Annealing Robust Radial Basis Function Networks for Modeling with Outliers

Chen-Chia Chuang*, Jin-Tsong Jeng* and Pao-Tsun Lin
* Department of Electronic Engineering
Hwa-Hsia College of Technology and Commerce
111, Hwa-Shin Street, Chung-Ho City,
Taipei Country, TAIWAN 235.
TEL: (886)-(2)-29426424-309
FAX: (886)-(2)-29426424-203
E-mail: chenchia@orion.ee.ntust.edu.tw

Abstract

In this paper, the annealing robust radial basis function networks (ARRBFNs) is proposed to improve the problems of robust RBFNs for function approximation with outliers. Firstly, the support vector regression (SVR) approach is used to obtain the initial structure of ARRBFNs. Because of the SVR approach is equivalent to solving a linear constrained quadratic programming problem under the fixed structure of SVR, the number of hidden nodes and adjustable parameters (e.g. initial structure) are easy obtained in the ARRBFNs. Secondly, we use the results of SVR as initialization of ARRBFNs. Then, the annealing robust backpropagation (ARBP) learning algorithm used as the learning algorithm of ARRBFNs and applied to adjust the parameters of ARRBFNs. The ARBP learning algorithm has been proposed to overcome the problems of initialization and cut-off points in the robust learning algorithm. Based on the initialization of ARRBFNs by SVR approach, the ARRBFNs have a fast convergence speed and robust against outliers. Simulation results are provided to show the validity and applicability of the proposed ARRBFNs.

Key words: Outliers, Annealing robust backpropagation learning algorithm, Radial basis function networks, Support vector regression.

1. Introduction

Radial basis function networks (RBFNs) are often used for modeling system due to its simplicity (i.e. only one layer of weights are required) and faster convergence [1]. In those approaches, the task is to obtain networks that can act as closely to the system to be modeled as possible. Since RBFNs approximated functions without requiring mathematical description of how the outputs functionally depend on the inputs, they are often referred to as model-free estimators [2]. The basic modeling philosophy of model-free estimators is that they build systems from input-output patterns directly, or in more abstract, they learn from examples without any knowledge of the model type. This kind of learning schemes used for neural networks can also be called data learning. Such learning schemes are to find functions that can match all training data as close as possible, no matter whether these data are trustable or not. In fact,

RBFNs with sufficiently many nodes in the hidden layer are referred to as universal approximators [3]. However, if the training data are corrupted by noise or outliers [4], those data learning schemes may not always come up with acceptable performance.

When the outliers are exists, the traditional RBFNs approaches are easily affected. Hence, the robust RBFNs approaches are proposed to overcome traditional RBFNs approaches while facing with outliers. In [5], the parameters of RBFNs (i.e. the parameters the of Gaussian kernel function and the synaptic weights) can be regarded as the initial structure of robust RBFNs that determined by singular values decomposition (SVD) method. However, the initial structure of robust RBFNs using SVD method still not obtains satisfying performance. Hence, robust learning algorithms that similar with the robust backpropagation (BP) learning algorithms [7] are applied to adjusting the parameters of RBFNs for the improving learning performance. Nevertheless, in the use of robust learning algorithms, there also exist the problems of initialization and the selection of cut-off points [8]. Moreover, the number nodes of RBFNs are pre-determined. In [6], the number of nodes and the parameters of robust RBFNs are obtained by adaptive growing methods and randomization. The adaptive growing method is only growing to a certain number beyond which a desired number cannot be reached. In this approach, it is difficult to determine when the adaptive growing methods based least square (LS) criterion can be switch to the adaptive growing methods based robust criterion.

In this paper, in order to overcome the problems of robust RBFNs approaches with outliers, a novel approach, called the Annealing Robust Radial Basis Function Networks (ARRBFNs), is proposed. In this approach, we using the support vector regression (SVR) approach [9] to obtain the initial structure of ARRBFNs (i.e. the properly number of nodes, the parameters of Gaussian function and the synaptic weights). The SVR approach with ε-insensitive function can be provides an estimated function within the ε zone that is not slightly affected by outliers. That justly provides better initialization to robust learning algorithm. Then, we use the annealing robust backpropagation (ARBP) learning algorithms to adjusting the parameters of Gaussian function and the synaptic weights [8]. Because of the ARBP learning algorithm has been proposed to overcome the problems of initialization and cut-off points selection in the robust backpropagation learning algorithms [8].

This paper is organized as follows. After this introduction section, the problems of robust RBFNs approaches are discussed in Section 2. Section 3, the ARRBFNs is proposed and discussed. In the section, SVR approach and ARBP learning algorithm are briefly descripted. The computer simulations are illustrated in Section 4. Finally, Section 5 concludes this paper.

2. The Problems of Robust RBFNs

The structure of RBFNs consists with an input layer, a hidden layer of radial basis functions and a linear output layer. The overall structure, assuming an input dimensionality p, implements a nonlinear mapping $F: R^p \to R$ expanded on a finite basis of nonlinear functions. When the radial basis functions are chosen as Gaussian functions, it can be expressed in the form

$$f(x) = \sum_{i=1}^{L} w_i G_i = \sum_{i=1}^{L} w_i \exp\left\{ - \left\| x - m_i \right\|^2 \Big/ 2\sigma_i^2 \right\}, \qquad (1)$$

where $x \in R^p$ is the input vector, w_i are the synaptic weights, m_i are the centers of Gaussian functions, σ_i are the width of Gaussian functions

and L are the number of Gaussian functions. The corresponding network structure is shown in figure 1. In general, the structure of RBFNs is often constructed by two catalogs. Firstly, the parameters of Gaussian function (i.e. the centers and width of Gaussian) and the synaptic weights are selected by the clustering algorithms [10] and random, respectively. The cross-validation or generalized cross-validation [10] or pre-determined are often used to obtaining the number nodes of RBFNs. Secondly, the structure of RBFNs are iteratively obtained by adaptive growing methods [6] or pruning methods. This process can be regarded as the initial structure of RBFNs. Then, the parameters of Gaussian function and synaptic weights are adjusted to improve approximated performance by the traditional learning algorithms.

However, most of traditional RBFNs approaches are based on the least square (LS) criterion that easily affected by outliers [5,6]. Hence, the robust RBFNs approaches are proposed to overcome the problems of traditional RBFNs approaches with outliers. Those robust RBFNs approaches are mainly focus on the using robust learning algorithms to adjust the parameters of Gaussian function and the synaptic weights. The robust learning algorithms, called the robust BP learning algorithm, adopt the concept of the M-estimators into backpropagation learning algorithms [12]. The basic idea of such algorithms is to use the loss function in the M-estimators to degrade the effects of those outliers. The cost function of a robust learning algorithm is defined as

$$E_R = \sum_{i=1}^{N} \sigma(e_i; \beta), \qquad (2)$$

where $\sigma(\cdot)$ is the so-called loss function, which is a symmetric function with a unique minimum at zero, β is the cut-off points serving as an index for discriminating against outliers, e_i is the estimated error for the i-th training pattern and N is the number of training data.

Nevertheless, the robust RBFNs approaches with robust learning algorithm could indeed improve the learning performance to some extent when training data contain outliers [5,6]. Thus, the robust RBFNs approaches are also exists some of problems. Firstly, the initial structure of robust RBFNs are very important that justly provides better initialization for robust learning algorithm. In fact, this initialization problem also occurs for nonlinear regression approaches in the statistics theory. In [5], the parameters of RBFNs are determined by singular values decomposition (SVD) method. However, this approach still not obtains the better initial structure of robust RBFNs, as outliers existed. Moreover, the number of nodes must be pre-determined. In [6], the number of nodes and the parameters of robust RBFNs are obtained by adaptive growing methods and randomization. The adaptive growing method is only growing to a certain number beyond which a desired number cannot be reached. In the growing process, the initial structure of robust RBFNs is obtained by the adaptive growing methods based LS criterion for a period of training. Then, the adaptive growing method based the robust criterion (i.e. the criterion of robust back-propagation learning algorithm) for rest period of training. The growing process is similarity to robust back-propagation learning algorithm. In this approach, the problem is difficult to determine when the adaptive growing methods based LS criterion can be switch to the adaptive growing methods based robust criterion.

Secondly, the outlier's effects also appear in traditional learning algorithm based LS criterion. Hence, various robust learning algorithms [5-7,13] have been proposed to overcome the outlier's effects

in traditional learning algorithms. Chen and Jain [7] firstly adopt the Hampel's M-estimator into the cost function to degrade the effects of outliers. Liano [13] took another new robust cost function by assuming errors belonging to the Cauchy distribution. In the use of robust learning approaches, there also exist some problems [8]. The important one is about the initialization. In those robust learning algorithms, to select a suitable initialization is extremely important. In [7], the authors suggested that their robust learning algorithm be applied after a period of training by the traditional back-propagation learning algorithm. However, this approach may have difficulty in determining when to switch from backpropagation learning algorithm to robust learning algorithm. Another problem arising in those robust approaches is regarding the selection of a parameter, the cut-off points of the M-estimator in the cost function. The cut-off points are used as a threshold for the rejection of outliers. Like in [7], the cut-off points are dynamically adjusted based on the value of the fixed percentage of the errors. Such an approach requires that the percentage of errors being considered as outliers must be defined first. Therefore, we propose a novel robust RBFNs approach to overcome the above problems.

3. The annealing robust RBFNs (ARRBFNs)

In this paper, we propose the annealing robust RBFNs (ARRBFNs) to improve robust RBFNs for modeling with outliers. In this approach, the initial structure of ARRBFNs is obtained by the SVR approaches. Then, the annealing robust back-propagation (ARBP) learning algorithm is applied to adjusting the parameters of Gaussian function and the synaptic weights.

3.1 The initial structure of ARRBFNs by SVR approach

The SVR approach is to approximate the given observations in an *m*-dimensional space by a linear function in another feature space *F*. The function in SVR is of the form

$$f(\vec{x},\vec{\theta}) = \left\langle \vec{\theta}, \Phi(\vec{x}) \right\rangle + b, \tag{3}$$

where $\langle \cdot,\cdot \rangle$ is an inner product defined on *F*, $\Phi(\cdot)$ is a nonlinear mapping function from R^m to F (i.e. $\Phi: R^m \to F$), $\vec{\theta} \in F$ is a parameter vector to be identified in the function, and b is a threshold. Suppose that those observations are generated from an unknown probabilistic distribution $G(\vec{x}, y)$. Then the solution for the problem is to find f that minimizes the following risk function [9]:

$$R[f] = \int L(y - f(\vec{x},\vec{\theta}))dG(\vec{x}, y), \tag{4}$$

where $L(y - f(\vec{x},\vec{\theta}))$ is the loss function measuring the difference between the desired y and the estimated output $f(\vec{x},\vec{\theta})$ for a given input $\vec{x}$. The loss functions are often chosen as the ε-insensitive function. The ε-insensitive function is defined as

$$L(e) = \begin{cases} 0, & for\ |e| \le \varepsilon \\ |e| - \varepsilon, & otherwise \end{cases}, \tag{5}$$

for some previously chosen nonnegative number ε. However, since $G(\vec{x}, y)$ is unknown, then $R[f]$ cannot be directly evaluated from (4). Usually, the following empirical risk function is used instead:

$$R_{emp}[f] = \frac{1}{P}\sum_{i=1}^{P} L(y_i - f(x_i;\vec{\theta})) = \frac{1}{P}\sum_{i=1}^{P} L(e_i), \tag{6}$$

where P is the number of training data. Although having the advantage of being relatively easy to compute and being uniformly consistent hypothesis classes with bounded complexity, the attempt to minimize R_{emp} may directly lead to the phenomenon of overfitting and thus, poor generalization occurs in the case of a high model capacity in *f*. To reduce the

overfitting effects, a regulation term is added into $R_{emp}[f]$, and (6) is modified as

$$R_{SV}[f] = R_{emp}[f] + C \cdot \left\|\vec{\theta}\right\|^2, \quad (7)$$

where $C>0$ is a regular constant. The regulation term in (7) controls the tradeoff between the model complexity and approximation accuracy in order to ensure good generalization performance.

It was shown that the solution of SVR approach can be expressed in terms of support vectors, $\vec{\theta} = \sum_{i=1}^{P} \beta_i \Phi(\vec{x})$ and therefore, the function f can be written as:

$$f(\vec{x}, \vec{\theta}) = \sum_{i=1}^{P} \beta_i \langle \Phi(\vec{x}_i), \Phi(\vec{x}) \rangle + b = \sum_{i=1}^{P} \beta_i K(\vec{x}_i, \vec{x}; \vec{\theta}) + b. \quad (8)$$

In the above equation, the inner product $\langle \Phi(x_i), \Phi(x) \rangle$ in the feature space is usually considered to be a kernel function $K(\vec{x}_i, \vec{x})$. The kernel function determines the smoothness properties of solutions and should reflect a prior knowledge on the data. In this paper, the Gaussian function is used as kernel function. The coefficients β_i in (8) can be solved by quadratic programming methods with suitable transformation of the above problem into constraint optimization problems and properly rearranging the equation into a matrix form. Note that only some of β_i's are not zeros and the corresponding vectors $\vec{x}_i$'s are called the support vectors (SVs). Hence, (8) can be rewritten as

$$f(\vec{x}, \vec{\theta}) = \sum_{i=1}^{SV} w_i K(\vec{x}_i, \vec{x}; \vec{\theta}) + b. \quad (9)$$

where *SV* is the number of SVs, $\vec{x}_i$ are support vectors and $w_i = \beta_i$ for some of $\beta_i \neq 0$. If the kernel function is chosen as Gaussian function, then (9) is equivalent to (1). That is, the *SV*, w_i and $\vec{\theta} \in \{m_i, \sigma_i\}$ can be represented as the number of Gaussian functions *L*, the synaptic weights and the parameters of Gaussian function, respectively.

3.2 The Learning Algorithm of ARRBFNs

In the learning algorithm of ARRBFNs, the annealing robust back-propagation (ARBP) learning algorithm is used [8]. An important feature of ARBP learning algorithms that adopt the annealing concept into the cost function of robust back-propagation learning algorithm is proposed. Based on the same idea, a cost function for ARBP learning algorithm is defined here:

$$E_{ARBP}(t) = \frac{1}{P} \sum_{j=1}^{P} \rho[e_j(t); \beta(t)], \quad (10)$$

where t is the epoch number, $e_j(t)$ is the error between the *j*-th desired output and the *j*-th output of the ARRBFNs at epoch t, $\beta(t)$ is a deterministic annealing schedule acting like the cut-off points and $\rho(\cdot)$ is a logistic loss function and defined as

$$\rho[e_j; \beta] = \frac{\beta}{2} \ln\left[1 + \left(e_j^2 \big/ \beta\right)\right]. \quad (11)$$

Based on the gradient-descent kind of learning algorithms, the synaptic weights w_i, the centers m_i and width σ_i of Gaussian function are updated as

$$\Delta w_i = -\eta \frac{\partial E_{ARBP}}{\partial w_i} = -\eta \sum_{j=1}^{P} \varphi(e_j; \beta) \frac{\partial e_j}{\partial w_i}, \quad (12)$$

$$\Delta m_i = -\eta \frac{\partial E_{ARBP}}{\partial m_i} = -\eta \sum_{j=1}^{P} \varphi(e_j; \beta) \frac{\partial e_j}{\partial m_i}, \quad (13)$$

$$\Delta \sigma_i = -\eta \frac{\partial E_{ARBP}}{\partial \sigma_i} = -\eta \sum_{j=1}^{P} \varphi(e_j; \beta) \frac{\partial e_j}{\partial \sigma_i}, \quad (14)$$

where η is a learning constant, $\varphi(e_j; \beta) = \partial \rho(e_j; \beta) / \partial e_j$ is usually called the influence function. When outliers exist, they have great impact on the approximated results. Such an impact can be understood through the analysis of the influence function. The using loss function (12) and its influence function in this papers are shown in figure 2. In the ARBP learning algorithm, the properties of annealing schedule $\beta(t)$ have (A) $\beta_{initial}, \beta(t)$ for first epoch, has a large values; (B)

$\beta(t) \to 0^+$ for $t \to \infty$; (C) $\beta(t) = k/t$ for any t epoch, where *k* is constants [8].

4. Simulation Results

In this section, simple example is tested to verify of the proposed ARRBFNs approach. The simulations were conducted in the *Matlab* environment. The support vector machine toolbox provided by the Steve Gunn and obtained through network service is used here. The root mean square error (RMSE) of the testing data is used to measure the performance of the learned networks (generalization capability). The RMSE is defined as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N}}, \quad (15)$$

where y_i is the desire value at x_i and $\hat{y}_i$ is the ARRBFNs output given x_i as its input. The learning constant η is chosen as 0.01 in the simulation. Now, the *sinc* function is considered as [9]:

$$y = \frac{\sin(x)}{x} \quad \text{with} \quad x \in [-10, 10]. \quad (16)$$

51 training data set are generated from (16) and three artificial outliers are added. After training, another 201 testing data set are used for evaluating the performance of ARRBFNs.

In the ARRBFNs, the initial structure of ARRBFNs is firstly obtained by the SVR approach. In the SVR approach, those required parameters are set as *C=3*, Gaussian kernel function with $\sigma = 3$ and $\varepsilon = 0.1, 0.15$. Two initial structures of ARRBFNs with the hidden nodes (i.e. the number of SVs) are obtained as 12 and 11 for $\varepsilon = 0.1$ and 0.15, respectively. These initial results SVR for ARRBFNs are shown in figure 3. From the figure 3, it is clear that the hidden nodes and initial structure (i.e. initial testing RMSE) of ARRBFNs are controlled by ε in the using SVR approach with ε-insensitive loss function. Based on the initial structure of ARRBFNs, the testing RMSE of ARRBFNs is 0.0683 and 0.1012 for $\varepsilon = 0.1$ and 0.15, respectively. Then, the parameters of ARRBFNs are adjusted by the ARBP learning algorithm, the number of epochs are needed as **156** and **323** under the testing RMSE < 0.01 for $\varepsilon = 0.1$ and 0.15, respectively. The final result of ARRBFNs under the testing RMSE < 0.01 is shown in the Figure 4. Besides, two errors convergence curves are shown in figure 5. From this example, the initial structure of ARRBFNs is obtained by the SVR approach that also provides a better initialization of ARBP learning algorithm. Hence, the proposed ARRBFNs have fast convergence speed.

5. Conclusions

In this paper, we propose ARRBFNs approach to improve the RBFNs for modeling with outliers. In the proposed approach, we use the SVR approach as the initial structure of ARRBFNs. Then, we apply ARBP learning algorithm to improve the performance of ARRBFNs. Based on the initial structure by SVR approach, the ARRBFNs have a fast convergence speed. Simulation results are provided to show the validity and applicability of the proposed ARRBFNs.

References

[1] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 2, pp 281-294, 1989.

[2] S. F. Su and S. R. Huang, "Analysis of Model-Free Estimators - Applications on Stock Market with the use of Technical Indices", Master thesis, NTUST, 1999.

[3] J. Park and I. W. Sandberg, "Approximation and Radial Basis Function Networks," *Neural Computation*, vol. 5, pp. 305-316,1993.

[4] D. M. Hawkins, *Identification of Outliers,*

Chapman and Hall, 1980.

[5] V. David Sanchez A., "Robustization of learning method for RBF networks," *Neurocomputing 9*, pp. 85-94, 1995.

[6] C. C. Lee, P. C. Chung, J. R. Tsai and C. I. Chang, "Robust Radial Basis Function Neural Networks," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 29, no. 6, pp 674-685, 1999.

[7] D. S. Chen and R. C. Jain, "A Robust Back Propagation Learning Algorithm for Function Approximation, " *IEEE Trans. Neural Networks*, vol. 5, no. 3, pp. 467-479, 1994.

[8] C. C. Chuang, S. F. Su and C. C. Hsiao, " The Annealing Robust Backpropagation (BP) Learning Algorithm," *IEEE Trans. Neural Networks*, vol. 11, no. 5, pp. 1067-1077, 2000.

[9] V. Vapnik, *The nature of statistical learning theory*, Springer-Verlag, 1995.

[10] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Englewood Cliffs, NJ: Prentice Hall, 1988.

[11] M. J. L. Orr, Introduction to Radial Basis Function Networks, University of Edinburgh, 1996.

[12] P. J. Rousseeuw, and M. A. Leroy, *Robust Regression and Outlier Detection*. Wiley, 1987.

[13] K. Liano, "Robust Error Measure for Supervised Neural Network Learning with Outliers," *IEEE Trans. Neural Networks*, vol. 7, no. 1, pp. 246-250, 1996.

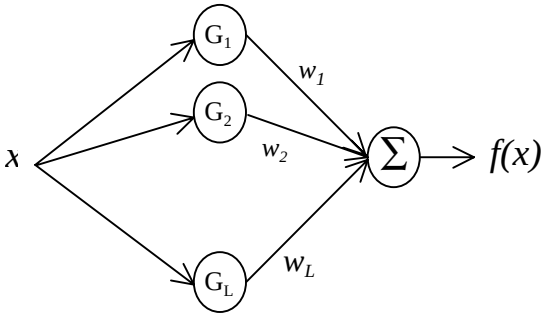

Figure 1: The structure of RBFNs is shown.

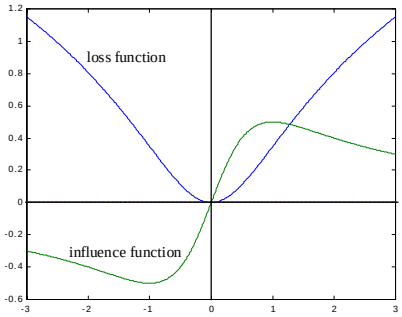

Figure 2: The logistic loss function and its influence function are shown.

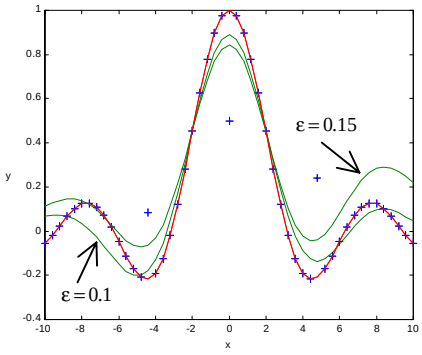

Figure 3: The training data points and two initial results of proposed ARRBFNs using SVR approach are represented as '+" and "-", respectively.

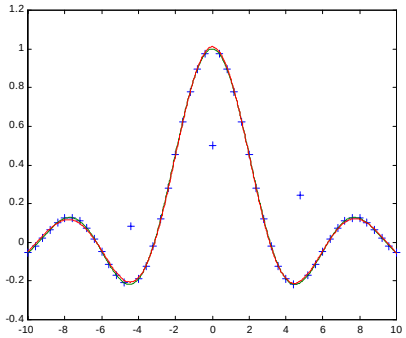

Figure 4: The final results of ARRBFNs under testing RMSE < 0.01 is shown.

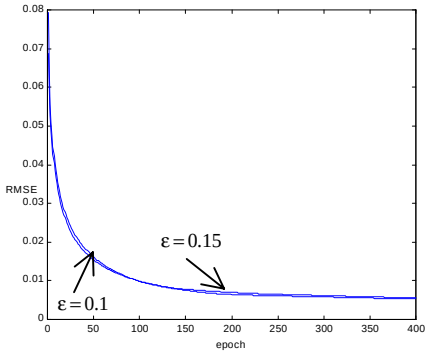

Figure 5: Error convergence curves of ARRBFNs are shown.