Far East J. Math. Sci. (FJMS) **17**(3) (2005), 329-335

A HYBRID SYSTEM OF RADIAL BASIS FUNCTION NETWORK WITH ITS APPLICATION TO RECURRENT FUNCTION MODEL

GUO SUIXUN and HUANG RONGBO

(Received March 29, 2005)

Submitted by K. K. Azad

Abstract

This paper presents a Hybrid System of Radial Basis Function (HSRBF) network and its learning algorithm. This architecture is composed of several sub-RBF networks which have their input subspace. The output of HSRBF is a linear combination of the sub-networks' outputs with the coefficients tuned together with each sub-network system parameters. We apply HSRBF to a recurrent model. The exerimental results have shown that the HSRBF outperforms the original RBF in the convergent speed and the generalization ability.

1. Introduction

The Radial Basis Function (RBF) network has been extensively applied to a lot of applications such as pattern recognition, image processing, and function approach because it allows the independent tuning of RBF network parameters and sufficiency of using one layer of neural network to establish input-output mapping [3]. When an RBF is

2000 Mathematics Subject Classification: 68T05.

Key words and phrases: HSRBF, recurrent model.

The work described in this paper was fully supported by 2005 Research Project of Guangdong Pharmaceutical University (No. 43543076).

used to map recurrent function model, the input dimension of RBF considerably increases due to its augmented input. The system structural complexity of RBF network depends on the number of hidden nodes which is further in proportion as the input dimension. In statistic learning theory, the performance of RBF is related to system structural complexity [4]. In this paper, we present a Hybrid architecture based radial basis function network which is composed of sub-RBF networks. In HSRBF each sub-RBF network has a low-dimensional input space. That is, the high-dimensional input space of original RBF is divided into low-dimensional input space as the input of sub-RBF networks in HSRBF respectively. The output of HSRBF is a linear combination of the sub-RBF networks' outputs. We give the HSRBF learning algorithm. We have applied HSRBF to recurrent function model. The experimental results have shown its outstanding performance.

This paper is organized as follows. In Section 2, we present the architecture of the HSRBF network, its learning algorithm and apply HSRBF to recurrent function model. The experimental results are provided in Section 3. Finally, we draw a conclusion in Section 4.

2. The Architecture of HSRBF and its Algorithm

The architecture of HSRBF is shown in Figure 1. The big input space V is decomposed into direct sum of q low-dimensional subspace by the input separator denoted as V_r, $r = 1, 2, ..., q$ respectively. The HSRBF consists of q sub-RBF networks denoted as $RBF_r, r = 1, 2, ..., q$. Let $d_r, r = 1, 2, ..., q$ be the dimension of rth input subspace V_r and d be the dimension of V. We let k_r, $r = 1, 2, ..., q$ be the number of rth sub-RBF network. That is, the input of rth sub-RBF network RBF_r is

$$x_t(r) = [x_t^{(i_1)}, x_t^{(i_2)}, ..., x_t^{(i_r)}] \in V_r, \tag{1}$$

where $\{i_1, i_2, ..., i_r\} \subset \{1, 2, ..., d\}$. Hereafter, we have the original input space V such that

$$V = V_1 \oplus V_2 \oplus \cdots \oplus V_q,$$

where $\oplus$ means for any $v \in V$, there exists a unique $v_i \in V_i$ such that $v = [v_1^T, v_2^T, ..., v_q^T]^T$, and d_r is the dimension of subspace V_r with

$$d = \sum_{r=1}^{q} d_r. \tag{2}$$

The actual output of HSRBF is denoted as $\hat{y}_t$ with

$$\hat{y}_t = \sum_{r=1}^{q} c_r z_t(r), \tag{3}$$

where $z_t(r)$, $r = 1, 2, ..., q$ is the RBF_r's output, and c_r is the linear combination coefficient. They are determined by learning.

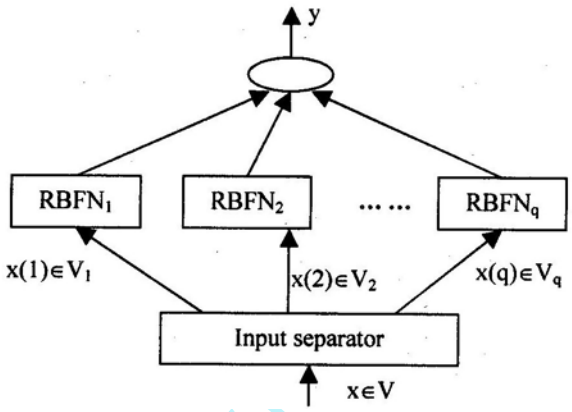

Figure 1

We give the learning algorithm of HSRBF as the extended algorithm of original RBF network. We can learn the coefficients c_r in Eq. (3) as well as the parameters of each RBF_r's by minimizing the empirical risk cost function

$$J(\Theta) = \frac{1}{N} \sum_{t=1}^{N} (y_t - \hat{y}_t)^T (y_t - \hat{y}_t), \tag{4}$$

where N is the number of inputs, $\Theta = C \cup \Theta_1 \cup \Theta_2 \cdots \cup \Theta_q$ with $C = \{c_1, c_2, ..., c_q\}$, and Θ_r being the parameters of the RBF_r. In implementation, at each step time t, we adaptively tune Θ with a little small step along

the direction of minimizing $(y_t - \hat{y}_t)^T(y_t - \hat{y}_t)$. That is, we adjust Θ by

$$c_r^{new} = c_r^{old} + \eta \hat{e}_t^T z_t(r),\ r = 1, 2, ..., q \tag{5}$$

$$\Theta_r^{new} = \Theta_r^{old} - \eta \frac{\partial J(\Theta)}{\partial \Theta_r} |_{\Theta_r^{old}}, \tag{6}$$

where η is the learning rate, and $\hat{e}_t = y_t - \hat{y}_t$.

The detailed steps in Eq. (6) depend on the implementation of each RBF_r, $r = 1, 2, ..., q$. Here we adopt the Extended Normalized RBF (ENRBF) network to implementation RBF_r. The output of RBF_r is

$$z_t(r) = \sum_{j=1}^{k_r} O_j(x_t(r))[W_j(r)x_t(r) + \beta_j(r)],$$

where $z_t(r) = [z_t^{(1)}, z_t^{(2)}, ..., z_t^{(n)}]^T$ is the output of RBF_r. Further, $W_j(r)$ is an $n \times d_r$ matrix, $\beta_j(r)$ is an $n \times 1$ vector, and $O_j(x_t(r))$ is the output of neural unit j in the hidden layer with

$$O_j(x_t(r)) = \frac{\exp\left[-0.5(x_t(r) - m_j(r))^T \sum_j^{-1} (x_t(r) - m_j(r))\right]}{\sum_{i=1}^{k_r} \exp\left[-0.5(x_t(r) - m_i(r))^T \sum_i^{-1} (x_t(r) - m_i(r))\right]},$$

where $m_j(r)$ is the rth sub-RBF network RBF_r's center vector, and $\sum_j(r)$ is the rth sub-RBF network RBF_r's receptive field of the basis function.

The two parameter sets in RBF_r should be learned from the above equation. One is $\left\{m_j(r), \sum_j(r) | j = 1, 2, ..., k_r\right\}$ in the hidden layer, and the other is $\{W_j(r), \beta_j(r) | j = 1, 2, ..., k_r\}$ in output layer.

Here, for simplicity, we prefer to learn the two parameter sets through two separate steps:

Step 1. Learn $\left\{m_j(r), \sum_j(r) | j = 1, 2, ..., k_r\right\}$, $r = 1, 2, ..., q$ in the hidden layer via k-means clustering algorithm [5].

Step 2. Learn $\{W_j(r), \beta_j(r) | j = 1, 2, ..., k_r\}$, $r = 1, 2, ..., q$ in the output layer under the least mean square criteria as well as C by minimizing Eq. (4). Consequently, the detailed implementations are given as follows:

Step 2.1. Given x_t and y_t, we calculate $\hat{y}_t$ by Eq. (3).

Step 2.2. We update

$$W_j^{new}(r) = W_j^{old}(r) + \eta \Delta W_j(r) \tag{7}$$

$$\beta_j^{new}(r) = \beta_j^{old}(r) + \eta \Delta \beta_j(r) \tag{8}$$

with

$$\Delta W_j(r) = c_r^{old} + O_j(x_t(r))\hat{e}_t x_t^T(r) \tag{9}$$

$$\Delta \beta_j(r) = c_r^{old} O_j(x_t(r))\hat{e}_t. \tag{10}$$

The iterations of Steps 2.1 and 2.2 do not stop until the parameters converge.

3. Experimental Results

To investigate the performance of the proposed architecture on recurrent function model, we generated 1100 data points $\{x_t, y_t\}$'s from the following equation

$$y_t = 0.8 \sin x_t^{(1)} + 0.4 \cos x_t^{(2)} - 0.6 x_t^{(3)} \cos x_t^{(4)} + 0.6 y_{t-1} - 0.2 y_{t-2} + e_t, \tag{11}$$

where $x_t = [x_t^{(1)}, x_t^{(2)}, x_t^{(3)}, x_t^{(4)}]$, $t \geq 2$, $y_0 = y_1 = 0$, e_t was white Gaussian noise. We let the first 1000 data points be the training set, and the remaining 100 data points be the testing set. In the experiment, the learning rate was fixed to be 0.001.

In comparison with the dual RBF model presented in [2] as well as an RBF with augmented input [1], we conducted three cases. Case 1

implemented the HSRBF with $q = 2$. Case 2 gave the dual-RBF model presented in [2], and case 3 performed the RBF with augmented input. The detailed decomposition of the six-dimensional augmented input was shown in the second column of Table 1. In these three cases, we simply set the number of hidden units in each sub-network to its input size as shown in the third column of Table 1. The fourth column shows the MSE values under the three cases, respectively. Figure 2 shows the performance curves of the HSRBF as well as the dual RBF model and RBF with augmented input. It can be seen that the HSRBF outperforms the other two in both of generalization ability and the learning speed.

Table 1. The MSE values of HSRBF, Dual-RBF and RBF with augmented input on The Testing Set

Case No.	Input Space	Hidden Units	MSE
1	$[x_t^{(1)}, x_t^{(2)}, x_t^{(4)}] \oplus [x_t^{(3)}, y_{t-2}, y_{t-1}]$	(4, 4)	0.0119
2	$[x_t^{(1)}, x_t^{(2)}, x_t^{(3)}, x_t^{(4)}] \oplus [y_{t-2}, y_{t-1}]$	(5, 3)	0.0277
3	$[x_t^{(1)}, x_t^{(2)}, x_t^{(3)}, x_t^{(4)}, y_{t-2}, y_{t-1}]$	8	0.0309

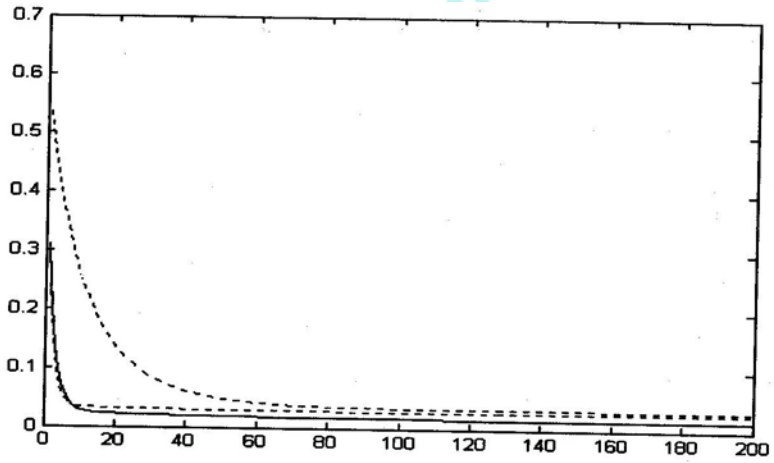

Figure 2. The learning curves of the HSRBF, Dual-RBF and RBF with augmented input respectively, where the solid line is acquired by the HSRBF, the middle dashed line is by the Dual-RBF, and the remaining dashed one is by the RBF with augmented input.

4. Conclusion

We have investigated the performance of HSRBF which is a hybrid system consisting of several sub-RBF networks. Each sub-RBF network takes an input subspace as its own input. The whole HSRBF output is the linear combination of output of sub-RBF networks. Since the high-dimensional modeling problem is divided into several low-dimensional ones, its structural complexity is generally simpler than the original RBF model. The empirical results have shown that the HSRBF reaches the better solution than the Dual-RBF and RBF with augmented input.

References

[1] S. A. Billings and C. F. Fung, Recurrent radial basis function networks for adaptive noise cancellation, Neural Networks 8(2) (1995), 273-290.

[2] Y. M. Cheung and L. Xu, A dual structure radial basis function network for recursive function estimation, Proceedings of International Conference on Neural Information Processing (ICONIP'2001) 2 (2001), 1093-1097.

[3] A. P. Topchy, O. A. Lebedko, V. V. Miagkikh and N. K. Kasabov, Adaptive training of radial basis function networks based on cooperative evolution and evolutionary programming, Proceedings of the 1997 International Conference on Neural Information Processing & Intelligent Information Systems, Springer-Verlag, Singapore, 1997, pp. 253-258.

[4] V. N. Vapnik, Statistical Learning Theory, John Wiley & Sons, Inc., New York, 1998.

[5] L. Xu, Rival penalized competitive learning, finite mixture, and multisets clustering, Proc. International Joint Conference on Neural Networks II (1998), 2525-2530.

Department of Computer Science
Guangdong Pharmaceutical University
Guangzhou, China
e-mail: xiaoxiaoguo@tom.com

Department of Mathematics
Guangdong Pharmaceutical University
Guangzhou, China
e-mail: hrongbo@tom.com