第 30 卷 第 4 期
2004 年 7 月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 30, No. 4
July, 2004

一种递归神经网络的快速并行算法[1)]

李鸿儒　　顾树生

（东北大学信息科学与工程学院　沈阳　110004）

（E-mail：hongru_li@163.com）

摘　要　针对递归神经网络 BP(Back Propagation)学习算法收敛慢的缺陷，提出一种新的递归神经网络快速并行学习算法. 首先，引入递推预报误差(RPE)学习算法，并且证明了其稳定性；进一步地，为了克服 RPE 算法集中运算的不足，设计完整的并行结构算法. 本算法将计算分配到神经网络中的每个神经元，完全符合神经网络的并行结构特点，也利于硬件实现. 仿真结果表明，该算法比传统的递归 BP 学习算法具有更好的收敛性能. 理论分析和仿真实验证明，该算法与 RPE 集中运算算法相比可以大大节省计算时间.

关键词　递归神经网络，递推预报误差，并行算法，集中运算

中图分类号　TP183

A Fast Parallel Algorithm for a Recurrent Neural Network

LI Hong-Ru　　GU Shu-Sheng

(*School of Information Science and Engineering, Northeastern University, Shenyang*　110004)

(E-mail: hongru_li@163.com)

Abstract　Because the recurrent BP algorithm suffers from the drawback of slow convergence, a new fast parallel learning algorithm for recurrent neural networks is proposed. First, the recursive predictive error(RPE) learning algorithm for recurrent neural networks is introduced, and its stability is demonstrated. Furthermore, in order to overcome the disadvantage of the centralized computing of RPE learning algorithm, a parallel structure algorithm is derived. In the new parallel learning algorithm, the computation is distributed to each neuron in the network, which is coherent with the massively parallel nature of the network, and also convenient for hardware implementation. Simulation results show that better convergence performance of the proposed algorithm over the traditional recurrent BP algorithm. Meanwhile, theoretical analysis and simulation results show that the new parallel algorithm also cut lots of computational time compared with the RPE centralized computing algorithm.

Key words　Recurrent neural networks, recursive predictive error, parallel algorithm, centralized computing

1) 国家“十五”重点攻关计划项目(2001BA401A06-0.4)资助
Supported by National 10th Five-year Key Plan Project of P. R. China(2001BA401A06-0.4)
收稿日期　2002-10-08　　收修改稿日期　2003-04-28
Received October 8, 2002; in revised form April 28, 2003

1　引言

神经网络由于具有理论上能逼近任意非线性函数的能力，在非线性系统的建模、辨识与控制上有广泛的应用[1~6]. 递归神经网络能通过内部反馈环节自动存储动态信息并实现动态映射，用于系统辨识与控制时无需知道详细的对象知识，这是其最突出的优点. 正因为如此，近年来递归神经网络的研究引起了人们的极大兴趣[4~9].

构成递归神经网络模型的方法有很多，对角递归神经网络(以下简称 DRNN(Diagonal Recurrent Neural Network))可以说是一种最简单的递归神经网络，但像其它的递归神经网络一样具有记忆功能，能反映以前时刻的网络输入，本文即以这种递归神经网络为研究对象.

神经网络的训练算法一直是其研究的重要方面，无论是前馈型网络还是递归型网络[6~10]. 如何寻求一种快速的递归神经网络学习算法具有很强的实用价值，已引起国内外许多专家学者的关注. 遗憾的是，这些新的算法中，只是突出了算法收敛速度快的问题，变 BP 算法的并行运算为集中运算了，而恰恰忽略了神经网络最根本的特点之一，就是并行运算. 本文将传统的参数估计方法——递推预报误差(RPE)法引入到递归神经网络权值的学习中来，克服了递归 BP 学习算法收敛速度慢的不足，所提出的算法仍然具有并行运算特点，对其它一些形式的递归神经网络可直接推出. 作者认为，本文的设计思想对研究神经网络学习算法的同行学者有一定借鉴作用.

2　对角递归神经网络的结构

DRNN 由输入层、隐层和输出层组成. 设在每个离散时刻 k，$X_i(k)$ 为网络输入，$i=1,2,\cdots,n$；$O(k)$ 为网络输出，为了表示方便，取一个输出，对于多输出情况可直接推得；$S_j(k)$，$Z_j(k)$ 分别为隐层输入和输出，$j=1,2,\cdots,m$；W_{ji}^h 为输入层至隐层的连接权，W_i^o 为隐层至输出层的连接权，W_i^d 为递归神经元连接权，则

$$S_j(k)=W_j^d\cdot Z_j(k-1)+\sum_{i=1}^{n}W_{ji}^h\cdot X_i(k),\ Z_j(k)=\varphi(S_j(k)),\ O=\sum_{j=1}^{m}W_j^o\cdot Z_j(k)$$

其中 $\varphi(\cdot)$ 是非线性激活函数，通常为 Sigmoid 型函数. 取合适的性能指标函数 J 就可以按各种学习算法对网络的权进行训练，求得最优权值 W^*，使 J 最小.

3　递归神经网络的递推预报误差算法

3.1　递归神经网络的 RPE 算法

假设由网络所有权按顺序组成的参数向量为 $\boldsymbol{\theta}$，其维数为神经网络所有权的总和 $n_{\boldsymbol{\theta}}$，$\boldsymbol{e}(k,\boldsymbol{\theta})$ 为预报误差矢量，N 是数据长度. 定义性能指标为

$$J(\boldsymbol{\theta})=\frac{1}{2N}\cdot\sum_{k=1}^{N}\boldsymbol{e}^{\mathrm{T}}(k,\boldsymbol{\theta})\cdot\boldsymbol{e}(k,\boldsymbol{\theta}) \tag{1}$$

RPE 算法参数向量的调整算式为

$$\boldsymbol{\theta}(k)=\boldsymbol{\theta}(k-1)+\mu(k)\cdot\boldsymbol{\sigma}[\boldsymbol{\theta}(k-1)] \tag{2}$$

式中 $\mu(k)$为学习速率,$\boldsymbol{\sigma}(\boldsymbol{\theta})$为 Gauss-Newton 搜索方向,定义为

$$\boldsymbol{\sigma}(\boldsymbol{\theta})=-\left[H(\theta)\right]^{-1}\nabla \boldsymbol{J}(\boldsymbol{\theta}) \tag{3}$$

式中$\nabla J(\boldsymbol{\theta})$是 $J(\theta)$关于 $\boldsymbol{\theta}$ 的梯度,$H(\theta)$是 $J(\theta)$的 Hessian 矩阵,它们分别为 $J(\theta)$关于 $\boldsymbol{\theta}$ 的一阶和二阶微分

$$\nabla J(\boldsymbol{\theta})=-\frac{1}{N}\cdot\sum_{k=1}^{N}\boldsymbol{\Psi}(k,\theta)\cdot \boldsymbol{e}(k,\boldsymbol{\theta}),\quad H(\theta)=\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}(k,\theta)\cdot\boldsymbol{\Psi}^{\mathrm{T}}(k,\theta)$$

其中 $\boldsymbol{\Psi}(\theta)=\left[\mathrm{d}\hat{O}(k,\theta)/\mathrm{d}\theta\right]^{\mathrm{T}}$是网络的一步预报值对 $\boldsymbol{\theta}$ 的一阶微分. RPE 算法整理为

$$\boldsymbol{e}(k)=O(k)-\hat{O}(k) \tag{4}$$

$$M(k)=\frac{P(k-1)\cdot\boldsymbol{\Psi}(k)}{\lambda(k)+\boldsymbol{\Psi}^{\mathrm{T}}(k)P(k-1)\boldsymbol{\Psi}(k)} \tag{5}$$

$$P(k)=\frac{1}{\lambda(k)}\cdot\left[I+M(k)\boldsymbol{\Psi}^{\mathrm{T}}(k)\right]\cdot P(k-1) \tag{6}$$

$$\boldsymbol{\theta}(k)=\boldsymbol{\theta}(k-1)+\mu(k)\cdot P(k)\boldsymbol{\Psi}(k)\boldsymbol{e}(k) \tag{7}$$

式中 $\lambda(k)$是遗忘因子,为简单可取常数 λ.

将 RPE 算法用于 DRNN,得 $\boldsymbol{\Psi}$ 阵的元素为

$$[\boldsymbol{\Psi}]_s=\frac{\mathrm{d}O}{\mathrm{d}W_s}=\begin{cases}Z_j(k), & W_s=W_j^o,\ 1\leqslant j\leqslant m\\ W_j^o\varphi'(S_j(k))\cdot Z_j(k-1), & W_s=W_j^d,\ 1\leqslant j\leqslant m\\ W_j^o\varphi'(S_j(k))\cdot X_j(k), & W_s=W_{ji}^h,\ 1\leqslant j\leqslant m,\ 1\leqslant i\leqslant n\end{cases} \tag{8}$$

式中 $s=1,2,\cdots n\times m+2m$ 是待求权的数目.

3.2 递归神经网络 RPE 学习算法的稳定收敛性分析

一般神经网络的稳定性主要取决于学习率和初始权值的选取. 如果学习率过小,则收敛速度慢;如果学习率过大,将出现振荡. 因此,适当地选择学习率非常重要. 下面利用 Lyapunov 定理推出保证控制系统稳定的学习率范围.

定理 1. 如果 DRNN 的 RPE 学习算法的学习率 $\mu(k)$满足 $0<\mu(k)<2$,则 RPE 算法保证 DRNN 在学习过程中稳定收敛.

证明. 定义 Lyapunov 函数为

$$V(k)=\frac{1}{2}\boldsymbol{e}^2(k)=\frac{1}{2}[O(k)-\hat{O}(k)]^2 \tag{9}$$

而

$$\Delta V(k)=V(k+1)-V(k),\ \boldsymbol{e}(k+1)=\boldsymbol{e}(k)+\Delta\boldsymbol{e}(k),\Delta\boldsymbol{e}(k)=\left[\frac{\partial\boldsymbol{e}(k)}{\partial W}\right]^{\mathrm{T}}\Delta W$$

对 DRNN 的 RPE 算法

$$\Delta W=\mu(\boldsymbol{\Psi}(k)\boldsymbol{\Psi}^{\mathrm{T}}(k))^{-1}\boldsymbol{\Psi}(k)\boldsymbol{e}(k) \tag{10}$$

其中 $\boldsymbol{\Psi}(k)=\dfrac{\partial\hat{O}(k)}{\partial W}=-\dfrac{\partial\boldsymbol{e}(k)}{\partial W}$,于是

$$\Delta V(k)=\frac{1}{2}[\boldsymbol{e}^2(k+1)-\boldsymbol{e}^2(k)]=\frac{1}{2}(-\mu\boldsymbol{e}(k))\cdot[2\boldsymbol{e}(k)-\mu\boldsymbol{e}(k)]=-\frac{1}{2}\mu(2-\mu)\boldsymbol{e}^2(k)$$

为了保证系统收敛,必须满足 $\Delta V(k)<0$. 于是 $\mu(2-\mu)>0$,即

$$0<\mu<2 \tag{11}$$

证毕.

3.3 改进的 RPE 算法

为进一步提高 RPE 算法的收敛特性，改进如下：

$$\boldsymbol{\delta}(k) = \alpha\boldsymbol{\delta}(k-1) + \mu\boldsymbol{\Psi}(k,\theta)\boldsymbol{e}(k,\boldsymbol{\theta}) \tag{12}$$

$$\boldsymbol{\theta}(k) = \boldsymbol{\theta}(k-1) + P(k) \cdot \boldsymbol{\delta}(k) \tag{13}$$

$$\lambda(k) = \lambda_0 \cdot \lambda(k-1) + (1-\lambda_0) \tag{14}$$

$$\bar{P}(k) = [I + M(k)\Psi^{\mathrm{T}}(k)] \cdot P(k-1) \tag{15}$$

$$P(k) = \frac{K_0}{\mathrm{tr}[\bar{P}(k)]} \cdot \bar{P}(k),\ K_0 > 0 \tag{16}$$

式中 $0<\alpha<1$ 为动量项参数，λ_0 为常数，$\mathrm{tr}[\cdot]$ 为矩阵迹的运算，K_0 为常数.

4 并行 RPE 算法

以上方法虽然可大大改进收敛特性，但也存在以下不足：一是增加了计算量，二是需要集中运算，而后者把神经网络并行结构的优点完全抹杀了，把神经元的并行运算变成了整个网络的集中运算. 这里我们把 RPE 算法改进为并行处理，把权值的修正计算分配到每个神经元上，为了与传统的 RPE 算法相区别，称之为并行 RPE 算法.

如前所述，定义参数矢量 $\boldsymbol{\theta}$ 和 $\boldsymbol{\Psi}(k,\theta)$. 假设整个神经网络中的神经元共有 n_p（输入神经元除外）个，而第 $i(i=1,\cdots,n_p)$ 个神经元的权，按顺序组成的参数矢量为 $\boldsymbol{\theta}_i$，其维数为 $n_{\boldsymbol{\theta}_i}(i=1,\cdots,n_p)$. 于是 $\boldsymbol{\theta}$ 和 $\boldsymbol{\Psi}(k,\theta)$ 可表示为

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_1,\cdots,\boldsymbol{\theta}_{n_p}]^{\mathrm{T}},\quad \boldsymbol{\Psi}(k,\theta) = [\boldsymbol{\Psi}_1(k,\theta),\cdots,\boldsymbol{\Psi}_{n_p}(k,\theta)]^{\mathrm{T}}$$

其中 $\boldsymbol{\Psi}_i(k,\theta)$ 是 $n_{\boldsymbol{\theta}_i}\times m$（$m$ 为神经网络输出维数）维矩阵，为网络的一步预报值对 $\boldsymbol{\theta}_i$ 的一阶微分$(i=1,\cdots,n_p)$. 选择准对角矩阵

$$\bar{H}(\theta) = \frac{1}{N}\sum_{k=1}^{N}\begin{bmatrix} \boldsymbol{\Psi}_1(k,\theta)\cdot\boldsymbol{\Psi}_1^{\mathrm{T}}(k,\theta) & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Psi}_2(k,\theta)\cdot\boldsymbol{\Psi}_2^{\mathrm{T}}(k,\theta) & \cdots & \vdots \\ \vdots & \cdots & \cdots & 0 \\ 0 & \cdots & 0 & \boldsymbol{\Psi}_{n_p}(k,\theta)\cdot\boldsymbol{\Psi}_{n_p}^{\mathrm{T}}(k,\theta) \end{bmatrix} \tag{17}$$

于是搜索方向变为

$$\boldsymbol{\sigma}(\boldsymbol{\theta}) = -[\bar{H}(\theta)]^{-1}\nabla\boldsymbol{E}(\boldsymbol{\theta})$$

$$= \begin{bmatrix} -\left[\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_1(k,\theta)\cdot\boldsymbol{\Psi}_1^{\mathrm{T}}(k,\theta)\cdot\right]^{-1}\cdot\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_1(k,\theta)\cdot\boldsymbol{e}(k,\boldsymbol{\theta}) \\ \vdots \\ -\left[\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_{n_p}(k,\theta)\cdot\boldsymbol{\Psi}_{n_p}^{\mathrm{T}}(k,\theta)\cdot\right]^{-1}\cdot\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_{n_p}(k,\theta)\cdot\boldsymbol{e}(k,\boldsymbol{\theta}) \end{bmatrix} \tag{18}$$

这样，搜索方向就可以按照神经元个数分成 n_p 个小的向量.

$$\boldsymbol{\sigma}_i(\boldsymbol{\theta}) = \left[\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_i(k,\theta)\cdot\boldsymbol{\Psi}_i(k,\theta)\cdot\right]^{-1}\cdot\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\Psi}_i(k,\theta)\cdot\boldsymbol{e}(k,\boldsymbol{\theta})\quad(i=1,\cdots,n_p) \tag{19}$$

于是,参数向量的修正也可以按照神经元来进行,具体算法为

$$\boldsymbol{\theta}_i(k)=\boldsymbol{\theta}_i(k-1)+\mu_i(k)\cdot\boldsymbol{\sigma}_i(\boldsymbol{\theta}(k-1))\quad(i=1,\cdots,n_p)\tag{20}$$

式(20)的每个子算式对应一个神经元,于是把神经网络的权值修正分配到每个神经元上了,式(17)是并行 RPE 算法的基础.

对于改进的 RPE 算法式(12)~(16)直接进行上述过程的推导就组成了并行 RPE 算法.改进 RPE 算法中 $P(k)$的元素数为 $n_{\boldsymbol{\theta}}^2$,而并行 RPE 算法中所有 $P_i(k)$ $(i=1,\cdots,n_p)$的总元素数为$\sum_{i=1}^{n_p}n_{\boldsymbol{\theta}_i}^2$,显然有 $n_{\boldsymbol{\theta}}^2=\left(\sum_{i=1}^{n_p}n_{\boldsymbol{\theta}_i}\right)^2\gg\sum_{i=1}^{n_p}n_{\boldsymbol{\theta}_i}^2$.所以说,并行 RPE 算法的计算只是改进 RPE 算法计算中的一小部分,而并行 RPE 算法的最大优点是:它象 BP 算法一样发挥了神经网络的并行结构的特点,一层中的所有神经元的权值调整可以同时进行.注意,并行 RPE 算法与 BP 算法相比,计算还是要复杂得多,但它的收敛特性要远远优于 BP 算法.在下一节的仿真中可得到证实.

5 仿真研究

5.1 非线性系统辨识

对于非线性系统辨识,假设系统模型为

$$y(k)=\frac{y(k-1)y(k-2)y(k-3)u(k-2)[y(k-3)-1]+u(k-1)}{[1+y^2(k-2)+y^2(k-3)]}\tag{21}$$

仿真中用到的 DRNN 结构为 DRNN(2, 7, 1),即输入有 2 个神经元,分别为 $u(k)$和 $y(k-1)$,隐层有 7 个神经元,输出有 1 个神经元,为 $y(k)$.初始权值为[-1.0, 1.0]内的随机数,学习率 $\mu=0.45$.仿真用计算机为 PⅡ400,仿真软件 Matlab6.1.

分别采用 DRNN 的 BP 学习算法、RPE 学习算法、改进的 RPE 算法和并行 RPE 算法对非线性系统式(21)进行辨识.取输入序列为 $u(k)=0.82\sin(2k\pi/100)+0.2\sin(2k\pi/25)$,具体的仿真结果,测试 10 次,取平均值.每种算法预定训练步数或误差要求,仿真结果如表 1 和表 2 所示.实际从理论上,仿真中的 DRNN 结构为 2-7-1, 共有权 $3\times7+7=28$ 个,RPE 算法中, $P(k)$的元素数为 $28^2=784$ 个,而并行 RPE 算法中所有 $P_i(k)$中的元素总数为 $7\times3^2+7^2=112$ 个, 可见, 并行 RPE 算法的计算量远远小于 RPE 算法.理论上,神经网络中的神经元数量越多,并行 RPE 算法的优点越突出,实际仿真中也充分证明了这一点.

从仿真结果看,RPE 算法可明显提高学习速度和模型精度,改进的 RPE 算法可进一步提高学习速度,但 RPE 算法所用时间远大于 BP 算法, 约是 BP 算法的 3 倍.与改进的 RPE 算法相比,并行 RPE 算法的学习速度和模型精度相差无几,但所用时间远少于改进的 RPE 算法.另外,所做的测试是在计算机上串行实现的,并没有体现出神经网络并行计算的优点,尤其是没有体现出所研究的并行算法的宗旨.对于 RPE 算法或改进的 RPE 算法, 因为运算本身就是集中运算, 与计算机是否串行实现无关.如果能实现神经网络的并行运算,每一层上的神经元计算能同时进行,并行 RPE 算法所用时间将会进一步地大大低于 RPE 算法或改进的 RPE 算法,具有非常好的实用价值.在神经网络的算法研究上, 这种并行结构的思想可广为推广.

表 1　仿真结果一

Table 1　No. 1 of the simulation result

学习算法	预训练步数	平均误差	平均需要时间(s)
原 BP 算法	5000	0.0183	46.4
RPE 算法	5000	0.0056	118.5
改进的 RPE 算法	5000	0.0012	192.6
并行 RPE 算法	5000	0.0013	58.3

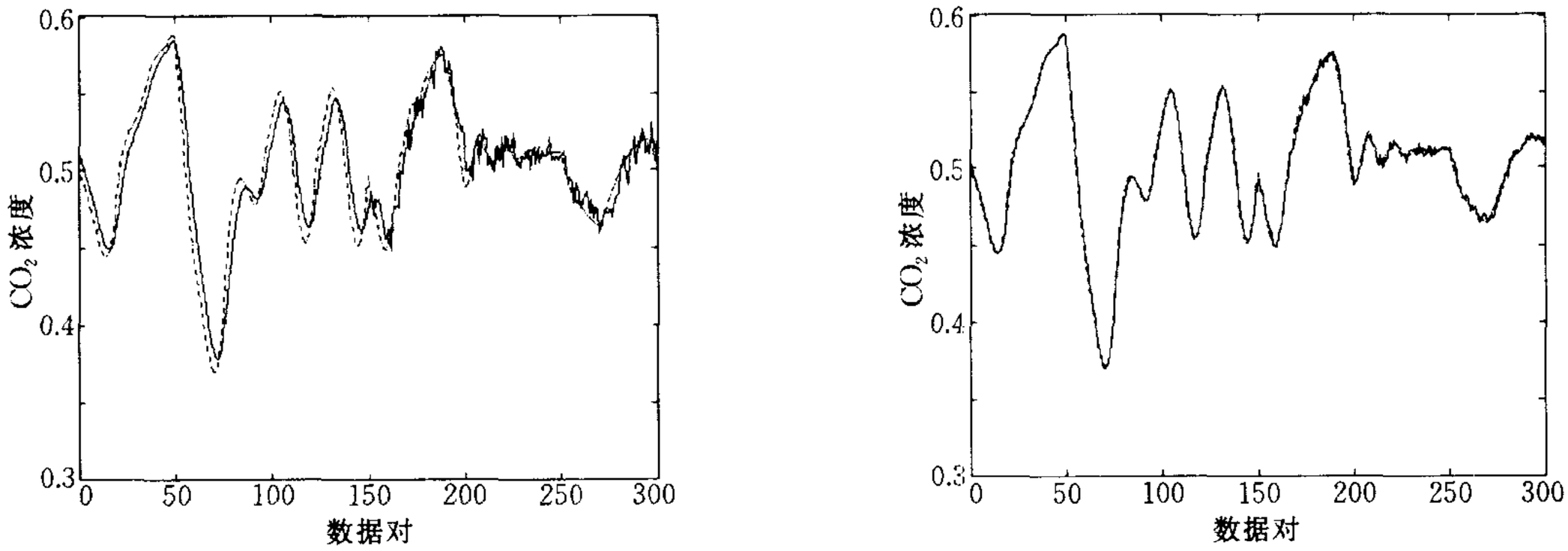
表 2　仿真结果二

Table 2　No. 2 of the simulation result

学习算法	要求误差	实际训练步数	平均需要时间(s)
原 BP 算法	≤0.001	26376	241.3
RPE 算法	≤0.001	7835	188.7
改进的 RPE 算法	≤0.001	5142	195.8
并行 RPE 算法	≤0.001	5153	59.6

5.2　实际过程数据建模

对实际过程建模，对象为某钢铁公司的燃油加热炉，输入是煤气流量，输出是 CO_2 的浓度. 取现场 300 对输入输出数据，进行正则化处理. 模型的动态结构未知，建模目的就是寻求输出量和输入量之间的非线性动态映射关系. 分别用并行 RPE 算法的 DRNN 和 BP 算法的 DRNN 进行对象的建模. 并行 RPE 算法预训练 500 步，BP 算法预训练 5000 步，训练样本取前 150 对输入输出数据，其余数据用于对所建模型进行校验，结果如图 1 所示，其中点线为系统输出的实际观测值，实线为网络的训练结果.



(a) DRNN BP 算法建模结果　　(b) DRNN 并行 RPE 算法建模结果

图 1　DRNN 实际过程建模结果

Fig. 1　The modelling results of actual process

从仿真结果可见，并行 RPE 算法取得了很好的建模结果，其建模精度明显优于 BP 算法.

6　结论

对递归神经网络的学习算法进行了研究. 首先提出了 RPE 学习算法，并且针对 RPE 学习算法的不足进行了改进，改进的 RPE 学习算法具有收敛速度快的优点，但计算量较大. 然

后提出了并行 RPE 学习算法，在收敛性变化不大的情况下，大大降低了计算量，最重要的是此算法变集中运算为分布运算，完全符合神经网络并行结构的特点，利于硬件实现，具有非常好的实用价值. 在神经网络的算法研究上，这种并行结构的思想应该广为推广. 仿真结果证明了所提出算法的有效性.

References

1 Chen S, Billings S A. Neural networks for non-linear dynamic system modelling and identification. *International Journal of Control*, 1992, **56**(2): 319～346

2 Noriega J R, Wang H. A direct adaptive neural-network control for unknown nonlinear systems and its application. *IEEE Transactions on Neural Networks*, 1998, **9**(1): 27～34

3 Rovithakis G A. Robustifying nonlinear systems using high-order neural network controllers. *IEEE Transactions on Automatic Control*, 1999, **44**(1): 102～108

4 Files C L, Kim J. Dynamic recurrent neural networks: Theory and applications. *IEEE Transactions on Neural Networks*, 1994, **5**(2): 153～155

5 Kremer S C. Identification of a specific limitation on local-feedback recurrent networks acting as Mealy-Moore machines. *IEEE Transactions on Neural Networks*, 1999, **10**(2): 433～438

6 Chow W S, Fang Y. A recurrent neural- network-based real-time learning control strategy applying to nonlinear systems with unknown dynamics. *IEEE Transactions on Neural Network*, 1998, **45**(1): 151～161

7 Liang J Z, He X G, Huang D S. Super-linearly convergent BP learning algorithm for feedforward neural networks. *Journal of Software*, 2000, **11**(8): 1094～1096 (in Chinese)

8 Pineda F J. Generalization of back-propagation to recurrent neural networks. *Physical Revise Letter*, 1987, **59**: 2229～2232

9 Wei W. A new on-line recursive learning algorithm for recurrent neural network. *Acta Automatica Sinica*, 1998, **24**(5): 616～621(in Chinese)

10 Jin L, Gupta M M. Stable dynamic backpropagation learning in recurrent neural networks. *IEEE Transactions on Neural Networks*, 1999, **10**(6): 1321～1334

李鸿儒 博士，副教授. 主要研究方向为神经网络、模糊控制、复杂工业过程建模与控制.

(**LI Hong-Ru** Ph. D, associate professor. His interests include neural network, Fuzzy control, Modelling and control of complex industrial process.)

顾树生 教授，博士生导师，主要研究方向为智能控制理论及其应用.

(**GU Shu-Sheng** Professor. His interests include intelligent control theory and its application.)