文章编号：1000－8152(2002)05－0673－08

RBF 网的动态设计方法

魏海坤[1]，丁维明[2]，宋文忠[1]，徐嗣鑫[1]
(1.东南大学 自动化研究所，南京 210096；2.东南大学 热能工程研究所，南京 210096)

摘要：提出一种 RBF 网的动态设计算法(DYNRBF 方法)，该算法有效地融合了 ROLS 算法和 RAN 网络的优点，不仅能动态调节 RBF 网的隐节点数，还能使网络的数据中心自适应变化．该方法所设计的 RBF 网不仅具有较好的泛化能力，当训练样本集变化时也具有好的鲁棒性．

关键词：RBF 网；结构设计；泛化能力
中图分类号：TP183 **文献标识码**：A

Dynamic method for designing RBF neural networks

WEI Hai-kun[1], DING Wei-ming[2], SONG Wen-zhong[1], XU Si-xin[1]
(1. Automation Institute, Southeast University, Nanjing 210096, China;
2. Thermal Energy Engineering Research Institute, Southeast University, Nanjing 210096, China)

Abstract: A dynamic designing method to improve RBF nets' generalization ability, which is called DYNRBF method, is proposed in this paper. The method combines the advantage of ROLS algorithm and RAN net, so the number and position of data centers of RBF net are both adapted during learning progress. RBF nets designed with DYNRBF method can not only generalize well, but remain good performance when train set is changed.

Key words: RBF net; structure design; generalization ability

1 引言(Introduction)

RBF 网设计的核心问题是确定隐节点的数目及相应的数据中心，设计出满足目标误差要求的尽可能小的神经网络，以保证神经网络的泛化能力．当 RBF 网络的隐节点数和数据中心确定后，RBF 网从输入到输出就形成了一个线性方程组，此时输出权矢量可用最小二乘方法获得．

RBF 网的设计方法可分为两大类：第一类方法中隐节点数据中心取随机值[1]或从样本输入中选取，如 OLS 算法[2]和 ROLS 算法[3]，进化优选算法(ESA 算法)[4]也属于这一类．这类算法的特点是数据中心一旦获得便不再改变，而隐节点的数目或者一开始就固定，或者在学习过程中动态调整．这类算法较容易实现，且能在权值学习的同时确定隐节点的数目，并保证学习误差不大于给定值．但是 OLS 算法并不一定能设计出具有最小结构的 RBF 网[5]，也无法确定基函数的扩展常数．另外，OLS 方法的数据中心从样本输入中选取是否合理，也值得进一步讨论．

第二类方法中数据中心的位置在学习过程中是动态调节的，如各种基于动态聚类(最常用的是 K-means 聚类或 Kohenen[6]提出的 SOFM 方法)的 RBF 网设计方法[7]，及 Platt[8]和 Kadirkamanathan[9]提出的资源分配网络(RAN)，及各种变结构学习方法[10,11]等．

比较之下，对数据中心进行动态调节显然更合理，因为在有限样本的情况下，最优的数据中心不一定正好位于样本输入点处，此时第一类方法显然是次优的．

聚类方法的优点是能根据各聚类中心之间的距离确定各隐节点的扩展常数，缺点是确定数据中心时只用到了样本输入信息，而没有用到样本输出信息；另外聚类方法也无法确定聚类的数目(RBF 网的隐节点数)．由于 RBF 网的隐节点数对其泛化能力有极大的影响[12,13]，所以寻找能确定聚类数目的合理方法，是聚类方法设计 RBF 网时需首先解决的问题．

除聚类方法和 OLS 方法外，资源分配网络(RAN)[8,9]也是一种重要的 RBF 设计方法．该方法循环地检查各样本输入输出对，当新样本满足“新性”(Novelty)条件时，则分配一个新节点．“新性”条

收稿日期：2000－09－30；收修改稿日期：2001－05－29．

件为：① 当前样本输入距离最近的数据中心超过某一定值 $\delta(t)$(距离准则)；② 网络的输出与样本输出的偏差大于某一定值 $e_{\min}$(误差准则).两个条件同时满足则分配新的隐节点.新隐节点的数据中心取当前样本的输入,输出权值取当前神经网络输出与当前样本输出之间的偏差,而扩展常数则取与该样本输入最近的数据中心之间的距离.如果当前神经网络的输出与新样本输出的偏差较小或样本输入离已有的数据中心都较近时,则不分配新隐节点,此时用梯度法调整各数据中心和权值以进一步消除误差.RAN 网络有以下问题：

1)“新性”条件对最终设计的神经网络规模有很大影响,且许多参数的设定有较大的随意性,尤其是分辨率 $\delta(t)$ 的取值.我们在试验中发现,即使 $\delta_{\max}$ 和 $\delta_{\min}$(尤其是 $\delta_{\min}$) 有较小的变化,也会使 RAN 的隐节点数变化较多,从而严重影响神经网络的泛化能力.

2) RAN 方法以单个样本的“新性”决定是否增加新隐节点并调节神经网络参数,尽管使得该方法可在线学习,也带来了两个问题:一是该方法的学习结果可能受样本输入顺序的影响;二是由于该方法在降低总体学习误差(所有样本的误差平方和)方面不如批处理方法,影响了最终神经网络的泛化能力.

针对上述问题,本文提出一种 RBF 网的动态设计算法(DYNRBF 方法),该算法有效地融合了 ROLS 算法和 RAN 网络的特点,不仅可以动态调节 RBF 网结构,同时使 RBF 网的数据中心能自适应变化,从而使最终设计的网络具有较小的结构和较好的泛化能力.

2 RBF 网结构描述(Formulation of RBF network structure)

考虑一个 n 输入单输出 RBF 网的设计.假设当前 RBF 网已有 M 个隐节点,且采用 Gaussian 型径向基函数.此时的 RBF 网模型(见图 1)为：

$$f(x) = \sum_{i=1}^{M} w_i \phi_{c_i}(x) + b. \tag{1}$$

其中 $x \in \mathbb{R}^n$ 为 RBF 网输入,$f(x) \in \mathbb{R}$ 为相应的输出,w_i 为第 i 个隐节点的输出连接权值,b 为输出偏移常数,$\phi_{c_i}(x) = e^{-\frac{\|x-c_i\|}{r_i}}$ 为 Gaussian 型径向基函数.$c_i \in \mathbb{R}^n$ 为已有的数据中心(未必为已有的样本输入),r_i 为该 RBF 函数的扩展常数.本文中我们令各 r_i 取相同的固定值.

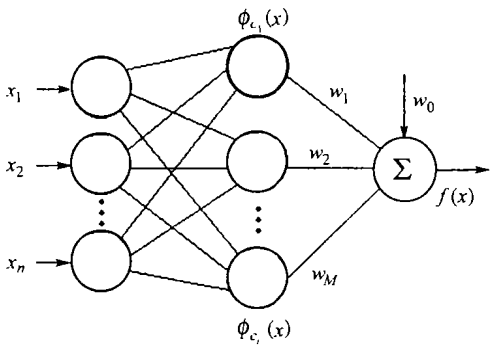


图 1 RBF网结构
Fig. 1 Structure of RBF net

尽管一些学者,如 Kadirkamanathan[9]认为,为描述简单起见,式(1)中的偏移项可以去掉,但我们认为,RBF 网模型中增加偏移项 b 是很有必要的.虽然从理论上说,没有偏移项,RBF 网也能以任意精度逼近紧集上的连续函数[14~16],但由于 RBF 网具有“局部”特性,RBF 网逼近常函数时将使用更多的隐节点和更长的学习时间[17].

假定 $T = \{(x_i, y_i) \mid i = 1,2,\cdots,N\}$ 为训练样本集,N 为训练样本数.称 $X = [x_1, x_2, \cdots, x_N]$ 为样本输入阵,相应的教师输出为 $Y = [y_1, y_2, \cdots, y_N]^{\mathrm{T}}$,而此时神经网络的输出为：

$$f(X) = P_M W + b = \sum_{i=1}^{M} w_i p_i + b. \tag{2}$$

其中 W 称为输出权矢量 $W = [w_1, w_2, \cdots, w_M]^{\mathrm{T}}$,$P_M = [p_1, p_2, \cdots, p_N]$,$p_i = [\phi_{c_i}(x_1), \phi_{c_i}(x_2), \cdots, \phi_{c_i}(x_N)]^{\mathrm{T}}$.一般来说,神经网络的输出 $f(X)$ 与教师输出总是存在偏差(矢量)e,即

$$Y = f(X) + e = P_M W + b + e = \tilde{P}_M \tilde{W} + e. \tag{3}$$

其中 $\tilde{P}_M = [p_1, p_2, \cdots, p_M, 1_N]$,$1_N$ 为元素全 1 的列向量,$\tilde{W} = [w_1, w_2, \cdots, w_m, b]^{\mathrm{T}}$.

在神经网络训练时,为了使偏差 e 维持在合理水平,我们必须动态调整 RBF 网的参数,如 W 和 b.但事实上,在 RBF 网的数据中心和扩展常数确定的情况下,W 和 b 实际上可由最小二乘方法得到,所以 W 和 b 的调节能力是有限的.另一种更好的方法是在学习过程中动态调节隐节点的数目及数据中心,这正是本文的讨论内容.

3 RBF 网动态设计方法(Dynamic designing method of RBF net)

DYNRBF 方法设计 RBF 网的过程分为“粗调”和“精调”两个阶段.粗调阶段不进行数据中心的调整,只动态增加隐节点的数目,并选取相应的样本输

入作为数据中心,直至隐节点数满足一定要求;精调阶段则用 Gaussian 正则化方法对粗调得到的 RBF 网进行学习,其中包括冗余隐节点的动态删除、数据中心和连接权值的动态调整.下面我们详细介绍这两个过程.

3.1 粗调过程(Rough tuning)

粗调过程类似于正则化前向选取[3]的过程,该过程的关键是按什么原则选取新的隐节点,及何时终止这一过程.

3.1.1 新隐节点选取(Selection of new hidden units)

如果当前神经网络对目标函数的逼近不能令人满意,或者已有的 M 个隐节点不能有令人满意的逼近,我们就应该增加 RBF 网的隐节点.

考虑每次只增加一个隐节点的情况.由于我们对目标函数没有任何先验知识,我们先从样本输入中选取数据中心,然后再逐步调整该数据中心的值.下面我们来讨论目标函数(能量函数)具有 Gaussian 正则化项时数据中心的选取问题.

由式(3)可见,如果把教师输出 Y 看成一个 N 维矢量,则可看作各 p_i 及 I_N 的线性组合,而 $\{p_1, p_2, \cdots, p_M, I_N\}$ 可看成 N 维空间的一组基,此时 Y 在该基下的坐标即为 $\tilde{W}$.

采用 Gaussian 正则化方法时的目标函数为:

$$E = e^{\mathrm{T}}e + \lambda \tilde{W}^{\mathrm{T}}\tilde{W}. \tag{4}$$

其中 λ 为正则化系数.上式也可写为

$$E = (Y - \tilde{P}_M\tilde{W})^{\mathrm{T}}(Y - \tilde{P}_M\tilde{W}) + \lambda\tilde{W}^{\mathrm{T}}\tilde{W}. \tag{5}$$

令 $\frac{\partial E}{\partial \tilde{W}} = 0$,可解得:

$$\tilde{W}_M = (\tilde{P}_M^{\mathrm{T}}\tilde{P}_M + \lambda I_M)^{-1}\tilde{P}_M^{\mathrm{T}}Y. \tag{6}$$

即当 $\tilde{W} = \tilde{W}_M$ 时,式(4)有极小值,此时式(4)有最小的正则化能量:$E_M = e_M^{\mathrm{T}}e_M + \lambda\tilde{W}_M^{\mathrm{T}}\tilde{W}_M$,整理后即得:

$$E_M = Y^{\mathrm{T}}H_MY. \tag{7}$$

其中 $H_M = I_N - \tilde{P}_M(\tilde{P}_M^{\mathrm{T}}\tilde{P}_M + \lambda I_M)^{-1}\tilde{P}_M^{\mathrm{T}}$.

由于新隐节点的数据中心从样本输入中选取,假定该数据中心为 x_i,令 $s_i = [\phi_{x_i}(x_1), \phi_{x_i}(x_2), \cdots, \phi_{x_i}(x_N)]^{\mathrm{T}}$,并称 $S = [s_1, s_2, \cdots, s_N]$ 为设计阵.令 $\tilde{P}_{M+1} = [\tilde{P}_M\ s_i]$,则此时的正则化能量为 $E_{M+1} = Y^{\mathrm{T}}H_{M+1}Y$,于是我们应该选择 s_j,使之满足

$$E_{M+1}(s_j) = \min\{E_{M+1}(s_i), i = 1,2,\cdots,N\}. \tag{8}$$

鉴于式(8)计算量较大,我们参考 Orr[3]的做法来减少计算量.

由于 $E_M - E_{M+1} = Y^{\mathrm{T}}(H_M - H_{M+1})Y$,而

$$H_M - H_{M+1} = \tilde{P}_{M+1}(\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1} + \lambda I_{M+1})^{-1}\tilde{P}_{M+1}^{\mathrm{T}} - \tilde{P}_M(\tilde{P}_M^{\mathrm{T}}\tilde{P}_M + \lambda I_M)^{-1}\tilde{P}_M^{\mathrm{T}}.$$

对矩阵 $\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1} + \lambda I_{M+1}$ 求逆阵,并将 $\tilde{P}_{M+1}(\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1} + \lambda I_{M+1})^{-1}\tilde{P}_{M+1}^{\mathrm{T}}$ 展开后整理,可得以下结果:

$$H_M - H_{M+1} = \frac{\tilde{P}_M s_i s_i^{\mathrm{T}}\tilde{P}_M^{\mathrm{T}}}{\lambda + s_i^{\mathrm{T}}\tilde{P}_M s_i}. \tag{9}$$

于是我们有

$$E_M - E_{M+1} = \frac{(Y^{\mathrm{T}}\tilde{P}_M s_i)^2}{\lambda + s_i^{\mathrm{T}}\tilde{P}_M s_i}. \tag{10}$$

由式(8),(10)可知:从设计阵 S 中选择一列使式(8)最小,等同于从设计阵 S 中选择一列,该列应使式(10)达到最大值.式(10)的计算量较小.

顺便指出,选取 RBF 网第一个数据中心时,可从样本输入中选择某一 x_i,使相应的 s_i 在 Y 上的投影最大,即:

$$E_1(x_i) = \max\{Y^{\mathrm{T}}s_j, j = 1,2,\cdots,N\}. \tag{11}$$

3.1.2 粗调的停止(Stop of rough tuning)

当我们每次增加一个新隐节点时,$\tilde{P}_M$ 将增加一列 s_j.如果 RBF 网增加这一新节点后学习误差下降有限,则意味着 s_j 与 $\tilde{P}_M$ 各列有较大的相关性,因此 $\tilde{P}_{M+1}$ 将更趋于病态.我们知道,矩阵的条件数可用于衡量其病态程度,因此我们也可以根据方阵 $\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1}$ 的条件数来决定是否终止粗调过程.具体地说,当下式满足时

$$C(\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1}) > C_{\max}, \tag{12}$$

我们就停止粗调.其中 $C(A) = \|A\|\,\|A^{-1}\|$ 为矩阵 A 的条件数,$\|A\|$ 为 A 的 Frobenius 范数.$C_{\max}$ 是一个需预先确定的量,一般可选为 10^6 量级,因为矩阵条件数超过这一值,一些计算软件,如 Matlab,会给出警告.

使用条件数准则的一个明显好处,是可以防止在粗调过程选入冗余隐节点,并在计算时避免出现因 $\tilde{P}_{M+1}$ 病态而出现的异常(如算法终止).

如果 $\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1}$ 的条件数下降过快,式(12)可改为 $C(\tilde{P}_{M+1}^{\mathrm{T}}\tilde{P}_{M+1} + \lambda I_{M+1}) > C_{\max}$,$\lambda$ 为正则化系数,可取一接近于零的正数.

3.2 精调过程(Precise tuning)

精调过程包括隐节点数据中心调节、输出权值和偏移量调节及冗余隐节点剪枝等几部分.

3.2.1 数据中心调整(Tuning of data centers)

由于新隐节点的数据中心来自样本输入,该数据中心的值可能离最优的数据中心有一定的偏差.为校正这一偏差,应在学习过程中动态调节各数据中心的值.

考虑到RBF网的局部特性,我们以该数据中心周围的部分样本为目标样本,调节该数据中心的值.假设某隐节点的数据中心为 c_i,扩展常数为 r_i,则参与调节的目标样本为

$$A_i = \{(x_j, y_j) \mid \| x_j - c_i \| < \kappa r_i, j = 1,2,\cdots,N\}. \tag{13}$$

κ 称为重叠系数.κ 值越大,参与调节数据中心的样本就越多.

我们以梯度法调节数据中心 c_i 的值.假定 $(x_j, y_j) \in A_i$ 为某一参与调节的目标样本,神经网络对应该样本输入 x_j 的输出为 $f(x_j)$.定义误差为 $\epsilon = \| f(x_j) - y_j \|^2$,根据梯度下降法,数据中心 c_i 对样本 (x_j, y_j) 的调节量为:

$$\Delta c_i(x_j, y_j) = 4\frac{\eta}{r_i}(x_j - c_i)o_i(y_j - f(x_j))w_i. \tag{14}$$

其中 $O_i = \phi_{c_i}(x_j)$,w_i 为该隐节点的前一调节时刻的输出权值,η 为学习率.

式(14)有一个直观的解释[7]:对样本输入 x_j,如果神经网络输出小于目标值(即 $y_j - f(x_j) > 0$)则数据中心被拉向 x_j;如果神经网络输出大于目标值(即 $y_j - f(x_j) < 0$)则数据中心将被推离 x_j.其实,在隐节点扩展常数不变的情况下,数据中心的这种变化是容易理解的.

于是第 i 个隐节点的调节公式为:

$$c_i = c_i + \sum_{(x_j, y_j) \in A_i} \Delta c_i(x_j, y_j). \tag{15}$$

3.2.2 输出权值调整(Tuning of output weights)

每调节一次RBF网的数据中心,我们都应该调节其输出权值和偏移.实际上,当网络的数据中心确定后,最优权值可通过最小化式(5)直接得到,即可得到式(6)的结果,于是我们有

$$[w_1, w_2, \cdots, w_M, b]^{\mathrm{T}} = \tilde{W}_M. \tag{16}$$

3.2.3 冗余节点剪除(Pruning redundant units)

DYNRBF对冗余隐节点的删除是通过正则化实现的.如前所述,我们在学习过程中增加了Gaussian正则化项,由于该正则化项具有剪枝特性[18,19],因此在精调过程中,一些冗余的输出权值将逐步衰减到零,从而可以剪除与之相连的隐节点.在实际操作中,当某隐节点输出权值 w_i 满足

$$\mathrm{abs}(w_i) < w_{\min} \tag{17}$$

时则删除该隐节点,其中 $w_{\min}$ 为临界权值.

正则化系数 λ 的取值对学习结果有很大的影响.为解决这一问题,一个合理的方法是在学习过程中动态改变 λ 的值,即在每次数据中心修正后,动态修改 λ.

我们参考Weigend等人[20]的思路,提出 λ 的动态修改策略.为达到上述目的,我们在学习过程中随时检测以下误差量之间的关系:

· $E(t-1)$:前一次数据中心调节时的误差.

· $A(t)$:当前时刻的加权平均误差,定义为 $A(t) = \mu A(t-1) + (1-\mu)E(t)$,其中 μ 为接近于1的数.

· D:期望误差值.如果没有先验知识,可设定 $D = 0$,但此时计算时间可能较长.

算法初始化时,我们令 λ 取粗调时的值,而在学习过程中,每次数据中心按式(12)和(13)调节后,我们计算当前时刻的学习误差 $E(t)$ 和加权平均误差 $A(t)$,并根据它们之间的关系对 λ 进行调节.具体规则如下:

· 如果 $E(t) < E(t-1)$,或 $E(t) < D$,则 $\lambda(t) = \lambda(t-1) + \Delta\lambda$.

· 如果 $E(t) \geqslant E(t-1)$,$E(t) < A(t)$,且 $E(t) \geqslant D$,则 $\lambda(t) = \lambda(t-1) - \Delta\lambda$.

· 如果 $E(t) \geqslant E(t-1)$,$E(t) \geqslant A(t)$,且 $E(t) \geqslant D$,则 $\lambda(t) = \rho\lambda(t-1)$,其中 ρ 为接近1的系数.

3.3 算法实现(DYNRBF realization)

综上所述,RBFDNC算法的实现程序如下:

· Step 1 根据式(11)选择RBF网第一个数据中心,并计算输出权值.

· Step 2 粗调阶段.根据式(10)或(8)选取RBF网的数据中心,直至满足停止准则(12).

· Step 3 进入精调阶段.根据式(14)和(15)调节网络各数据中心的值.

· Step 4 根据式(6)和(16),调整网络的输出权值和输出偏移.

· Step 5 根据式(17)对隐节点进行剪枝.

· Step 6 计算当前所有训练样本的总误差 $E(t)$ 和平均误差 $A(t)$.

· Step 7　如果 $E(t)$ 已达到给定值 D，或算法已达到给定运算次数，则结束算法，否则转 Step 7.

· Step 8　根据 $E(t)$，$A(t)$ 和 D 间的关系调整正则化系数，然后转 Step 3，继续进行精调.

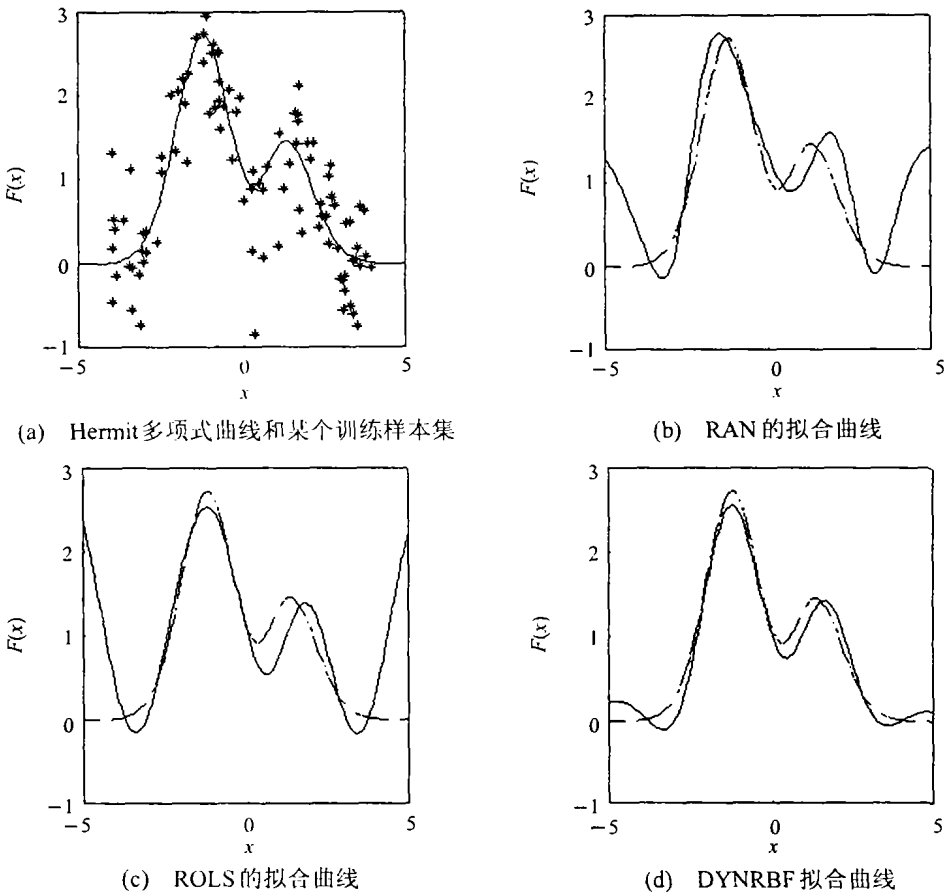
4　仿真和应用(Simulation and application)

4.1　函数逼近(Function approximation)

考虑以下 Hermit 多项式的逼近问题[21]：

$$F(x) = 1.1(1 - x + 2x^2)\exp\left(-\frac{x^2}{2}\right). \quad (18)$$

其中 $x \in \mathbb{R}$. 训练样本产生方式如下[3]：样本数 N = 100，样本输入 x_i 服从区间[-4,4]内的均匀分布，样本输出为 $F(x_i) + e_i$，e_i 为添加的噪声，服从均值为0，方差为0.5的正态分布. 图 2(a)显示了一个典型的训练样本集. 图 2 的(b)，(c)，(d)显示了 RAN，ROLS，DYNRBF 三种方法对该训练样本集的拟合曲线. 其中 RAN 产生 7 个隐节点. 学习参数为：$\delta_{max} = 2.0$，$\delta_{min} = 1.0$，$\gamma = 0.977$，$\kappa = 0.87$，学习率 $\eta = 0.05$，$e_{min} = 0.2$，共学习 400 次. ROLS 方法产生 25 个隐节点. 学习参数为：扩展常数 $r = 1.5$，正则化系数为 $\lambda = 3e-4$，条件数极限 $C_{max} = 1e6$.



(a) Hermit多项式曲线和某个训练样本集　(b) RAN的拟合曲线

(c) ROLS的拟合曲线　(d) DYNRBF拟合曲线

图 2　训练样本集和各种方法的拟合曲线

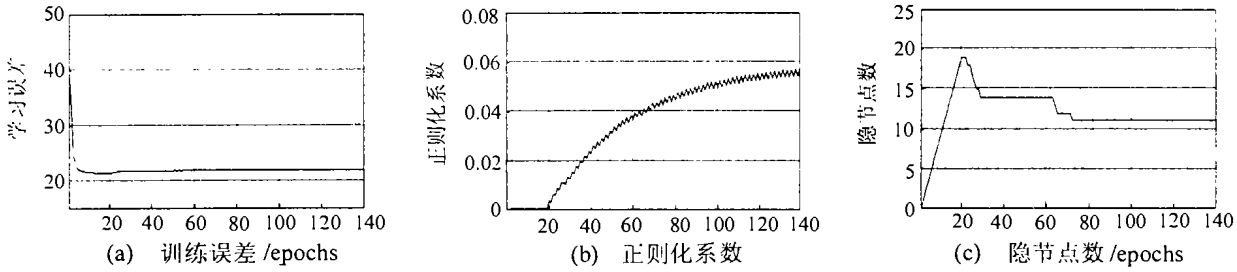
Fig. 2　Training set and fitting curves of RAN, ROLS and DYNRBF

与 RAN 和 ROLS 方法相比，DYNRBF 产生 11 个隐节点. 学习参数为：扩展常数 $r = 1.5$，$\kappa = 1.0$，$\lambda = 3e-4$，正则化系数增量 $\Delta\lambda = 2e-3$，$\mu = 0.95$，$\rho = 0.95$，学习率 $\eta = 1e-4$，条件数极限 $C_{max} = 1e6$，目标误差设为 0，剪枝临界权值 $w_{min} = 0.1$，共学习 150 次.

图 3 显示了学习过程中 RBF 网的学习误差、正则化系数、隐节点数变化的动态过程. 由图 3 可见，在学习的粗调阶段，随着隐节点数的增加，学习误差迅速减小，当隐节点数到达 19 后，神经网络也达到条件数极限，于是粗调结束，进入精调阶段. 精调阶段学习误差变化不大，但随着正则化系数的增加，神经网络的隐节点数逐步减少，当到达第 71 个 epoch 时，神经网络的隐节点数为 11，此后尽管正则化系数仍有所上升，但隐节点数已不再变化. 可见精调过程在不减小训练误差的情况下，不仅调节了神经网络的权值，还简化了 RBF 网的结构.

为了进一步对比三种方法的性能，我们绘制了

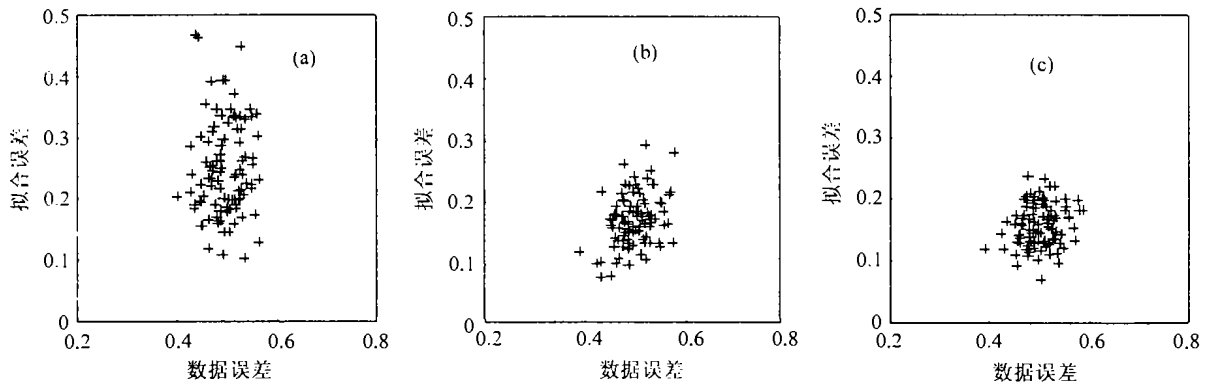
三种方法对 100 个训练样本集的数据误差-拟合误差图,如图 4 所示.数据误差为样本输出与目标输出的平均误差,即$\sqrt{1/N\sum_{1}^{N}e_i^2}$,拟合误差是神经网络输出对目标输出在[-4,4]范围内 100 个等间隔点的平均误差,即$\sqrt{1/100\sum_{1}^{100}(G(\tilde{x}_i)-f(\tilde{x}_i))^2}$,表示学习后 RBF 网的泛化误差.在图 4 中数据误差为横轴,拟合误差为纵轴,每个训练集对应一个点.由图 4 可见,数据误差集中在噪声方差 0.5 附近.



(a) 训练误差 /epochs (b) 正则化系数 (c) 隐节点数 /epochs

图 3 DYNRBF学习过程中学习误差、正则化系数隐节点数的变化

Fig. 3 Change of training eror, regular and number of hidden units during DYNRBF learning



(a) RAN 方法 (b) ROLS 方法 (c) DNYRBY 方法

图 4 100个样本时三种方法的数据误差 -拟合误差图

Fig. 4 Data-fitting error plot of three methods when using 100 training samples

表 1 显示了对 100 个训练集三种方法的一些主要性能对比.三种方法学习时,所有的学习参数都已整定至最佳.

表 1 三种方法学习后神经网络的主要性能对比

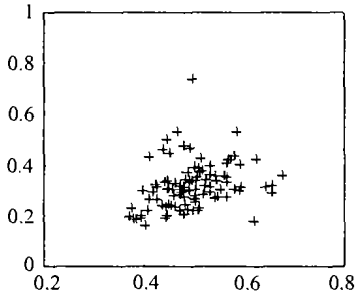
Table 1 Comparing of RBF nets obtained with three methods

	RAN	ROLS	DYN RBF	ROLS	DYN RBF
训练样本数	100	100	100	40	40
平均隐节点数	8.35	21.6	16.1	27.5	18.6
隐节点数方差	1.037	5.471	8.628	16.47	18.64
平均数据误差	0.496	0.499	0.501	0.496	0.489
平均拟合误差	0.265	0.172	0.157	0.327	0.231
拟合误差方差	0.008	0.001	0.001	0.016	0.003

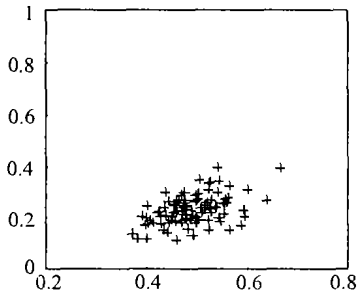
由图 4 和表 1 可见,尽管 RAN 方法所设计的神经网络规模较小,但 RAN 方法的泛化能力不如 DYNRBF 方法和 ROLS 方法.我们曾尝试减小 RAN 方法的最小分辨率 δ_{min} 和衰减常数 γ,结果是隐节点数急剧上升,同时拟合误差也迅速增加.相比之下,DYNRBF 方法设计的 RBF 网不仅结构较小,测试误差也较小,说明 DYNRBF 方法的精调过程既精简了 RBF 网结构,还改善了网络的泛化能力.ROLS 方法的测试误差与 DYNRBF 达到可比较的程度,表明 DYNRBF 的粗调过程较为有效.另外 DYNRBF 方法的拟合误差方差小于 ROLS,说明精调过程在改进 RBF 网泛化的同时,也改善了神经网络对训练样本变化的鲁棒性.

在上述试验中我们使用了 100 个样本.由于样本数较多,神经网络数据中心点的最优值与精调后的优化值之间偏差不大,因此精调在改善泛化能力方面的作用不是很明显.如果我们减少训练样本数,我们就会发现 ROLS 方法的测试误差将大于 DYNRBF.在学习参数和噪声方差不变的情况下,我们令

训练样本数为40,依然用100个训练集进行测试,结果如图5和表1右边所示.可见此时DYNRBF的测试误差与ROLS方法有了较大差别,说明在训练样本数较少的情况下DYNRBF性能明显优于ROLS方法.另外DYNRBF的拟合误差方差依然只有0.003,说明DYNRBF方法依然维持了对训练样本变化的较好的鲁棒性.



(a) ROLS方法



(b) DYNRBF方法

图5 40个训练样本时三种方法的数据误差-拟合误差图

Fig. 5 Data-fitting error plot of three methods when using 40 training samples

4.2 煤灰软化温度预测(Prediction of coal ash softening temperature)

煤灰的软化温度直接影响着燃煤的结渣等特性,所以对煤灰的软化温度进行准确预测,是成功进行动力配煤的前提.煤灰中含有多种氧化物,如SiO_2,Al_2O_3,Fe_2O_3,CaO,MgO,TiO_2,K_2O和Na_2O等,它们的含量决定煤灰的软化温度.因此我们用一个8输入1输出结构的RBF网实现这一模型,其中8个输入分别为上述8个氧化物含量,输出为软化温度预测值.用于训练神经网络的样本共155个,用于测试的样本共50个样本.所有这些样本均通过试验获得.建模进行前,我们将所有样本输入都归一化到[0,1]范围内.

我们用DYNRBF方法和RAN方法实现上述建模问题.DYNRBF方法的运算次数为180,扩展常数0.9,正则化系数2e-4,条件数极限1e6,正则化增量8.0e-3,滤波系数0.95,目标误差为0,学习系数1e-8,删除极限权值1.0.对RAN方法,学习过程中参数为:重叠系数1.0,最大分辨率3.0,最小分辨率1.0,衰减常数160,输出权调节步长0.01,输出偏移调节步长0.01,数据中心调节步长2e-6,循环次数155,样本期望精度60.

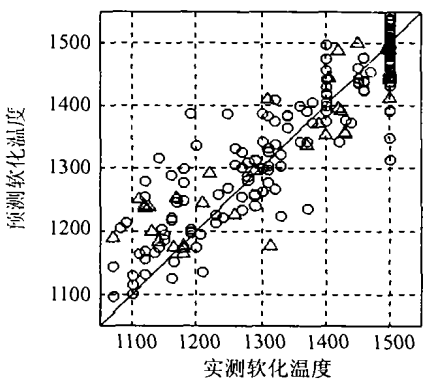
两种方法学习后的结果如表2所示.图6是两个预测模型对测试样本和训练样本的预测软化温度-实测软化温度图.图中,横轴均为实测输出,纵轴均为模型输出,图中的"△"为测试样本点,"。"为训练样本点,对角直线表示预测值和测试集完全一致.

由表2和图6可见,同RAN方法相比,DYNRBF方法所设计的RBF网不仅训练误差和测试误差均较小,神经网络的规模也小得多.

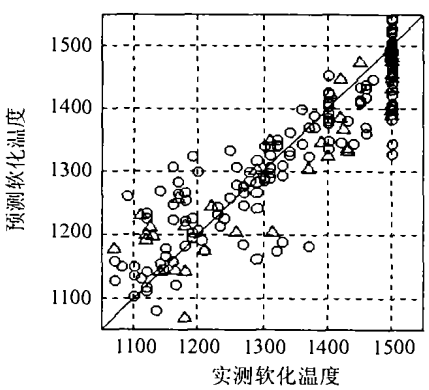
表2 DYNRBF和RAN方法的学习结果比较

Table 2 Comparing of RBF nets obtained with DYNRBF and RAN methods

	学习误差	测试误差	隐节点数
RAN	5.661e+005	1.864e+005	10
DYNRBF	5.401e+005	1.846e+005	4



(a) RAN方法



(b) DYNRBF方法

图6 RAN和DYNRBF的预测软化温度-实测软化温度

Fig. 6 Predicting-test softening temperature of RBF nets obtained with DYNRBF and RAN method

5 讨论(Discussion)

本文提出一种RBF网的动态设计方法,该方法

可以在学习过程中动态调节神经网络的规模和网络参数,包括各隐节点数据中心的位置.试验表明,DYNRBF方法设计的神经网络不仅有较小的结构,而且有较好的泛化能力,另外,该算法的鲁棒性也较好,即对不同的训练样本集,该方法所设计的神经网络有较小的拟合误差方差.该方法有以下问题值得讨论:

1) 在DYNRBF方法中,粗调可看作是一个构造过程,而精调则是一个剪枝过程,这两个过程是分别进行的.由于如同所有剪枝算法所要求的那样,我们必须在粗调阶段产生一个"较大规模"的网络,然后再进行剪枝,这无疑增加了学习时间.如何把并行的进行剪枝和构造,即在学习过程中,当神经网络欠拟合时增加隐节点,而在神经网络过拟合时减少隐节点,显然是一个更令人感兴趣的问题.

2) 在DYNRBF方法中,我们使用了固定的扩展常数.尽管扩展常数的取值不影响RBF网的逼近能力,但扩展常数取值对网络的结构和泛化能力都有影响.显然,扩展常数取值决定于目标函数的情况,但是在只有训练样本的情况下,如何设定扩展常数的初值,或者在学习过程中对该值进行动态调整,是值得进一步研究的问题.

参考文献(References)

[1] Broomhead D S, Lowe D. Multi-variable functional interpolation and adaptive networks [J]. Complex Systems, 1988, 2:321 - 355

[2] Chen S, Cowan C F N, Grant P N. Orthogonal least squares learning algorithms for radial basis function networks [J], IEEE Trans. Neural Networks, 1991, 2(2):302 - 309

[3] Orr M J L. Regularization in the selection of radial basis function centers [J]. Neural Computation, 1995,7:606 - 623

[4] Wei Haikun. Xu Sixin, Song Wenzhong. An evolutionary selecting algorithm for designing minimal RBF nets [J]. Control Theory & Applications.2000, 17(4): 604 - 608

[5] Sherstinsky A, Picard R W. On the efficiency of the orthogonal least squares training method for radial basis function networks [J]. IEEE Trans. Neural Networks, 1996,7(1):195 - 200

[6] Kohonen T. Self-Organization and Associative Memory [M]. Berlin: Springer-Verlag, 1984

[7] Moody J, Darken C. Fast learning in networks of locally-tuned processing units [J]. Neural Computation, 1989, 1:281 - 294

[8] Platt J. A resource-allocating network for function inerpolation[J]. Neural Computation, 1991, 3:213 - 225

[9] Kadirkamanathan V. A function estimation approach to sequential learning with neural networks [J]. Neural Computation, 1993, 5: 954 - 975

[10] Song Aiguo. A novel on-line learning structure variable radial basis function nets with application on passive sonar target classification [J]. Acta Electronica Sinica, 1999, 27(10):65 - 69

[11] Yingwei L, Sundararajan N, Saratchandran P. Identification of time-varying nonlinear systems using minimal radial basis function neural networks [J]. IEE Proc.-Control Theory Appl., 1997, 144 (2): 202 - 208

[12] Moody J E. The effective number of parameters: an analysis of generalization and regularization in nonlinear learning system [A]. In: Advances in Neural Information Processing Systems [C]. San Mateo, 1992,4:847 - 854

[13] Niyogo P, Girosi F. On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis function [J]. Neural Computation, 1996,8:819 - 842

[14] Hartman E, Keeler J D, Kowalski J. Layered neural networks with Gaussian hidden units as universal approximators [J]. Neural Computation, 1990, 2:210 - 215

[15] Park J, Sandberg I W. Universal approximation using radial-basis-function [J]. Neural Computation, 1991,3:246 - 257

[16] Gorisi F, Poggio T. Networks and the best approximation property [J]. Boil. Cybernet., 1990, 63:169 - 176

[17] Hartman T, Keeler J D. Predicting the future: Advantages of semilocal units [J]. Neural Computation , 1992, 3:566 - 578

[18] Krogh A, Hertz J H. A simple weight decay can improve generalization [A]. In: Advances in Neural Information Processing Systems [C]. San Mateo, 1992,4: 950 - 957

[19] Hansen L K, Rasmussen C E. Pruning from adaptive regularization [J]. Neural Computation, 1994, 6:1223 - 1232

[20] Weigend A S, Rumelhart D E, Huberman B A. Generalization by weight-elimination with application to forcasting [A]. In: Advances in Neural Information Processing Systems [C]. San Mateo, 1991, 3:857 - 882

[21] Mackay D J C. Bayesian interpolation [J]. Neural Computation. 1992, 4(3):415 - 447

本文作者简介

魏海坤 1971年生.1994年毕业于北方工业大学工业自动化专业,1997年、2000年东南大学分获控制学科硕士,博士学位.现在东南大学自动化所作博士后研究工作.研究方向为神经网络,神经网络泛化理论和方法,神经网络控制. Email:hkwei@seu.edu.cn

丁维明 1959年生.1982年毕业于南京工学院数力系,现为东南大学热能工程研究所副教授.研究领域为神经网络,模糊控制及DCS系统.

宋文忠 1936年生.1959年毕业于南京工学院动力系,现为东南大学自动化所教授,博士生导师.研究领域为复杂系统辨识及控制.

徐嗣鑫 1940年生.1962年毕业于南京工学院动力系,现为东南大学自动化所教授.研究领域为模糊控制,人工神经网络以及电厂、造纸厂的计算机控制与管理.