BP神经网络的C++实现及讨论

邓 静,马传松,李振坤
(广东工业大学 计算机学院,广东 广州 510090)

摘 要:通过C++构造出一个五层的BP神经网络,在满足相对误差要求的情况下,实现了指定样本函数的功能。并针对学习效率和权系数修正常数对算法做了改进,有效地加快了收敛速度。最后讨论了当样本函数中 Y 值非[0,1]区间时,样本的归一化问题。

关键词:BP神经网络;样本;权系数;学习效率;归一化
中图分类号:TP183 **文献标识码**:A **文章编号**:1005-3751(2003)07-0093-04

Discussion and Implementation for BP Neural Network with C++

DENG Jing, MA Chuan-song, LI Zhen-kun
(Sch. of Comp. Sci., Guangdong Univ. of Techn., Guangzhou 510090, China)

Abstract: Constitute a sketch of five layers BP neural network which fulfils the assigned pattern function while the relative error is satisfied. Improves the learning rate and weight coefficient of the arithmetic. The improvements also bring faster convergence speed. At last discuss the normalizati on about pattern when the value of Y is out of [0,1].

Key words: BP neural networks; pattern; weight coefficient; learning rate; normalization

0 引 言

BP(Back Propagation)神经网络又称为多层前馈神经网络,是人工神经网络中比较成熟的一种方法,使用无反馈的多层前向网络拓扑结构,BP算法也称反向传播算法。这种神经网络模型的特点是:各层神经元仅与相邻层神经元之间有连接;各层内神经元之间无任何连接;各层神经元之间无反馈连接[1]。

1 BP神经网络模型及其C++实现

1.1 BP神经网络模型

BP神经网络由多个网络层构成,其中包括一个输入层、一个或几个中间层(隐层)、一个输出层,相邻两层各神经元相互联系,同层神经元之间不存在相互连接[2]。文中采用1-4-4-3-1,五层模型(如图1所示)。BP网络的学习过程由前向传播和反向传播组成,在前向传播过程中,输入模式经输入层、隐层逐层处理,并传向输出层,如果在输出层不能得到期望的输出,则转入反向传播过程,将误差值沿连接通路逐层反向传送,并修正各层连接权值。对于给定的一组训练模式,不断用一个训练模式训练网络,重复前向传播和误差反向传播过程,直至误差函数 e 小于给定值为止[1]。前向传播的过程按③式进行,反向传播过程按④、⑤进行。

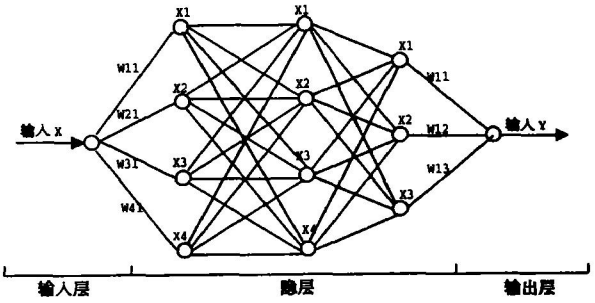


图1 BP神经网络拓扑结构图

每个神经元数学模型[3]如图2所示。

其中:$x_1, x_2, \cdots, x_n$ 是神经元的输入

θ_i 是 i 神经元的阈值

$w_1, w_2, \cdots, w_n$ 是权系数

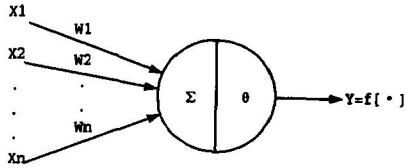


图2 神经元模型

① 样本函数:$Y = 3X^2 - 6X + 7 \quad X \in [-5,5]$

② 激发函数:$f(x) = 1/(1 + \exp(-x))$

③ 各层输出:$U_i^k = \sum W_{ij} X_i^{k-1}, X_{n+1}^{k-1} = 1$,

收稿日期:2003-02-21
作者简介:邓 静(1979—),女,四川人,硕士研究生,研究方向为网络与分布式系统。

$W_{i,j+1} = -\theta \quad X_i^k = f(U_i^k)$

X_i^k— 节点输出；X_i^{k-1}— 节点输入；W_{ij}— 节点连接权值；θ— 阈值；U_i^k— 中间变量。

④学习误差：$d_i^m = X_i^m(1 - X_i^m)(X_i^m - Y_i)$

$$d_i^k = X_i^k(1 - X_i^k)\sum W_{1i}d_1^{k+1}$$

d_i^m— 输出层学习误差；d_i^k— 其它层学习误差；Y_i— 期望输出。

⑤权值修正：$W_{ij}(t+1) = W_{ij}(t) - \eta d_j^k X_j^{k-1} + \alpha \triangle W_{ij}(t)$

$\triangle W_{ij}(t) = W_{ij}(t) - W_{ij}(t-1)$

η— 学习速率；α— 权系数修正常数。

⑥误差函数：$e = 1/2\sum(X_i^m - Y_i)^2$

Y_i— 输出单元期望值；X_i^m— 实际输出值[1]。

1.2 BP网络模型的C++构造实现

用C++实现的BP网络模型，包括两大模块，即网络模型数据结构模块和网络功能函数模块。

1.2.1 模型数据结构模块

为了实现BP模型算法，并使程序结构清晰，易于阅读、扩展，定义了以下网络模型数据结构。

```
Typedef struct
{  double in [];            //输入样本 X
   double input[];          //学习输入样本 X,归一化后的结果
   double out [];           //期望输出 Y
   double output[];         //学习期望输出 Y,归一化后的结果
   int layer[];             //记录每层神经元的个数
   double w[][][];          //权系数
   double ww[][][];         //前一时刻权系数
   double www[6][6][6];     //权系数差值
   double y[21];            //记录实际输出值
   double x[6][6];          //每层输出
   double d[6][6];          //每层学习误差
   double p=0.2,q=0.8       //p-学习速率,q-权系数修正常熟
} BP-network;
```

1.2.2 模型功能函数模块

BP程序中所包含的函数较多，下面简要介绍几个关键函数及其功能。

①void FormCreate( )的功能：初始化网络的权系数：即0～1之间的random( )，及其它相关变量的初始化和样本归一化。

②double forward( )的功能：实现BP网络的前向传播过程，计算出各层每个神经元的输出。

③void backward( )的功能：实现网络的误差反向传播，计算各层的学习误差。

④void modify( )的功能：修正网络的权系数和阈值。

⑤Bool judge( )的功能：判断学习结束的条件。

1.2.3 BP实现算法

BP神经网络程序的主流程如图3所示。

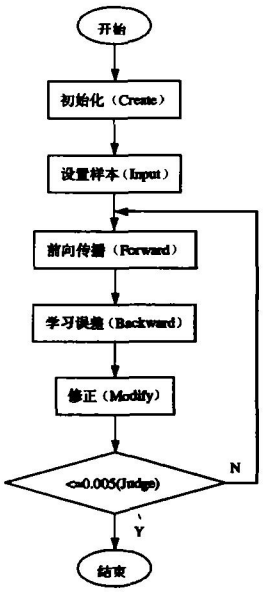


图3 算法流程图

①初始化。对各层的权系数 W_{ij} 置一个较小的非零随机数，其中 $W_{in+1} = -\theta$；同时将程序中用到的各个变量初始化。

②设置样本。依次输入符合条件的样本 $X = (X_1, X_2, \cdots, X_n, 1)$，计算出对期望输出 $Y = (Y_1, Y_2, \cdots, Y_n)$，及在给定样本范围内的最大最小输出，分别为112和4，用函数 $f(x) = (x - (-5))/(112 - (-5))$ 将样本和期望输出归一化，作为神经网络学习中的输入和期望输出。

③前向传播。调用forward函数，计算出各个神经元的输出。

④学习误差。调用backward函数，计算出各层的学习误差。

⑤修正。调用modify函数，根据误差，修改权系数 W_{ij} 和阈值 θ。

⑥判断。通过judge函数，判断期望输出和实际输出之间的平方和是否达到小于0.005的要求，满足，则结束学习，显示结果；反之，则循环③④⑤步，直到满足学习结束条件。

2 学习结果

2.1 输出对照

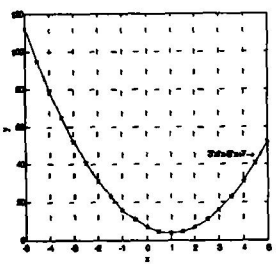
20组输入样本 X、期望输出 Y 和实际输出结果 Y' 及输入输出图如表1、图4所示。

表1 输入与输出对应表

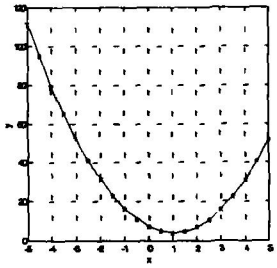
组数	1	2	3	4	5
输入 X	-5	-4.5	-4	-3.5	-3
期望 Y	112	94.75	79	64.75	52
实际 Y'	111.630	94.826	78.742	64.668	52.028
组数	6	7	8	9	10
输入 X	-2.5	-2	-1.5	-1	0
期望 Y	40.75	31	22.75	16	7
实际 Y'	40.497	31.192	22.852	15.999	6.962
组数	11	12	13	14	15
输入 X	0.5	1	1.5	2	2.5
期望 Y	4.75	4	4.75	7	10.75
实际 Y'	4.954	4.276	4.403	6.668	10.487
组数	16	17	18	19	20
输入 X	3	3.5	4	4.5	5
期望 Y	16	22.75	31	40.75	52
实际 Y'	15.940	22.656	30.826	40.413	51.999

可以看到误差函数 $e < 0.005$ 时，模拟出来的双曲抛物线与所期望的曲线已经非常接近，这说明我们已经成功用1-4-4-3-1型实现了函数 $Y = 3X^2 - 6X + 7$（-5

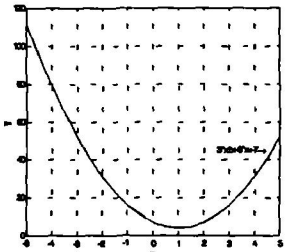
≤x≤5)的功能,BP网络学习成功。



a.期望输出图形



b.网络输出图形(e<0.005)



c.重合后的图形

(实线是期望输出,虚线为网络实际输出)

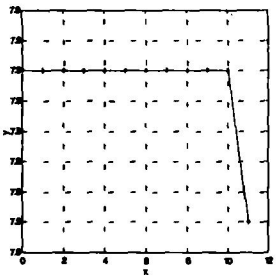
图4　输入输出图

2.2　权系数

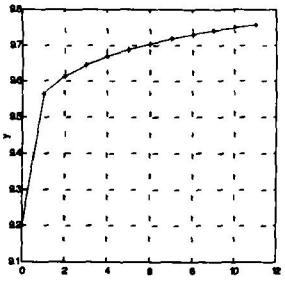
每层10组经典的权系数及变化图如表2和图5所示(除第一层输入层)。

表2　权系数变化表

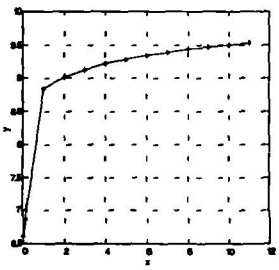
层次	W	第1组	第2组	第3组	第4组	第5组
第二层	W41	0.79	0.79	0.79	0.79	0.79
第三层	W31	0.92	0.956 4	0.961 2	0.964 4	0.966 8
第四层	W13	0.66	0.883 9	0.901 4	0.912 9	0.921 5
第五层	W11	0.5	1.781 00	1.830 89	1.864 15	1.889 14
层次	W	第6组	第7组	第8组	第9组	第10组
第二层	W41	0.79	0.79	0.79	0.79	0.789 9
第三层	W31	0.968 7	0.970 3	0.972 9	0.973 9	0.975 6
第四层	W13	0.928 3	0.933 9	0.943 0	0.946 7	0.952 6
第五层	W11	1.909 17	1.925 88	1.952 78	1.963 96	1.981 57



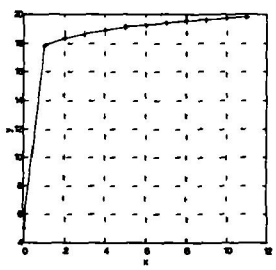
(1)第二层 W41 变化曲线



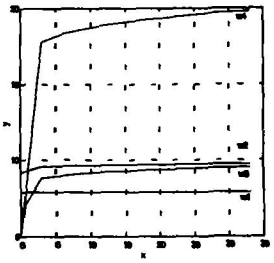
(2)第三层 W31 变化曲线



(3)第四层 W13 变化曲线



(4)第五层 W11 变化曲线



(5)四个权系数变化曲线

图5　权系数变化图(放大比1:10)

观察这些权系数变化图,可以看到一些共同点:权系数的变化趋势随着学习次数的增加而减小;各层权系数的变化程度随着层数的增加而增大,越接近输出层的变化越明显。

3　BP算法改进的讨论

3.1　对η和α的讨论

在实际应用中BP算法存在两个重要问题:收敛速度慢;样本函数存在局部极小[4]。在提高收敛速度方面,已有许多人做了研究并提出许多方法,如α—权系数修正常数的加入。在本次作业中作者尝试了通过对学习率η和权系数修正常数α的改变,分别比较了不通情况下的学习次数,如表3所示(其中$\eta=0.2$, $\alpha=0.8$,是上述学习过程中选用的参数值,样本数为1)。

表3　学习次数对照表(单位:次)

η变,α不变	1	2	3	4	5	平均
η=0.1,α=0.8	27 972	28 971	29 255	28 998	29 533	28 946
η=0.2,α=0.8	14 425	14 226	14 522	13 897	13 859	14 185
η=0.4,α=0.8	7 047	7 019	7 002	7 058	7 214	7 068
η变,α不变	1	2	3	4	5	平均
η=0.2,α=0.7	21 338	21 150	21 367	21 003	21 163	21 204
η=0.2,α=0.8	14 425	14 226	14 522	13 897	13 859	14 185
η=0.2,α=0.9	7 030	6 908	6 930	6 791	6 938	6 919
η变,α变	1	2	3	4	5	平均
η=0.1,α=0.7	42 626	43 133	43 718	42 942	41 499	42 783
η=0.2,α=0.8	14 425	14 226	14 522	13 897	13 859	14 185
η=0.4,α=0.9	3 250	3 150	3 287	3 474	3 416	3 315

从上表可以看出,当α不变,η由小变大的过程中,*BP*网络的学习次数随着η的增大而减少;当η不变,α由小变大的过程中,BP网络的学习次数随着η的增大而减少;当η和α均改变,在η和α同时由小变大的过程中,BP网络的学习次数随着η和α的增大而明显减少。另外,偏大的学习速率可能增加系统的不稳定因素。因此,在*BP*网络的学习算法中,学习率η(0.1～0.4)和权系数修正常数α(0.7～0.9)的选择在正确的取值范围内应尽可能取偏大的值,以加快收敛速度。

3.2　对归一化的讨论

在BP神经网络中,我们采用Sigmoid, $f(x)=1/(1+\exp(-x))$为激发函数,由函数特性可知$0\leqslant f(X)\leqslant 1$,当给定的样本函数$Y$取值不在[0,1]区间内时,需要进行样本归一化,样本归一化的方法有两种:(以本文中样本

函数 $Y = 3X^2 - 6X + 7$ 为例，设样本数为 20，$X = (X_1, X_2, \cdots, X_n)$)

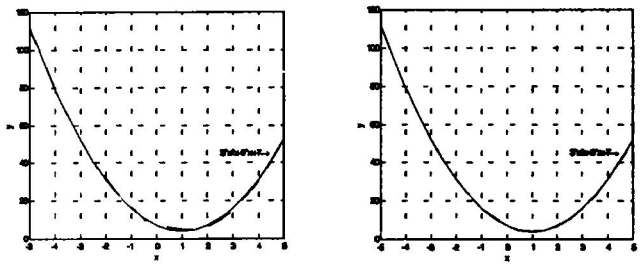
第一种，分别将 X、Y 以不同归一因子缩小，即 $(MAX_x - MIN_x) \times Q_1 = 1 - 0$，$(MAX_y - MIN_y) \times Q_2 = 1 - 0$，算出归一因子后，用 Q_1 与20个样本一一相乘，得出归一化后的输入样本 $X' = (X'_1, X'_2, \cdots, X'_{20})$；用 Q_2 与20个样本对应的期望输出相乘，得出归一化后的期望输出 $Y' = (Y'_1, Y'_2, \cdots, Y'_{20})$，然后用归一化后的输入输出代进神经网络中进行学习，学习后的实际输出，用函数 $f(x) = (x - MIN_y)/Q_2$ 进行复原。

第二种，将 X、Y 放在同一坐标系中以同一个归一因子缩小，即 $(MAX - MIN) \times Q = 1 - 0$，然后，用 Q 与 X、Y 相乘，得出归一化后的输入样本 $X' = (X'_1, X'_2, \cdots, X'_{20})$ 和期望输出 $Y' = (Y'_1, Y'_2, \cdots, Y'_{20})$，将归一化后的输入输出代进神经网络中进行学习，学习后的实际输出用函数 $f(x) = (x - MIN)/Q$ 进行复原。其中第二种较第一种简单，本文中采用了第二种归一化方法(见表4)。

表4 归一化后输入输出表

组数	1	2	3	4	5
输入 X	-5	-4.5	-4	-3.5	-3
归一化后输入 X'	0	0.004 273	0.008 547	0.012 805	0.017 094
期望输出 Y	112	94.75	79	64.75	52
归一化后输出 Y'	1	0.852 564	0.717 948	0.596 153	0.487 179
组数	6	7	8	9	10
输入 X	-2.5	-2	-1.5	-1	0
归一化后输入 X'	0.021 367	0.025 641	0.029 914	0.034 188	0.042 735
期望输出 Y	40.75	31	22.75	16	7
归一化后输出 Y'	0.391 025	0.307 692	0.237 179	0.179 487	0.102 564
组数	11	12	13	14	15
输入 X	0.5	1	1.5	2	2.5
归一化后输入 X'	0.047 008	0.051 282	0.055 555	0.059 829	0.064 102
期望输出 Y	4.75	4	4.75	7	10.75
归一化后输出 Y'	0.083 333	0.076 923	0.083 333	0.102 564	0.134 615
组数	16	17	18	19	20
输入 X	3	3.5	4	4.5	5
归一化后输入 X'	0.068 376	0.072 649	0.076 923	0.081 196	0.085 470
期望输出 Y	16	22.75	31	40.75	52
归一化后输出 Y'	0.179 487	0.237 179	0.307 692	0.391 025	0.487 179

无论算法中使用何种归一化方法，归一因子的选择是极其重要的，这关系到学习算法的成功与否[5]。在本文的样本函数中，$X \in [-5, 5]$，$Y \in [4, 112]$，正确的 MAX = 112，MIN = -5，归一因子 $Q = 1/117$，学习后期望输出和实际输出对照如图6(b)所示。倘若误将 MAX = 112，MIN = 4，归一因子 $Q = 1/108$，学习后期望输出和实际输出对照如图6(a)所示。两幅图对照均误差不大，具有一定迷惑性。a图在误差允许范围内十分象正确结果，而实质却使用了错误的归一因子，是不正确的。故在BP神经网络学习算法中，正确的计算归一因子是极其重要的。



a.错误归一化重合后的图形 b.正确归一化重合后的图形

图6 归一化结果对比图

4 小 结

BP网络是一种无反馈的多层前向网络，可以对网络中各层的权系数进行修改，目前，BP算法是自动控制上最重要、应用最多的一种有效算法[6]。通过用 C++ 对 BP 神经网络模型算法程序的构造和实现，加强了对 BP 算法的了解和学习，对 BP 算法中收敛速度和样本函数局部极小等问题，还有待进一步的研究。

参考文献：

[1] 张庆年. 前馈神经网络的特性分析与应用[J]. 武汉交通科技大学学报，1999，23(4)：372－374.

[2] 戴 葵. 神经网络实现技术[M]. 长沙：国防科技大学出版社，1998.

[3] 余永权. 神经网络模糊逻辑控制[M]. 北京：电子工业出版社，1999.

[4] 党建武. 神经网络网络技术及应用[M]. 北京：中国铁道出版社，2000.

[5] 吴云芳. 改进的BP神经网络模型在大坝安全监测预报中的应用[EB/OL]. http://www.lodestar.com.cn/files/wx/sdzsj/2002－2/5.htm，2002.

[6] 张乃尧. 神经网络与模糊控制[M]. 北京：清华大学出版社，1998.

(上接第92页)

仿真步距和计算工作量以及数值解稳定性的矛盾。采用非线性计算步距 h 的方法，解决了病态系统数值仿真中出现的数值解失真的问题，解决了数值仿真中计算速度、计算精度与仿真稳定性之间的矛盾。

仿真表明，在各种数值计算中表现出优越的自适应性，具有计算速度快、精度高和稳定性好的特点，同时对非病态系统同样适用。

参考文献：

[1] 张晓华. 控制系统数字仿真与CAD[M]. 北京：机械工业出版社，1999.

[2] 郎泉江. SISO病态系统的识别和数字仿真[J]. 东北电力学院学报，1996，(1)：45－51.

[3] 王正中. 系统仿真技术[M]. 北京：科学出版社，1986.

[4] 薛定宇. 控制系统计算机辅助设计—Matlab语言及应用[M]. 北京：清华大学出版社，1996.

[5] 李庆扬. 数值分析[M]. 武汉：华中理工大学出版社，1995.