文章编号:1000 - 582X(2002)10 - 0056 - 05

一种新型径向基函数神经网络学习算法*

——递归正交最小二乘法(ROLS)

张 兴 兰,曹 长 修,梅　彬
(重庆大学 自动化学院, 重庆 400044)

摘　要:径向基函数神经网络在很多领域都得到了成功的应用。但迄今为止仍没有一种有效的方法来确定隐层中心数目。笔者将递归正交最小二乘(ROLS)方法引入 RBFNN 建模训练,利用 ROLS 算法训练网络后所得的有用信息,采用后向选择算法,逐步去掉那些使网络残差增加最小的中心,在得到网络有效中心的同时,还满足了精度要求,从而大大简化了 RBF 网络结构,节约了大量的存储空间以及计算量。仿真和实验结果表明该方法是有效而实用的。

关键词:RBF 神经网络;递归正交最小二乘;后向选择;简化
中图分类号:TP183　　**文献标识码**:A

径向基函数神经网络[1](RBFNN)已经成功地在非线性函数逼近[2]和数据分类等方面得到广泛应用,如信号处理、系统建模、控制及故障诊断等领域。标准的 RBF 网络是一个两层结构的前馈网络:非线性隐含层及线性输出层。RBFNN 的训练也可分为两个阶段来进行:首先利用无监督学习方法(如聚类算法)选择网络中心,这些中心在某种程度上反映了输入数据在训练集空间的分布;其次根据 RBF 网络的结构特点,采用最小二乘算法来确定隐含层到输出层之间的权值。这与梯度下降法[3]、共轭梯度法等非线性优化算法相比训练时间短、收敛速度快。如果训练后的网络具有一定数目且质量好的中心,通常网络性能良好。然而网络本身应该具备的中心数目事先是很难确定的,而按照前面所讲的聚类方法选择的中心数目常常比网络本身所具有的中心数目多。这样往往会导致网络复杂性和运算量的增加以及预报时矩阵数值病态[4]情况的发生。

针对这一问题,笔者采用递归正交最小二乘算法(ROLS)训练 RBF 神经网络。利用训练后正交矩阵中的信息,采用后向选择方法,逐步去掉那些使网络残差增加最小的中心,在满足要求精度的同时,使网络结构得到简化。

1　RBF 神经网络的基本思想

标准 RBF 网络表征的是 $x \in R^n \to \hat{y} \in R^p$ 的非线性映射及线性输出的变换过程

$$\hat{y}^T(t) = \Phi^T[x(t)]W \qquad (1)$$

这里,$x(t)$ 及 $\hat{y}(t)$ 分别是在 t 时刻网络的输入和输出向量。W 是权值矩阵,其元素 w_{ij} 为连接隐含层第 i 个单元响应到第 j 个输出的权值;$\Phi(t) \in R^n h$ 为隐含层非线性基函数的输出矩阵向量,第 i 单元的径向基函数可表示为 $\phi_i(x(t))$,其数学形式为 $\phi_i(x(t)) = \phi(\| x(t) - c_i \|)$,$\| \cdot \|$ 为欧氏范数,ϕ 为高斯函数或其他关于中心点 c_i 径向对称的函数。笔者采用 k_ 均值聚类法[5] 来确定网络的初始中心。通过计算权值矩阵使 $\| y(t) - \hat{y}(t) \|$ 最小,$y(t)$ 为目标输出。

2　递归正交最小二乘法

递归正交最小二乘法是建立在正交最小二乘法基础之上的,因此,首先介绍利用正交最小二乘法训练 RBF 神经网络的基本思想。

2.1　正交最小二乘法(OLS)

正交最小二乘法(OLS) 来源于线性回归模型[6]。RBF 网络从隐含层到输出层正好是一个线性变换过

* 收稿日期:2002 - 06 - 17
基金项目:国家教育部博士点基金资助项目(98061117),重庆市基础研究资助项目。
作者简介:张兴兰(1972 -),女,重庆江津人,重庆工学院助教,重庆大学硕士。

程，因此可以采用正交最小二乘法。由于多输入多输出(MIMO)的形式也适用于单输出的形式，因此本文只讨论 MIMO 的情形。根据线性回归模型，径向基函数网络的期望输出响应可表示为：

$$Y = \hat{Y} + E = \Phi W + E \tag{2}$$

$Y \in R^{N\times P}$ 为期望输出矩阵，$\hat{Y} \in R^{N\times P}$ 为神经网络模型的输出矩阵，$\Phi \in R^{N\times n_h}$ 为隐含层输出矩阵(也称回归矩阵)，$E \in R^{N\times P}$ 为输出误差矩阵，且有

$$Y^T = [y(1),\cdots,y(N)],$$
$$\hat{Y}^T = [\hat{y}(1),\cdots,\hat{y}(N)],$$
$$\Phi^T = [\phi(1),\cdots,\phi(N)],$$
$$E^T = [e(1),\cdots,e(N)]。$$

现在 MIMO 最小二乘问题可以被视为求权值矩阵 W 的问题，即将如下价值函数极小化：

$$J(W) = \| E \|_F = \| Y - \Phi W \|_F \tag{3}$$

这里 $\| * \|_F$ 为 F_ 范数，其数学形式为

$$\| A \|_F^2 = \mathrm{trace}(A^T A)$$

如果每个基函数中心都各不相同，则 Φ 为满秩[1]，且可以正交分解成如下形式：

$$\Phi = Q\begin{bmatrix} R \\ 0 \end{bmatrix} \tag{4}$$

这里 Q 为 $N \times N$ 阶正交矩阵，满足 $Q^T Q = QQ^T = I$，R 为与 Φ 秩相同的 $n_h \times n_h$ 阶上三角阵，由于矩阵的 F_ 范数是通过正交变换而来，则(3)等价于

$$J(W) = \| Q^T Y - Q^T \Phi W \|_F \tag{5}$$

令 Q^T　　$Y = \begin{bmatrix} \vec{Y} \\ \tilde{Y} \end{bmatrix}$

其中 $\vec{Y}$ 为 $n_h \times p$ 维矩阵，$\tilde{Y}$ 为 $(N - n_h) \times p$ 维矩阵，将上式代入(3)式，有

$$J(W) = \left\| \begin{bmatrix} \vec{Y} \\ \tilde{Y} \end{bmatrix} - \begin{bmatrix} R \\ 0 \end{bmatrix} W \right\|_F = \left\| \begin{bmatrix} \vec{Y} - RW \\ \tilde{Y} \end{bmatrix} \right\|_F \tag{6}$$

使价值函数极小的最优权值矩阵 W 可由下式解出

$$RW = \vec{Y} \tag{7}$$

残差为 $\| \tilde{Y} \|_F$。

文[4]采用了上述 OLS 法训练 RBF 网络，并且利用训练后的回归矩阵信息选择 RBF 网络中心，达到了减少有效中心的目的。但当 N 很大时，(4)式中要求的正交分解计算量和存储空间将会很大，也不能从根本上避免数值病态的发生。

2.2　递归正交最小二乘算法(ROLS)

为了得到递归最小二乘算法，须极小化

$$J(t) = \| E(t) \|_F = \left\| \begin{bmatrix} Y(t-1) \\ y^T(t) \end{bmatrix} - \begin{bmatrix} \Phi(t-1) \\ \phi^T(t) \end{bmatrix} W(t) \right\|_F \tag{8}$$

由于矩阵 Φ 和 Y 的维数随着新数据的增加而不断增大，直接计算 Φ 和 Y 势必会越来越困难。但是由于(8)式中的 R 及 $\vec{Y}$ 的维数小且固定，因而对 Φ 和 Y 变换后再作计算就比较容易。按上述过程对以下两式实施正交分解

$$\Phi(t-1) = Q(t-1)\begin{bmatrix} R(t-1) \\ 0 \end{bmatrix}$$
$$Q^T(t-1)Y(t-1) = \begin{bmatrix} \vec{Y}(t-1) \\ \tilde{Y}(t-1) \end{bmatrix} \tag{9}$$

从而等式(9)变为

$$J(t) = \left\| \left\{ Q(t-1)\begin{bmatrix} \vec{Y}(t-1) \\ \tilde{Y}(t-1) \end{bmatrix} \right\} - \left\{ Q(t-1)\begin{bmatrix} R(t-1) \\ 0 \end{bmatrix} \right\} W(t) \right\|_F =$$
$$\left\| \begin{bmatrix} \vec{Y}(t-1) \\ y^T(t) \\ \tilde{Y}(t-1) \end{bmatrix} - \begin{bmatrix} R(t-1) \\ \phi^T(t) \end{bmatrix} W(t) \right\|_F \tag{10}$$

为了寻求 $R(t-1)$ 及 $\vec{Y}(t-1)$ 的迭代式，计算如下的正交分解

$$\begin{bmatrix} R(t-1) \\ \cdots \\ \phi^T(t) \end{bmatrix} = Q_1(t)\begin{bmatrix} R(t) \\ \cdots \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} \vec{Y}(t) \\ \cdots \\ \tilde{y}^T(t) \end{bmatrix} = Q_1^T(t)\begin{bmatrix} \vec{Y}(t-1) \\ \cdots \\ y^T(t) \end{bmatrix} \tag{11}$$

代入(10)式，则价值函数的最终形式为

$$J(t) = \left\| \left\{ Q_1(t)\begin{bmatrix} \vec{Y}(t) \\ \tilde{y}^T(t) \end{bmatrix} \right\} - \left\{ Q_1(t)\begin{bmatrix} R(t) \\ 0 \end{bmatrix} \right\} W(t) \right\|_F =$$
$$\left\| \begin{bmatrix} \vec{Y}(t) - R(t)W(t) \\ \tilde{Y}^T(t) \\ \tilde{Y}(t-1) \end{bmatrix} \right\|_F \tag{12}$$

因此,(12) 式中的最优权值矩阵可以由下式求出

$$R(t)W(t) = \vec{Y}(t) \tag{13}$$

第 t 代的残差可由如下的递归式计算

$$\| \tilde{Y}(t) \|_F^2 = \left\| \begin{bmatrix} \tilde{y}^T(t) \\ \tilde{Y}(t-1) \end{bmatrix} \right\|_F^2 =$$

$$\| \tilde{y}^T(t) \|_F^2 + \| \tilde{Y}(t-1) \|_F^2 \tag{14}$$

$\| \tilde{y}^T(t) \|_F^2$ 是当前代第 t 代的残差增量的平方。考虑到 $R(t)$ 为上三角阵,因此权值矩阵可很容易地由(13)式后向迭代计算得出。

因此,ROLS算法的基本步骤为在第 t 代,利用 QR 分解计算出 $R(t)$,根据(12)式计算出 $\vec{Y}(t)$,再由(13)式求出权值 $W(t)$,最后由(14)式计算出残差 $\| \tilde{y}^T(t) \|_F^2$。初始化 R,$\vec{Y}$ 值,则残差可以赋值为 $R(0) = \alpha I$,α 是一个很小的正数,如 0.01,$\vec{Y}(0) = 0$ 以及 $\| \tilde{Y}(0) \|_F^2 = 0$。在离线训练网络过程中权值并不影响递归算法迭代,因而只需在网络训练的最后一代进行权值的计算存储即可,这样计算量大大减小。

3 后向选择中心算法

利用ROLS算法训练网络后,在 $t = N$ 时的最终三角方程组(13)中包含着与学习网络有关的重要信息。从这些信息可以导出一种选择网络中心的方法:后向选择法。

3.1 后向选择法

在第 1 节中初选中心时,采用 k_- 均值聚类算法所获得的中心数目往往会超过网络本身所具有的中心数目[7]。后向选择算法的基本原理是相继从网络中一个一个地去掉某个特定中心,使得网络残差增加最小,从而达到简化网络的目的。因删除中心 j 而产生的针对整个训练数据集的残差增量,可以这样计算出:设中心 j 被删除,相应地,矩阵 Φ 的第 j 列 ϕ_j、R 的第 j 列 r_j 也分别从 Φ 和 R 中删除。

结果矩阵变为

$$R' = [r_1, \cdots, r_{j-1}, r_{j+1}, \cdots, r_{n_h}] \tag{15}$$

为保持一致,权值矩阵也相应变为

$$W'^T = [w_1, \cdots, w_{j-1}, w_{j+1}, \cdots, w_{n_h}] \tag{16}$$

此时 w_i^T 表示 w 的第 i 行。

通过以上删除操作,省去下标 N,价值函数(12)的简写式为

$$J_j^2 = \left\| \begin{bmatrix} \hat{Y} - R'W' \\ \tilde{Y} \end{bmatrix} \right\|_F^2 =$$

$$\left\| \begin{bmatrix} \hat{Y}_j - r_j w_j^T \\ \tilde{Y} \end{bmatrix} \right\|_F^2 =$$

$$\| \hat{Y}_j - r_j w_j^T \|_F^2 + \| \tilde{Y} \|_F^2 \tag{17}$$

此时式(13)中的最优权值矩阵已不再最优,取而代之的是矩阵 W_j,可由化简式(17)推导得出。推导过程如下:首先对三角方程组作进一步正交变换:

$$R' = Q' \begin{bmatrix} R_j \\ \cdots \\ 0 \end{bmatrix}, \begin{bmatrix} \hat{Y}_j \\ \cdots \\ \tilde{y}_j^T \end{bmatrix} = Q'^T \hat{Y} \tag{18}$$

在删除 r_j 列后,R' 的维数降为 $(n_h - 1) \times (n_h - 1)$。

将(18)式代入(17)式,价值函数变为

$$J_j^2 = \left\| \begin{bmatrix} Q' \begin{bmatrix} \hat{Y}_j \\ \tilde{y}_j^T \end{bmatrix} \hat{Y} - Q' \begin{bmatrix} R_j \\ 0 \end{bmatrix} W_j \\ \tilde{Y} \end{bmatrix} \right\|_F^2 =$$

$$\| \hat{Y}_j - R_j W_j \|_F^2 + \| \tilde{y}_j^T \|_F^2 + \| \tilde{Y} \|_F^2 \tag{19}$$

等式(19)表明删除第 j 个中心后的最优权值矩阵 W_j 可由下式求出

$$R_j W_j = \hat{Y}_j \tag{20}$$

残差为

$$\| \tilde{Y}_j \|_F^2 = \| \tilde{y}_j^T \|_F^2 + \| \tilde{Y} \|_F^2 \tag{21}$$

比较(21)及(17)式可以看出,在删除第 j 个中心后,整个训练数据集残差平方的增量为 $\| \tilde{y}_j^T \|_F^2$。采用后向选择方法删除中心后的简化网络无须再用 ROLS 方法训练[7],由(20)式解出的 W_j 即为最终模型的最优权值。

3.2 最终预报误差准则(FPE)

以上内容介绍了通过逐步删除中心来简化训练后 RBF 网络结构的基本思想,以达到使网络结构简化且精度满足要求的目的。因而,必须有一个准则来确定删除哪一个中心以及何时停止中心的选择。事实上,每删除一个中心,网络的复杂度降低,同时精度也逐步下降。也就是说,网络结构的简化是以网络精度下降为代价的。因此,停止选择中心的方法是在网络的复杂度与精度之间寻求折中,既要使网络结构简化,又要确保网络具有所要求的精度。由于 RBF 网络的输出层是线性的,因而可以采用一种称为"最终预报误差准则[8]"(FPE)的方法。

$$E_{FPE} = \frac{1 + \beta(n_p/N)}{1 - \beta(n_p/N)} V \tag{22}$$

式中损失函数 $V = \| \tilde{Y}(N) \|_F^2 / (N)$,$n_p$ 是网络输出层权值的个数,与网络中心数目成正比($n_p \propto n_h$)。随着中心的逐个删除,n_h 逐步减小,n_p 也逐步减小。β 为加权因子,文[7]证明当 $\beta = 2$ 时,利用 FPE 准则选择 RBF 网络模型中心,将得到很高的最终预报精度。V 的

初值很容易由递推等式(14)式计算得到。E_{FPE} 是损失函数的加权形式，用来惩罚考虑网络参数 n_p 所得的复杂网络模型精度。本文中也引用了这一结论。

一般情况，对与训练数据集不太匹配(精度低)、结构过于简化的小网络以及与训练数据集匹配(精度高)但结构过于复杂的大网络来说，E_{FPE} 值会很大。在这二者之间，E_{FPE} 有一个交点，这一点表征了在网络复杂度和精度之间达到了平衡。下面将 FPE 准则用于后向选择中心。随着前述讨论的中心的逐个删除，E_{FPE} 值也会从一个初值逐渐减小到这个平衡点，在这一点中心的选择计算停止。

后向中心选择算法可归结为如下几个步骤：

1) 利用训练后网络的信息初始化 V 及 E_{FPE} 值

2) 利用(18)和(21)式计算出可能删除的每一个网络中心的损失函数 $V_j, j = 1, \cdots, n_h$。

3) $m = \text{argmin}(V_j)$，计算使损失函数增量最小的 E_{FPE}，即 E_{FPE_m}。

4) 如果 $E_{FPE_m} < E_{FPE}$，则从网络中删掉中心 m，且设 $R = R_m, \hat{Y} = \hat{Y}_m, V = V_m, n_h = n_h - 1, E_{FPE} = E_{FPE_m}$。转 2) 步。

5) 否则，停止中心选择。根据(20)式计算最终网络的最优权值。

4　仿真实验

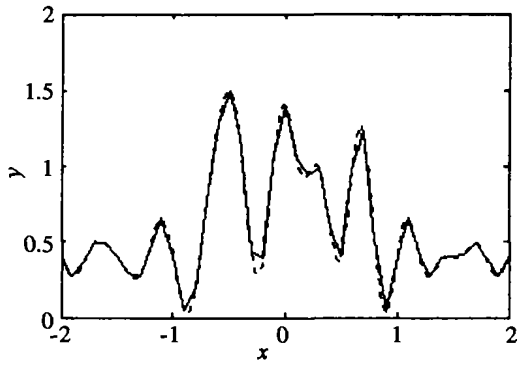
用径向基函数做如下一元函数的拟合

$$y = 0.4 + \text{sinc}(4x) + 1.1\text{sinc}(4x + 2) + 0.8\text{sinc}(6x - 2) + 0.7\text{sinc}(6x - 4) \quad (23)$$

式中

$$\text{sinc}(x) = \begin{cases} 1 & x = 0 \\ \dfrac{\sin(\pi x)}{\pi x} & x \neq 0 \end{cases} \quad (24)$$

(24)式是一个典型的多峰函数，如果样本点太少，势必会影响模型精度。本例取步长为 0.01，训练样本 401 个，采用 $k\_$ 均值聚类法，中心为 31 个，用 ROLS 法进行训练后，选择中心为 12 个，平均输出误差为 2.017%，训练时间也明显缩短。如图 1 所示。



---- 真实输出；—— 模型输出

图 1　RBF 神经网络拟合结果

5　实例计算

某化工厂取工业用的甲醇、乙醇、丙烷以及碘和丙烷混合物这四类化学药品中的一种，分别同 NaCl、KCl、LiCl、KNO_3、$Ca(NO_3)_2$、NaBr 等化学药品混合再兑入一定量的水做化学反应实验，根据其反应压力、浓度、临界温度以及各成分比重等数据，测得反应后溶液的最终温度以及比值 y1/s1(即 ylds1)。共计 1 500 组数据，中抽取 1 100 组数据进行 RBF 网络训练，其余数据用作测试。本实验是一个 MIMO 过程，总共有 12 个输入，2 个输出。下面是分别采用 k_ 均值算法和 ROLS 算法训练 RBF 网络的结果比较(见表 1、2、3)：

表 1　RBF 网络的部分测试集数据

	测试数据												
	输入											输出	
	p	x_3	x_1	ls	pr	tc	RA	RS	mul	epsionl	solubitity	omigal	tk	ylds1
①	21.47	0.048	0.168	2.453	0.00265	512.58	9.5	18.1	0.544	32.70	0.270	0.559	318.2	0.682
②	50.66	0.041	0.799	3.114	0.00794	516.25	9.9	30.0	1.078	24.55	0.782	0.635	336.0	0.906
③	32.66	0.006	0.543	0.251	0.00404	512.58	9.5	18.1	0.544	32.70	0.270	0.559	318.2	0.844
④	100.0	0.012	0.555	0.360	0.01568	516.25	9.5	30.0	1.078	24.55	0.549	0.635	352.2	0.702
⑤	7.77	0.047	0.859	1.133	0.00122	516.25	13.3	45.0	1.078	24.55	0.778	0.635	298.1	0.933
⑥	37.59	0.020	0.712	0.717	0.00464	512.58	9.5	18.1	0.544	32.70	0.270	0.559	318.2	0.914
⑦	7.48	0.016	0.909	0.364	0.00117	516.25	9.5	21.6	1.078	24.55	0.720	0.635	298.2	0.937
⑧	101.3	0.077	0.363	2.417	0.0196	536.71	13.3	19.6	2.237	20.33	0.461	0.624	362.2	0.592
⑨	46.84	0.006	0.099	0.263	0.00983	508.31	6.8	18.1	1.765	18.30	0.496	0.666	338.1	0.502
					.....		.....						...	...

经过训练后的 RBF 网络采用测试集数据进和测试，以下是两种方法的测试结果：

表 2 两种方法的比较

实验输出		采用 k_mean 算法				采用 ROLS 算法			
		测试结果		误差 /%		测试结果		误差 /%	
tk	y1ds1	tk	61ds1	Δt/tk	Δy/y1ds1	tk	y1ds1	Δt/tk	Δy/y1ds1
①318.2	0.682	310.3	0.725	2.483	6.305	312.4	0.701	1.823	2.786
②336.0	0.906	330	0.972	1.786	7.285	332.5	0.932	1.042	2.870
③318.2	0.844	313.1	0.798	1.603	5.450	320.6	0.832	0.754	1.422
④352.2	0.702	348.1	0.711	1.164	1.282	349.5	0.710	0.767	1.140
⑤298.1	0.933	289.4	0.982	2.918	5.252	292.7	0.963	1.811	3.215
⑥318.2	0.914	321.1	0.879	0.911	3.829	322.2	0.900	1.257	1.532
⑦298.2	0.937	302.2	0.903	1.341	3.842	301.9	0.915	1.341	2.348
⑧362.2	0.592	368.0	0.561	1.601	5.236	367.4	0.578	1.436	2.365
⑨338.1	0.502	344.3	0.527	1.834	4.980	343.6	0.512	1.627	1.992
…	…	…	…	…	…	…	…	…	…

表 3 两种方法训练的时间和隐单元数比较

	K_mean 法	ROLS 法
训练时间/s	19	11
隐单元数目/个	18	10

6 结 论

仿真和实验结果表明，在训练样本数目大的情况下，递归最小二乘法训练 RBF 网络能有效的节约内存和计算量，从而加快 RBF 网络模型的训练速度。采用后向选择法选择中心后，网络中心减少，结构也得到简化。这样在满足精度的同时，也有效避免了 RBFNN 模型在实际预报过程中数值病态问题的发生。

参考文献：

[1] KAMINSKI W, STRUMILLO PI. Kernel orthonomalization in radial basis function neural networks[J]. IEEE Trans. Neural Networks, 1997,8(5):1 171 - 1 183.

[2] 王旭东，邵惠鹤．RBF 神经元网络在非线性系统建模中的应用[J]．控制理论与应用，1997,14(1):59 - 66.

[3] 靳蕃．神经计算智能基础原理与方法[M]．成都：西南交通大学出版社，2000.

[4] 陈涛，蒋林，屈染生．基于正交最小二乘学习算法的径向基函数网络设计[J]．中国机械工程，1997,8(6):95 - 97.

[5] 王碧泉，陈祖荫．模式识别理论、方法和应用[M]．北京：地震出版社，1989.

[6] 王永骥，涂健．神经元网络控制[M]．北京：机械工业出版社，1998.

[7] BARRY GOMM J, DING LI YU. Selection Radial Basis Function Network Centers with Recursive Orthogonal Least Squares Training[J]. IEEE Trans Neural Networks, 2000,11(2):306 - 314.

[8] AKAIKE H. Fitting autoregressive models for prediction[J]. Ann Inst Statist Math, 1969,21:425 - 439.

A New Learning Algorithm Based on Radial Basis Function Neural Network ——Recursive Orthogonal Least Squares (ROLS) Algorithm

ZHANG Xing-lan, CAO Chang-xiu, MEI Bin

(College of Automation, Chongqing University, Chongqing 400044, China)

Abstract: The Recursive Orthogonal Least Squares (ROLS) algorithm is applied to train Radial Basis Function Neural Network (RBFNN) when modeling, so as to save large memory and computational efforts. Using the information available from the trained network with ROLS algorithm, the effective centers of network can be obtained by adopting backward selection algorithm, which achieve acceptable accuracy with significant reduction of network structure. The results of simulation and experimentation show that the algorithm is efficient and useful.

Key words: recursive orthogonal least squares; RBFNN; backward selection; reduction

(责任编辑 吕赛英)