

Pi-sigma 神经网络的乘子法随机 单点在线梯度算法 *

喻 昕, 邓 飞, 唐利霞

(广西大学 计算机与电子信息学院, 南宁 530004)

摘 要: 在利用梯度算法训练 Pi-sigma 神经网络时, 存在因权值选取过小导致收敛速度过慢的问题, 而采用一般罚函数法虽然可以克服这个缺点, 但要求罚因子必须趋近于 ∞ 且惩罚项绝对值不可微, 从而导致数值求解困难。为克服以上缺点, 提出了一种基于乘子法的随机单点在线梯度算法。利用最优化理论方法, 将有约束问题转换为无约束问题, 利用乘子法来求解网络误差函数。从理论上分析了算法的收敛速度和稳定性, 仿真实验结果验证了算法的有效性。

关键词: Pi-sigma 神经网络; 梯度算法; 乘子法; 收敛速度; 稳定性

中图分类号: TP183; TP301.6

文献标志码: A

文章编号: 1001-3695(2011)11-4074-04

doi:10.3969/j.issn.1001-3695.2011.11.019

Training Pi-sigma neural network by stochastic simple point online gradient algorithm with Lagrange multiplier method

YU Xin, DENG Fei, TANG Li-xia

(School of Computer & Electronics Information, Guangxi University, Nanning 530004, China)

Abstract: When the on-line gradient algorithm is used for training Pi-sigma neural network, there is a problem that the chosen weights may be very small, resulting in a very slow convergence. The shortcoming can be overcome by the penalty method, but there are the difficulties in numerical solution, caused by the facts that the penalty factor must approach infinity and the absolute value of penalty term is nondifferentiable. Based on Lagrange multiplier algorithm, this paper proposed a stochastic simple point on-line gradient algorithm to overcome the deficiencies of small weights and penalty function. Using the optimized theory method, transformed the restrained question into the non-constraint question. Proved the convergence rate and stability of the algorithm. The simulated experimental results indicate that the algorithm is efficient.

Key words: Pi-sigma neural network; gradient algorithm; Lagrange multiplier method; convergence rate; stability

神经网络在实际的商业、金融、文学等领域得到了广泛应用^[1,2]。1987 年 Giles 和 Maxwell 提出了高阶神经网络 (high-order neural networks, HONN)^[3], 这种网络既可以加强多层前馈型神经网络的非线性分类能力, 又可以将训练时间大大减少。常见的 HONN 有 sigma-pi 神经网络、Pi-sigma 神经网络、ridge polynomial 神经网络、递归 Pi-sigma 神经网络和 sigma-pi-sigma 神经网络等。

Pi-sigma 神经网络 (Pi-sigma neural network, PSNN)^[4] 是从高阶神经网络演化而来的。Pi-sigma 神经网络保持了多层 HONN 强大的学习能力, 避免了权值数目随输入维数增加的组合性增长。此网络已被广泛应用于可视化密码技术方案设计^[5]、机械控制^[6]、杂交的遗传学习算法^[7] 以及股票交易^[8] 等。

梯度算法分为在线梯度算法和批处理梯度算法, 两种算法均被广泛地应用于高阶前馈神经网络^[9~11], 但存在因初始权值选取不当导致的收敛速度过慢的问题。针对这个问题, 文献[12]给出了在 Pi-sigma 神经网络中采用在线梯度算法时初

始权值与权值改变量以及网络误差改变量的关系式, 并指出初始权值选取过小会导致收敛速度降低, 初始权值选取过大, 则会在初始值附近陷入局部最小。为了克服初始权值选取困难的问题, 文献[13]提出了基于罚函数的在线梯度方法, 但是在使用罚函数时, 罚因子 M 必须趋于 ∞ , 并且罚函数对约束不好处理, 一般用它们的绝对值作为惩罚项, 但是绝对值不可微, 可能导致 Pi-sigma 神经网络求解过程的数值困难。

研究者对将乘子法应用到优化神经网络中已展开了一些研究^[14,15], 因此, 针对文献[13]中的罚函数法的缺点, 本文将乘子法应用到随机单点在线梯度算法中, 用来训练 Pi-sigma 神经网络, 有效地克服了罚函数法出现的病态问题。

1 Pi-sigma 神经网络乘子法随机单点在线梯度算法

1.1 随机单点在线梯度算法

PSNN 由一个输入层、一个隐含层 (求和层) 和一个输出层

收稿日期: 2011-04-17; 修回日期: 2011-05-20 基金项目: 国家自然科学基金资助项目 (60763013); 广西人才小高地创新团队计划资助项目 (桂教人[2007]71 号); 广西大学科研基金资助项目 (X081017)

作者简介: 喻昕 (1973-), 男, 重庆人, 副教授, 博士, 主要研究方向为人工神经网络、智能计算、互联网; 邓飞 (1986-), 男, 云南泸西人, 硕士研究生, 主要研究方向为人工神经网络 (feigoffice@163.com); 唐利霞 (1987-), 女, 湖南永州人, 硕士研究生, 主要研究方向为信息安全、人工智能。

(求积层)组成,假设输入层、隐含层和输出层的神经元个数分别为 N, K 和 1,如图 1 所示。输入样本 $X = (x_1, x_2, \dots, x_N)^T \in R^N$, 其中 $x_N = -1$ 是对应的阈值,相应的实际输出为 $y \in R$, 理想输出为 $O \in R$; w_{kj} 为第 k 个输入点与第 j 个求和层节点间的权值, $w_j = (w_{1j}, w_{2j}, \dots, w_{Nj})$ 为输入层各节点与求和层 j 节点的权值向量,其中 $w_{Nj} = 1$, 则求和层 h_j 的为

$$h_j = \sum_{k=1}^N w_{kj} x_k = \sum_{k=1}^{N-1} w_{kj} x_k - 1 = w_j X \quad (1)$$

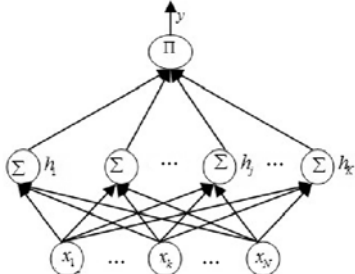


图1 Pi-sigma神经网络

设激活函数为 $f(x)$, 则对于样本集 $(y^j, O^j)_{j=1}^J$, 网络实际输出为

$$y^j = f(\sum_{j=1}^K h_j) = f(\sum_{i=1}^K w_i x^j) \quad (2)$$

网络误差函数取为传统的平方误差函数:

$$\Delta E(w) = \frac{1}{2} \sum_{j=1}^J (O^j - y^j)^2 = \frac{1}{2} \sum_{j=1}^J (O^j - f(\sum_{i=1}^K w_i x^j))^2 \quad (3)$$

随机单点在线梯度算法是 Pi-sigma 神经网络以及以此为模块的网络常用的训练算法^[4,7]。用于训练 PSNN 的随机单点在线梯度算法是每随机输入一个样本时, 仅修改部分权值, 即修改一个求和单元对应的权值。

在使用梯度算法时, $\Delta E(w)$ 关于权向量 w_n 的梯度为

$$\begin{aligned} \nabla [\Delta E(w_n)] &= - \sum_{j=1}^J (O^j - f(\sum_{i=1}^K w_i x^j)) \times [f'(\sum_{i=1}^K w_i x^j)]' = \\ &= - \sum_{j=1}^J (O^j - f(\sum_{i=1}^K w_i x^j)) \times f'(\sum_{i=1}^K w_i x^j) \times (\sum_{i=1, i \neq n}^K w_i x^j) \times x^j \end{aligned} \quad (4)$$

对于权值 $w^{(0)}$, 假设前 $m-1$ 次迭代已经完成, 第 m 次迭代随机选取的样本和需要调整权值的求和层节点分别为 x^j 和 n , 则权值更新为

$$\begin{cases} w_n^{(m+1)} = w_n^{(m)} + \Delta_j w_n^{(m)} & q = n \\ w_q^{(m+1)} = w_q^{(m)} & q \neq n \end{cases} \quad (5)$$

其中: 对于 $w_n^{(m+1)}$, 因上标是从 0 开始, 所以第 n 次迭代要加 1, q 表示第 q 个求和层节点。式(5)保证了只有在 $q = n$ 时才修改权值, 一次只修改一个, 即随机单点算法。 $\Delta_j w_n^{(m)}$ 为权值改变量

$$\begin{aligned} \Delta_j w_n^{(m)} &= \eta \times (O^j - f(\sum_{i=1}^K w_i x^j)) \times f'(\sum_{i=1}^K w_i x^j) \times \\ &\quad (\sum_{i=1, i \neq n}^K w_i x^j) \times x^j \end{aligned} \quad (6)$$

其中: η 为学习率。

可以看出, 权值改变量 $\|\Delta_j w_n^{(m)}\|$ 与误差改变量 $|\Delta E(w)|$ 都受乘积项 $\prod_{i=1}^K w_i x^j$ 的影响很大。这里指出权值较小对随机单点在线梯度算法的影响^[12]:

$$\|\Delta_j w_n^{(m)}\| \leq c \times \eta \times \delta_m^{K-1} \quad (7)$$

$$|\Delta E(w^{(m)})| \leq c \times \eta \times \delta_m^{2(K-1)} \quad (8)$$

其中: $\delta_m = \max_{1 \leq n \leq K} \|w_n^{(m)}\|$ 表示第 m 次迭代时各个求和层节点的最大输出模值。 δ_m^{K-1} 和 $\delta_m^{2(K-1)}$ 分别表示 δ_m 的 $K-1$ 和 $2(K-1)$ 次方。

1) 次方。

由式(7)和(8)可看出, $\|\Delta_j w_n^{(m)}\|$ 与 δ_m^{K-1} 同阶, $|\Delta E(w^{(m)})|$ 与 $\delta_m^{2(K-1)}$ 同阶, 当 $\delta_m \ll 1$ 时, 就会出现新旧权值变化不大, 网络误差改变量微小, 收敛性很差。

1.2 基于乘子法的随机单点在线梯度算法

对 PSNN 训练的最终目的是为了求 w^* , 使

$$\Delta E(w^*) = \min_w \Delta E(w)$$

实质就是求解无约束优化问题。

a) 把求解 $\min_w \Delta E(w)$ 转换为对乘积因子 $\prod_{i=1}^K w_i x^j$ 加以限制的约束优化问题(因为 $\Delta E(w)$ 的大小主要由乘积项 $\prod_{i=1}^K w_i x^j$ 决定)。

$$\begin{cases} \min \Delta E(w) \\ \text{s. t. } |\prod_{i=1}^K w_i x^j| > \alpha \end{cases} \quad (9)$$

其中: α 取适当常数(取值不宜太大), α 表示当 $\prod_{i=1}^K w_i x^j$ 小到什么程度时加入惩罚。

b) 把有约束问题转换为无约束问题, 令 $g_i(w) = |\prod_{i=1}^K w_i x^j| - \alpha$, 假设存在一 c_j^2 使 $g_j(w) - c_j^2 = 0$ 。

于是乘子法函数^[16]可定义为

$$\begin{aligned} \varphi_1(w, c, \gamma, M) &= \Delta E(w) - \sum_{j=1}^J \gamma_j (g_j(w) - c_j^2) + \\ &\quad \frac{M}{2} \sum_{j=1}^J (g_j(w) - c_j^2)^2 \end{aligned} \quad (10)$$

其中, γ 为不等式约束的拉格朗日乘子向量, $\gamma = (\gamma_1, \gamma_2, \dots, \gamma_J)^T$, M 为罚因子。因此式(9)转换为求解 $\min \varphi_1(w, c, \gamma, M)$ 。

用配方法将 $\min \varphi_1(w, c, \gamma, M)$ 化为

$$\begin{aligned} \varphi_1(w, c, \gamma, M) &= \Delta E(w) + \sum_{j=1}^J [-\gamma_j (g_j(w) - c_j^2) + \\ &\quad \frac{M}{2} (g_j(w) - c_j^2)^2] = \Delta E(w) + \\ &\quad \sum_{j=1}^J \{ \frac{M}{2} [c_j^2 - \frac{1}{M} (M g_j(w) - \gamma_j)]^2 - \frac{\gamma_j^2}{2M} \} \end{aligned} \quad (11)$$

令 $\frac{\partial \varphi_1}{\partial c_j} = \partial \frac{M}{2} [c_j^2 - \frac{1}{M} (M g_j(w) - \gamma_j)] \times 2 c_j =$

$$2 M c_j [c_j^2 - \frac{1}{M} (M g_j(w) - \gamma_j)] = 0 \quad (12)$$

则 $\varphi_1(w, c, \gamma, M)$ 对于 c_j 取极小时, c_j 取值如下:

当 $M g_j(w) - \gamma_j \geq 0$ 时, $c_j^2 = \frac{1}{M} (M g_j(w) - \gamma_j)$;

当 $M g_j(w) - \gamma_j < 0$ 时, $c_j^2 = 0$ 。

两种情况统一为一个表达式:

$$c_j^2 = \frac{1}{M} \max(0, (M g_j(w) - \gamma_j)) \quad (13)$$

将式(13)代入(11)得到不等式约束类问题的乘子罚函数为

$$\begin{aligned} \varphi_1(w, \gamma, M) &= \Delta E(w) + \\ &\quad \frac{1}{2M} \sum_{j=1}^J \{ [\max(0, (\gamma_j - M g_j(w)))]^2 - \gamma_j^2 \} \end{aligned} \quad (14)$$

原问题就转换为求解乘子罚函数 $\varphi(w, \gamma, M)$ 的无约束极小问题。

在求解迭代过程中, 给定一个大的罚因子 M (不必趋于无穷大) 和给定不等式的拉格朗日乘子向量的初始估计 $\gamma^{(1)}$, 然后通过修正第 k 次迭代中的 $\gamma^{(k)}$ 得到第 $k+1$ 次迭代 $\gamma^{(k+1)}$, 修正如下:

$$\gamma_j^{k+1} = \max(0, (\gamma_j^{(k)} - M g_j(w^{(k)}))) \quad j = 1, 2, \dots, J \quad (15)$$

最后目标函数即网络误差函数为

$$\min_w \Delta E(w) = \Delta E(w) + \frac{1}{2M_j} \sum_{j=1}^J \{ [\max(0, (\gamma_j - M g_j(w)))]^2 - \gamma_j^2 \} \quad (16)$$

利用随机单点在线梯度算法求解该问题,式(16)关于权向量 $w_n(n=1,2,\dots,k)$ 的梯度为

$$\nabla [\Delta E(w)] = - \sum_{j=1}^J (O^j - f'(\sum_{i=1}^K w_i x_i^j)) \times f'(\sum_{i=1}^K w_i x_i^j) \times (\sum_{i=1, i \neq n}^K w_i x_i^j x^j + \frac{1}{2M_j} \sum_{j=1}^J \nabla p_j(w)) \quad (17)$$

其中:

$$\begin{aligned} p_j(w) &= [\max(0, (\gamma_j - M g_j(w)))]^2 - \gamma_j^2 \\ \nabla p_j(w) &= \begin{cases} 2(\gamma_j - M g_j(w)) \times (-M) \times \nabla g_j(w) & \gamma_j - M g_j(w) \geq 0 \\ 0 & \gamma_j - M g_j(w) < 0 \end{cases} \\ g_j(w) &= \begin{cases} |\sum_{i=1}^K w_i x_i^j| - \alpha & -\alpha < \sum_{i=1}^K w_i x_i^j < \alpha \\ 0 & \text{其他} \end{cases} \\ \nabla g_j(w) &= \begin{cases} (\sum_{i=1, i \neq n}^K w_i x_i^j) x^j & 0 < \sum_{i=1}^K w_i x_i^j < \alpha \\ -(\sum_{i=1, i \neq n}^K w_i x_i^j) x^j & -\alpha < \sum_{i=1}^K w_i x_i^j < 0 \\ 0 & \text{其他} \end{cases} \end{aligned}$$

则对应于式(5)和(6)的权值更新式为

$$\begin{cases} w_n^{(m+1)} = w_n^{(m)} + \Delta_j w_n^{(m)} & q = n \\ w_q^{(m+1)} = w_q^{(m)} & q \neq n \end{cases} \quad (18)$$

其中: $\Delta_j w_n^{(m)} = \eta \times [(O^j - f'(\sum_{i=1}^K w_i x_i^j)) \times f'(\sum_{i=1}^K w_i x_i^j) \times (\sum_{i=1, i \neq n}^K w_i x_i^j x^j - \frac{1}{2M} \nabla p_j(w))]$

由式(9)可以看出, α 表示当乘积 $\sum_{i=1}^K w_i x_i^j$ 项小到什么程度时,加入乘子罚函数, α 不宜过大。

2 收敛速度和稳定性分析

以上讨论了 Pi-sigma 神经网络基于乘子法的随机单点在线梯度算法,本章将讨论相应算法的收敛速度和稳定性,在此之前提出以下假设:

假设 1 $w_k \rightarrow w^*$. w^* 为问题式(9)的解。

假设 2 $\nabla g(w_k)^T$ 为列满秩。

注:文献[16]中已经证明了假设 1;解决优化问题时,只要保证式(9)中所有约束条件都线性无关,就可保证 $\nabla g(w_k)^T$ 为列满秩。

2.1 收敛性分析

定理 若在假设 1、2 成立的情况下,基于乘子法的随机单点在线梯度算法的收敛速度是线性收敛的。

证明: 对式(16)求关于 w_{k+1} 的一阶梯度,有 $\nabla [\Delta E(w_{k+1})] = 0$, 记 $A(w_k) = \nabla g(w_k)^T$, 即

$$A(w_{k+1}) \gamma(w_{k+1}) = \nabla [\Delta E(w_{k+1})] \quad (19)$$

所以由算法中式(16)所产生的 γ^{k+1} 和式(19)所定义的 $\gamma(w_{k+1})$ 是完全等价的。由假设 2 和式(19)可知

$$\gamma(w) = A(w)^+ \cdot \nabla [\Delta E(w_{k+1})] \quad (20)$$

又由式(20)可推导

$$\gamma(w)^T = \nabla [\Delta E(w_{k+1})]^T [A(w)^+]^T$$

$$\begin{aligned} \nabla \gamma(w)^T &= \nabla^2 [\Delta E(w_{k+1})]^T \times [A(w)^+]^T + \\ &\nabla [\Delta E(w_{k+1})]^T \times \nabla [A(w)^+]^T \end{aligned} \quad (21)$$

由此算法中式(15)可知:

$$\gamma(w_{k+1}) + M g(w_{k+1}) = \gamma(w_k) \quad (22)$$

对式(22)使用泰勒公式展开,可得

$$\gamma(w^*) + \nabla \gamma(w^*)^T \times (w_{k+1} - w^*) + M [g(w^*) + \nabla g(w^*)^T \times (w_{k+1} - w^*)] \approx \gamma(w^*) + \nabla \gamma(w^*)^T \times (w_k - w^*)$$

又因 $g(w^*) = 0$, 可推得

$$\begin{aligned} [\nabla \gamma(w^*)^T + M \times \nabla g(w^*)^T] (w_{k+1} - w^*) \approx \\ \nabla \gamma(w^*)^T \times (w_k - w^*) \end{aligned} \quad (23)$$

由式(23)可得

$$\begin{aligned} [\nabla \gamma(w^*)^T + M \times A(w^*)] (w_{k+1} - w^*) \approx \\ \nabla \gamma(w^*)^T \times (w_k - w^*) \end{aligned} \quad (24)$$

其中: $\nabla \gamma(w^*)^T = \nabla^2 [\Delta E(w^*)]^T \times [A(w^*)^+]^T + \nabla [\Delta E(w^*)]^T \times \nabla [A(w^*)^+]^T$ 。

由式(24)可知,除非 M 趋于 ∞ , 否则结论成立(乘法中 M 不必趋于 ∞), 即该算法的收敛速度为线性收敛的。

2.2 稳定性分析

稳定性是本文中 Pi-sigma 神经网络能否应用于优化计算的关键问题,在对稳定性进行分析之前,先给出一个记号:式(10)中的 c_j^2 可由式(13)代替, M 为常值,所以可以记式(10)为 $\varphi_1(w, \gamma)$ 。根据经典极值理论^[17],给出动力学系统方程:

$$\begin{cases} \frac{dw}{dt} = - \cdot w \varphi_1(w, \gamma) \\ \frac{d\gamma}{dt} = - \cdot w \varphi_1(w, \gamma) \end{cases} \quad (25)$$

接着定义网络的能量函数^[18]:

$$E(w, \gamma) = \frac{1}{2} \| \cdot w \varphi_1(w, \gamma) \|^2 + \frac{1}{2} \| \cdot \gamma \varphi_1(w, \gamma) \|^2 \quad (26)$$

其中: $\| \cdot \|$ 表示欧式向量的模。

下面分析本文中神经网络的稳定性,将 $E(w, \gamma)$ 对时间求导:

$$\begin{aligned} \frac{dE}{dt} &= \frac{dE}{dw} \frac{dw}{dt} + \frac{dE}{d\gamma} \frac{d\gamma}{dt} = \\ &((\cdot w \varphi_1(w, \gamma))^T G_1 + (\cdot \gamma \varphi_1(w, \gamma))^T \cdot w^2 \varphi_1(w, \gamma)) \frac{dw}{dt} + \\ &((\cdot w \varphi_1(w, \gamma))^T \cdot w \gamma^2 \varphi_1(w, \gamma) + (\cdot \gamma \varphi_1(w, \gamma))^T G_2) \frac{d\gamma}{dt} \end{aligned} \quad (27)$$

其中, G_1, G_2 都是与 $g_1(w), g_2(w), \dots, g_j(w)$ 有关的对角正定矩阵^[18]。

将式(25)代入式(27)后得

$$\begin{aligned} \frac{dE}{dt} &= -(\cdot w \varphi_1(w, \gamma))^T G_1 \cdot w \varphi_1(w, \gamma) - \\ &(\cdot \gamma \varphi_1(w, \gamma))^T G_2 \cdot \gamma \varphi_1(w, \gamma) \end{aligned} \quad (28)$$

由 G_1, G_2 为对角正定矩阵可得

$$\frac{dE}{dt} \leq 0 \quad (29)$$

当且仅当 $\cdot w \varphi_1(w, \gamma) = 0, \cdot \gamma \varphi_1(w, \gamma) = 0$ 时, $\frac{dE}{dt} = 0$ 成立(此时 $w = w^*$)。因此本文的 Pi-sigma 神经网络是稳定的。

3 仿真实验及分析

实验应用 MATLAB 作为工具,以最基本的奇偶分类为例:
输入模式: $[1, 1; 1, -1; -1, 1; -1, -1]$;
输出模式: $[0; 1; 1; 0]$ 。

选择的网络为单输出的 PSNN,输入层、求和层和求积层的节点数分别为 3、2、1,激活函数取为经典的 sigmoid 函数($1/(1 + e^{-x})$),初始权值在 $[-1, 1]$ 间随机选取,学习速率取为 0.3。算法中的参数分别为: $M = 2$ 、 $\alpha = 0.25$ 、 γ 初始值取为 1 矩阵、 $\alpha = 3$ 。算法终止的条件:误差精度为 10^{-3} ;最大迭代次数为 5 000。当满足其中一个条件时,算法就终止。

由于初始权值是随机选取的,所以对问题进行训练时,每次训练的结果都不相同,但训练的最终结果大概趋势差别不会太大。针对无乘子法和基于乘子法两种情况进行训练,其实验结果如图 2 所示。

由图 2 可以看出,当达到相同误差精度时,基于乘子法的随机单点在线梯度算法的迭代次数比无乘子法算法的迭代次数明显降低,说明了此算法的有效性。从图 2(b)中可以看出曲线基本上成一条直线,近似于线性关系,也验证了第 2 章中的收敛速度的结论。

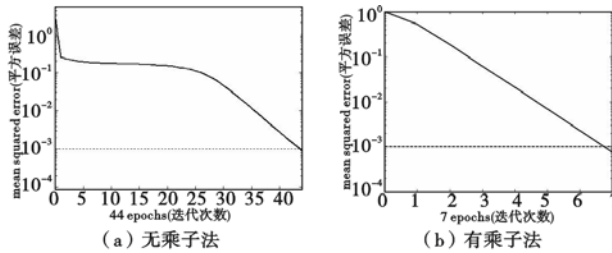


图2 无乘子法和基于乘子法的随机单点在线梯度算法结果比较

4 结束语

使用基于乘子法的随机单点在线梯度算法来训练 Pi-sigma 神经网络,避免了使用罚函数所带来的数值计算困难的病态问题。实验结果表明,该算法相比一般的随机单点在线梯度算法训练 Pi-sigma 神经网络,具有很好的收敛速度和稳定性。

参考文献:

[1] 阎平凡,张长水.人工神经网络与模拟进化计算[M].2版.北京:清华大学出版社,2005.

[2] EPITROPAKIS M G, PLAGIANAKOS V P, VRAHATIS M N. Hardware-friendly higher-order neural network training using distributed evolutionary algorithms[J]. *Applied Soft Computing*, 2010 (10): 398-408.

[3] SHIN Y, GHOSH J. The Pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation [C]//Proc of International Joint Conference on Neural Networks. 1991:13-18.

[4] GILES C, MAXWELL T. Learning, invariance and generalization in high-order neural networks[J]. *Applied Optics*, 1987, 26 (23): 49-78.

[5] SONG Ge, PENG Chang-gen, MIAO Xue-lan. Visual cryptography scheme using Pi-sigma neural networks [C]//Proc of International Symposium on Information Science and Engineering. Washington DC: IEEE Computer Society, 2008: 679-682.

[6] HACIB T, ACIKGOZ H, Le BIHAN Y, et al. Microwave characterization using ridge polynomial neural network and least-square support vector machines[J]. *IEEE Trans on Magnetics*, 2011, 47 (5): 990-993.

[7] NIE Yong, DENG Wei. A hybrid genetic learning algorithm for Pi-sigma neural network and the analysis of its convergence [C]//Proc of the 4th International Conference on Natural Computation. Washington DC: IEEE Computer Society, 2008: 19-23.

[8] GHAZALI R, HUSSAIN A J, LIATIS P. Dynamic ridge polynomial neural network: forecasting the univariate non-stationary and stationary trading signals[J]. *Expert Systems with Applications*, 2011, 38 (4): 3765-3776.

[9] 徐昕,贺汉根.神经网络增强学习的梯度算法研究[J]. *计算机学报*, 2003, 26 (2): 227-233.

[10] ZHANG Chao, WU Wei, CHEN Xian-hua, et al. Convergence of BP algorithm for product unit neural network with exponential weights [J]. *Neurocomputing*, 2008, 72 (1-3): 513-520.

[11] ZHANG Hui-sheng, WU Wei. Boundedness and convergence of on-line gradient method with penalty linear output feedforward neural networks[J]. *Neural Processing Letters*, 2009, 29 (3): 205-212.

[12] 熊焱. Pi-sigma 神经网络的几种梯度学习算法 [D]. 大连: 大连理工大学, 2007.

[13] YAN Xiong, WU Wei, KANG Xi-dai. Training Pi-sigma network by online gradient algorithm with penalty for small weight update [J]. *Neural Computation*, 2007, 19 (12): 1-13.

[14] 黄远灿, 孙圣和, 韩京清. 基于 Lagrange 乘子法的非线性规划神经网络[J]. *电子学报*, 1998, 26 (1): 24-28.

[15] 李海滨, 王亮, 尚凡华, 等. 基于 Lagrange 乘子法神经网络求解弹塑性力学有限元问题[J]. *重庆工学院学报*, 2007, 21 (3): 16-19.

[16] 袁亚湘, 孙文瑜. 最优化理论与方法 [M]. 北京: 科学出版社, 1997.

[17] BAZARAA M S, SHELLEY C. Nonlinear programming theory and algorithms [M]. New York: Wiley, 1979.

[18] EFFATI S, BAYMANI M. A new nonlinear neural network for solving convex nonlinear programming problems [J]. *Applied Mathematics and Computation*, 2005 (168): 1370-1379.

(上接第 4070 页)

[8] ANKERST M, BREUNIG M, KRIEDEL H P. OPTICS: ordering points to identify the clustering structure [C]//Proc of ACM-SIGMOD International Conference on Management of Data. New York: ACM Press, 1999: 49-60.

[9] HINNEBURG A, KEIM D A. An efficient approach to clustering in large multimedia databases with noise [C]//Proc of the 4th International Conference on Knowledge Discovery and Data Mining. 1998: 58-65.

[10] NERENBERG M A H, ESSEX C. Correlation dimension and systematic geometric effects [J]. *Physical Review A*, 1999, 42 (12): 7065-7074.

[11] ROSENSTEIN M T, COLLINS J J, DeLUCA C J. A practical method for calculating largest Lyapunov exponents from small datasets [J]. *Physica D*, 1993, 65 (1-2): 117-134.

[12] SCOTT D W. Multivariate density estimation: theory, practice, and visualization [M]. Hoboken: Wiley, 1992.

[13] KLEINBERG J. An impossibility theorem for clustering [C]//Advances in Neural Information Processing Systems. Cambridge: MIT Press, 2002: 446-453.

[14] HALKIDI M, VAZIRGIANNIS M. A dataset oriented approach for clustering algorithm selection [C]//Proc of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases. London: Springer-Verlag, 2009: 165-179.