating new units and adjusting the parameters of existing units. If a new pattern is far away from all the centres and the error between the network input and the desired output is significant then the data is considered to have novelty and a new unit must be added. A drawback is that once a hidden unit is created it can never be removed. Therefore, a natural improvement is to detect

and remove the hidden units contributing little to the RBFN response, Fritzke (1994). Our method is based on the Minimal Resource Allocating Network, M-RAN, Yingwei and Sundararajan (1998), that uses a pruning strategy to remove units together with an additional growth criterion compared to RAN. To ensure the transition in the number of hidden units due to growing and pruning is smooth, M-RAN augments the basic novelty of RAN with an additional condition based on the RMS value of the output error over a sliding window. The pruning strategy for removing units works as follows: for every observation (k), compute the m hidden unit outputs $\mathbf{o}_i(k)$ and find the largest hidden unit output value $o_{\max}$. Compute the normalised output values:

$$n_i(k) = \left| \frac{o_i(k)}{o_{\max}} \right| \quad (i=1,\ldots,m) \tag{3}$$

If $n_i(k) < b$ for M consecutive observations then remove the ith unit, where b is the threshold selected for this criterion.

Concerning the weights computation, due to the linearity of the error function with respect to the weights, the weight matrix can be solved such that the error is minimised in the sense of recursive least squares algorithm. Other methods such as the gradient descent, Platt (1991), or the Extended Kalman Filter (EKF), Kadirkamanhatan and Niranjan (1993), can also be used. In this work, the selective forgetting factor (SFF) is applied, Parkum *et al.* (1992). The conditions assuring that the network inputs provide persistency of excitation, with fixed centres, are given in Gorinevski (1995). However if the input is sufficiently exciting, the elements of the covariance matrix converge to zero, loosing the tracking capability if the system is time-variant. A selective forgetting algorithm allows solving this problem and guarantees that the parameter covariance matrix is bounded from above and below. Another disadvantage of the usual methods is that the forgetting is uniform in space, leading to problems when only some directions of the parameter space are excited. This is the case with RBFN due to the locality property of the Gaussian functions. One input will only activate significantly a very restrict number of neurons. The selective forgetting algorithm applied in RBF learning guarantees that the forgetting is proportional to the amount of information received in each direction.

In addition, we have limited the maximum number of neurons (rules) to 49, considering 7 common sense linguistic values: negative large, negative medium, negative small, zero, positive small, positive medium and positive large. The control structure is represented in figure 2. The process is considered an unknown form function and we approximate its partial derivatives by comparing changes with previous iterations. All the possible units are represented in figure 3.

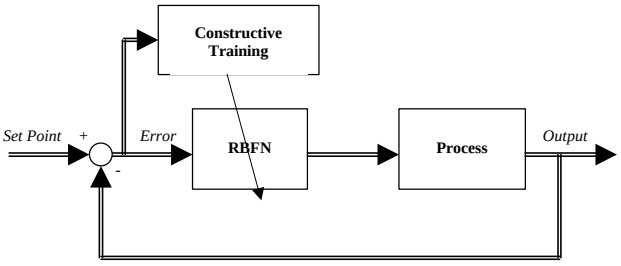


Figure 2: Control structure with RBFN in the role of a direct adaptive fuzzy controller.

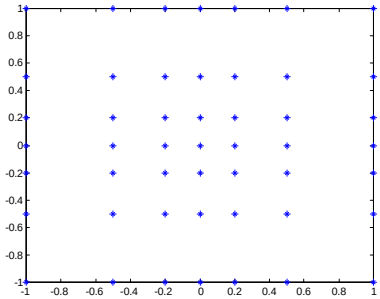


Figure 3: The 49 possible units (rules).

The error (*GE)*, change-in-error *(GDE)* and output (*GU)* scaling factors computation plays an important role in the performance of the system. In order to utilise the whole space the following simple but effective strategy is applied, Linkens and Nie (1993):

$$GE = \frac{J}{e_{\max}} \tag{4}$$

$$G\Delta E = \frac{m}{\Delta e_{\max}} \tag{5}$$

$$GU = 1 \tag{6}$$

Where J, m are the maximum elements in E and C, respectively, and $e_{\max}$ and $\Delta e_{\max}$ are the maximum measured error and change in error. The GU is simply set to 1 because no scaling operation is imposed on the learned control action. The transparent constructive algorithm starts from a simple set of rules (allowing reasonable initial controller behaviour). Then, for each sample k:

Computes the controller error $e(k) = r(k) - y(k)$ and the change-in-error $\Delta e(k)$;

Updates the scaling factors according to (4,5,6);

Computes the closest centre $\mathbf{c}_n$ to the new input pattern $\mathbf{x}(k)=(e(k)/\mathrm{GE}; \Delta e(k)/G\Delta E)$, among the 49 possible centres forming the grid (see figure 2);

If $|e(k)| > e$ and $\|\mathbf{x}(k) - \mathbf{c}_n\| > d$ and $e_{rms} = \sqrt{\sum_{i=k-(M-1)}^{k} e_i^2 / M} > e'$ then allocates a new unit $= \mathbf{c}_n$, and increases the dimensionality of SFF.

Else if a centre is not activated during *M* consecutive samples, according to (3), then removes that unit and decreases the dimensionality of SFF.

Else Adjust weights according to SFF:

Step 1: Measurement Update:

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{P}(k)\mathbf{o}(k)e(k) \quad (7)$$

where $\mathbf{P}(k)$ is the covariance matrix and $\mathbf{o}(k)$ the *m* hidden units outputs.

Step 2: Update of Covariance Matrix

$$\mathbf{P}(k+1) = \sum_{i=1}^{m} \frac{a_i(k)}{l_i(k)} n_i(k) n_i(k)^T \quad (8)$$

where α_i and ν_i are the eigenvalues and the corresponding unitary eigenvectors of $\mathbf{P}(k)$. In the conventional forgetting factor method all the λ_i are equal. In this directional forgetting, Parkum *et al.* (1992), the chosen forgetting factor can be a function of the amount of information received in the direction ν_i as an increasing function of α_i. The forgetting factor $\lambda_i(k)$ associated with the direction $\nu_i(k)$, can then be evaluated by:

$$c_i(k) = \begin{cases} 1 & if \quad a_i(k) > a_{\max} \\ a_i(k)\left[a_{\min} + a_i(k)\dfrac{a_{\max} - a_{\min}}{a_{\max}} \right]^{-1} & if \quad a_i(k) \le a_{\max} \end{cases} \quad (9)$$

4. RESULTS

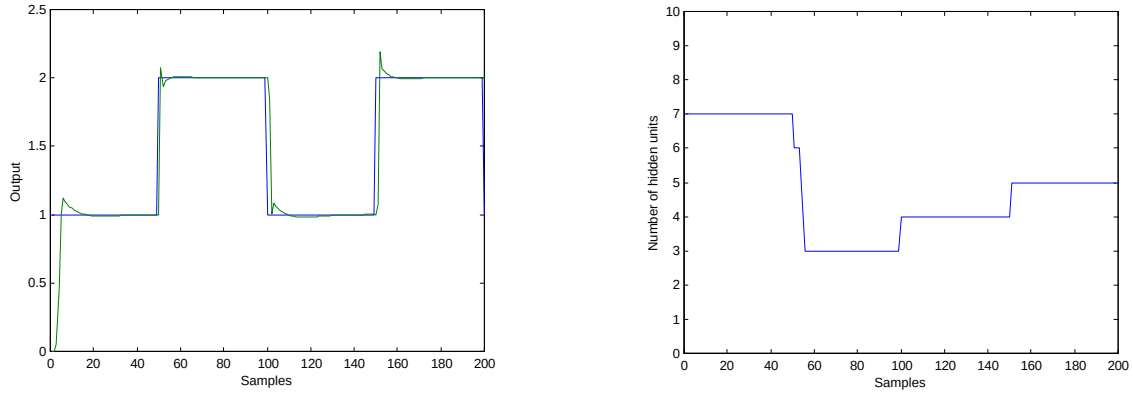
In order to test the control performances of the RBFN constructive training a simulated non-linear first-order system and an experimental benchmark process are used.

4.1 SIMULATION EXAMPLE

Consider the first-order non-linear system described by:

$$y(k) = \frac{y(k-1)}{1 + y^2(k-1)} + u^3(k-1) \quad (10)$$

The parameters for the proposed algorithm are chosen as: Initial centres: {(-1;-1),(-1;0),(-1;1),(0;0),(1;-1),(1;0),(1;1)}, *m*=7, $e = 0.1$, $d = 0.5$, $e' = 0.1$, *M*=20, *b*=50; $a_{\max} = 0.2$, $a_{\min} = 0.01$, $\mathbf{P}(0) = 0.1\mathbf{I}_m$.



(a) (b)

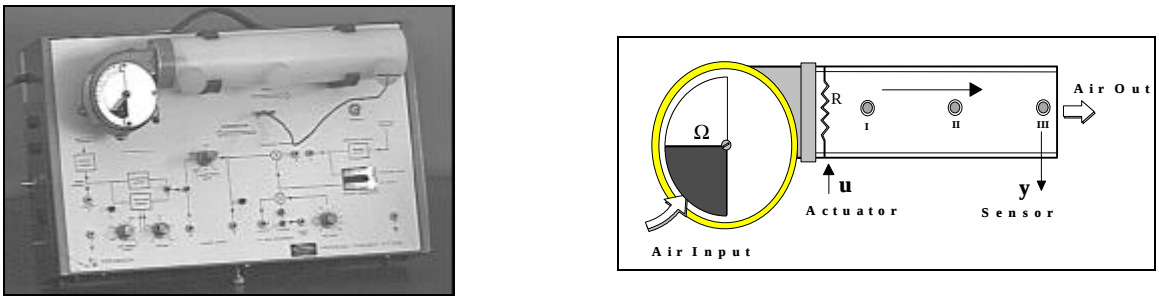
Figure 4: Set point tracking (a) Set point trajectory and output (b) Growth pattern (number of rules).

The controller performs considerably well in tracking the set point (see figure 4). The final centres positions were {(-1;0),(0;0),(1;0),(-0,5;0,2),(-0,2;1,0)}. The weight vector at instant 200 was [-0,50; -0,01; 0,50; -0,08; -0,22]. With

respect to the initial centres, four have been removed and two hidden units allocated. The goal of getting a minimal structure was fulfilled. Also, knowledge from the network can be easily extracted, validating the structure obtained. As a particular example, considering the unit (-1;0), we get: If the error, e, is negative large and the change–in-error, Δe, is zero then the first difference of the control signal, Δu =-0,5.

4.2 PT326 EXPERIMENTAL RESULTS

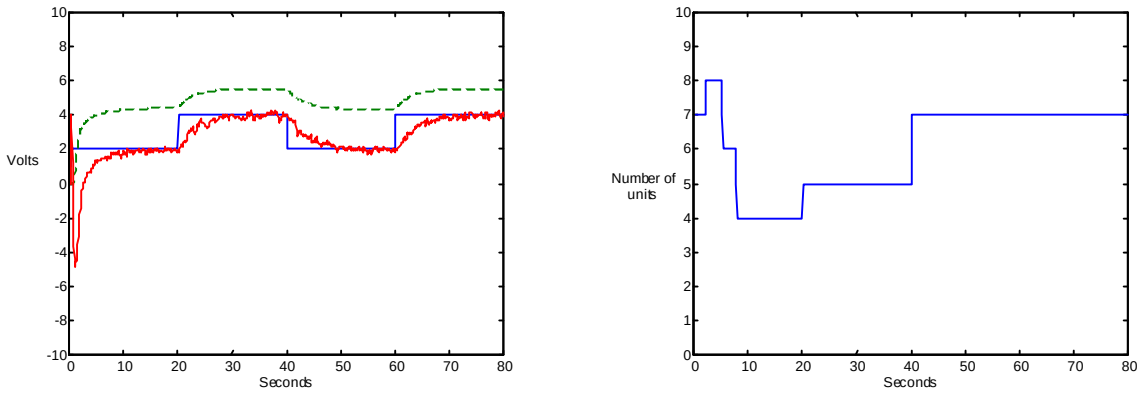
The experimental benchmark is composed by two laboratory processes, the process trainer PT326 and the process control simulator PCS327. In the PT326, depicted in figure 5, air is forced to circulate by a fan blower through a tube and heated at the inlet. There is an energised electric resistance inside the tube, and due to Joule effect, heat is released by the resistance and transferred, by convection, to the circulating air, resulting in heated air. This is a non-linear system with a pure time delay. The pure time delay depends on the position of the temperature sensor element (position I, II, and III, see figure 5b) and the damper position (Ω). The system input is the voltage applied to the power electronic circuit feeding the heating resistance, and the output is the outlet air temperature, expressed by a voltage, between -10 and 10 volts, issued from the transducer and conditioning electronics. The PCS327 is an analogue electronic apparatus allowing the introduction of additional poles of different values and additional time-delays.



(a) (b)

Figure 5: (a) experimental bench, PT326 (b) Schematic diagram.

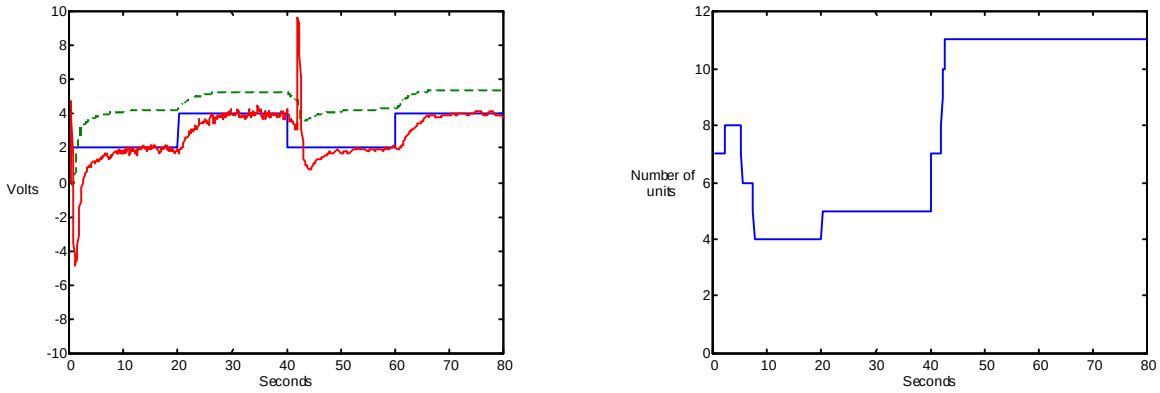
Two types of experiments were carried out, set-point tracking and systems dynamics change by the introduction of an additional pole. Each experiment was 80 seconds duration. In these experiments the air damper was positioned at Ω=20º and the sensor located at position III. The sampling time was 200 ms. The following parameter values were chosen: Initial centres {(-1;-1),(-1;0),(-1;1),(0;0),(1;-1),(1;0),(1;1)}, m=7, $e = 0.05$, $d = 0.25$, $e' = 0.1$, M=20, β=50, $a_{\max} = 0.2$, $a_{\min} = 0.01$ and $\mathbf{P}(0) = 0.1\mathbf{I}_m$. In the first experiment the tracking performance of the controller with respect to the set point changes (step input) was studied. Figures 6a and 6b show the controller behaviour. Note that the RBFN despite being put into control with just a common rule set, its initial performance is quite reasonable.



(a) (b)

Figure 6: PT326 set-point tracking (a) set-point trajectory, actuation signal '--' and output (b) Growth pattern.

The second experiment uses the PT327 for changing the process dynamics. Figure 7 illustrates the inclusion of an additional first order system (constant time equal to 1 second) at the instant 40 seconds. Six hidden units have been generated after dynamics change. The RBFN showed robustness in these situations of process order changing.



(a) (b)

Figure 7: PT326 dynamics change (a) Set-point trajectory, actuation signal '--' and output (b) Growth pattern.

5. CONCLUDING REMARKS

A constructive training method for RBF networks keeping the transparency property is investigated. It is shown that a reduced number of units (minimal structure) is obtained while the controller can exhibit reasonable performance without a-priori knowledge of the process. Finally, we notice that some issues regarding the robustness of the learning algorithm have not been addressed yet. Further research must be done to determine the controller performance effects imposed by the RBF learning restrictions, in order to keep the functional equivalence to a grid partitioning fuzzy system.

ACKNOWLEDGEMENTS

This work was supported by the Portuguese Ministry of Science and Technology (MCT), under the program PRAXIS XXI and Alice/Ollicon project.

REFERENCES

Chen, S.; Billings, S.; Grant, P., 1990, "Non-linear system identification using neural networks", Int. J. Control, vol. 51, no. 6, pp. 1191-1214.

Frietzke, 1994, "Fast Learning with Incremental RBF Networks", Neural Processing Letters, vol. 1, no. 1, pp. 2-5.

Gorinevsky, 1995, "On the Persistency of Excitation in Radial Basis Function Network Identification of Non-linear Systems", IEEE Transactions on Neural Networks, vol. 6, no. 5, pp. 1237-1244.

Jang, J.; Sun, C., 1993, "Functional equivalence between radial basis function networks and fuzzy inference systems", IEEE Transactions on Neural Networks, vol. 4, no. 1, pp. 156-159.

Kadirkamanhatan, V.; Niranjan, M., 1993, "A function estimation approach to sequential learning with neural network", Neural Computation, vol. 5, pp. 954-975.

Linkens, D.;Nie, J., 1993, "Constructing rule bases for multivariable fuzzy control by self learning", Int. J. Systems, vol. 24, no.1, 129-157.

Moody, J.; Darken, C, 1989, "Fast learning in networks of locally-tuned processing units", Neural Computation, vol. 1, no. 2, pp. 281-294.

Parkum, J.; Poulsen N., Holst J., 1992, "Recursive Forgetting Algorithms", Int. Journal of Control, vol. 55, no. 1, pp. 109-128.

Pereira, C.; Henriques, J.; Dourado, A., 1997, "Adaptive RBFNN versus Self-Tuning Control. An Experimental Comparative Study", ECC97.

Platt, J., 1991, "A Resource allocating Network for Function Interpolation", Neural Computation, vol. 3, pp. 213-225.

Yingwei, Lu; Sundararajan, N., 1998, "Performance Evaluation of a Sequential Minimal Radial Basis Function (RBF) Neural Network Learning Algorithm", IEEE Transactions on Neural Networks, vol. 9, no. 2, pp. 308-318.