ISSN1000-9825 Journal of Software 软件学报 2000,11(1):133~137

提高小脑模型神经网络精度的算法及仿真应用*

朱庆保 陈蓁

(南京师范大学数学与计算机科学学院 南京 210097)

E-mail: qb.zhu@jlonline.com

摘要 CMAC(cerebella model articulation controller)神经网络的局部结构使得学习非线性函数更快.然而,在许多应用领域,CMAC的学习精度不能满足应用要求.该文提出了一种改进CMAC学习精度的联想插补算法,同时给出了一个仿真实验.其结果表明,使用此算法,改进的CMAC的学习精度比改进前提高了10倍,学习收敛也更快.

关键词 小脑模型,神经网络,联想插补,仿真,精度,算法.

中图法分类号 TP183

小脑模型神经网络又称为小脑模型关节控制器(cerebella model articulation controller,简称CMAC),是J.S.Albus在1975年提出来的[1].最初,CMAC主要用于机器人控制,经过以后的不断研究和实践表明,CMAC可以学习种类广泛的非线性函数,而且迭代次数比BP网络少得多,因此特别适合于机器人的轨变学习控制、实时学习控制、非线性函数映射以及模式识别等.

CMAC作为一种查表技术[2~4]被广大研究者所公认,其输入必须量化,相近的输入具有相近或相同的输出,故输出是不连续的,在一个量化状态内,其输出是一个常数,亦即输出函数的导数不存在[4,5],也就是说,CMAC映射非线性函数的精度是非常有限的.文献[6]通过仿真研究后也指出,CMAC的学习算法比较粗糙,因此系统的映射能力也比较粗糙、映射的精度不高,只能说大体上具有一定的泛化非线性映射的能力.

如何在保证CMAC收敛速度的前提下进一步提高其精度,就成为当前CMAC神经网络理论研究的一项重要课题.因为CMAC的实质是一种查表技术,提高精度的一个重要途径是减小量化间隔,增加训练样本数.而量化间隔的减小使训练学习过程占用的容量急剧增加,学习速度变慢.为解决这一矛盾,可采用自适应技术,根据误差反馈自适应地改变量化间隔,而不是采用固定间隔[2].也有研究提出了分组量方法[3].这些方法虽然调和了精度、量化间隔与容量之间的矛盾,使它们之间的分配趋于最佳,但没有从根本上解决精度问题.提高精度的另一条途径是对输出进行处理.一种方法是基于CMAC是一种查表技术这一事实,把CMAC的寻址技术与加权回归技术(weighted regression)相结合,对输出进行拟合平滑处理,使输出除了量化区间的边界之外处处可导[4],从而提高输出精度.但这种方法的算法较复杂,使学习速度大大降低,其精度受到回归精度的限制.基于国内外研究的现状和不足,本文提出了一种提高输出精度的联想插补算法.仿真实验表明,该算法使CMAC输出精度提高了10倍,而且基本不占内存,算法简单,速度快,易于实现.我们用本文提出的算法去学习逼近一个非线性传感系统特性曲线和用于多自由度的关节控制所表示的坐标变换问题,取得了十分令人满意的效果.

1 CMAC的基本结构和基本原理

1.1 CMAC的基本原理结构

CMAC的简单原理结构模型如图1所示,S为输入状态空间,由所有可能的输入向量S_i组成,对于一个输

* 本文研究得到江苏省科委应用基础基金(No.BJ97122)资助.作者朱庆保,1955年生,副研究员,主要研究领域为自动控制、智能控制.陈蓁,女,1974年生,硕士生,主要研究领域为微机应用.

本文通讯联系人:朱庆保,南京 210097,南京师范大学数学与计算机科学学院

本文1998-08-03收到原稿,1999-02-01收到修改稿

入向量 $S=(s_1,s_2,\ldots,s_n)$,CMAC 产生一个输出向量 $P=f(s)$. 为了对给定的输入状态 S 计算输出向量,进行了二次映射:

$$f:S\to A,\ g:A\to P.$$

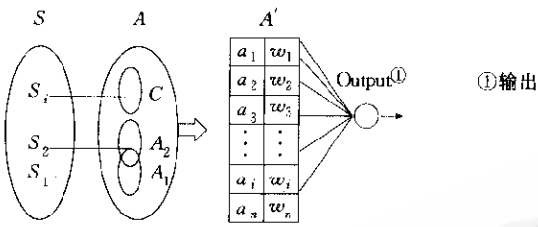

Fig. 1　Basic model structure of CMAC
图1　CMAC的基本模型结构

1.2　$S\to A$ 及 $A\to P$ 的映射

$S\to A$ 的映射是通过映射函数 f 将 S_i 映射到一个巨大的概念存储器 A 中的 C 个单元. 输入空间邻近的两个输入向量在存储器 A 中有部分重叠的单元,距离越近,重叠越多. 反之,若输入空间远离的两个输入向量,在 A 中则并不重叠,如图 1 所示. 这种特性使得 CMAC 具有很强的泛化能力.

对于一个实际的系统,输入空间的向量数是很大的,存储器 A 所需的容量也是十分庞大的,用计算机实现是相当困难的. 为此,应用一种计算机存储压缩技术——Hash 映射技术把所需的存储空间映射到一个小得多的物理存储器 A'. 任何 CMAC 网络的输入激活 C 个真实存储位置,而把这些位置的值相加即得到输出向量 P.

1.3　基本学习算法

设训练样本为 (S^*,P^*),CMAC 的输出为 P,给定一个误差限 ε_j,若 $|P_j^*-P_j|\leqslant\varepsilon_j(j=1,2,\ldots,L)$,对应的权不修改,否则按下式调整权:

$$w_j(t+1)=w_j(t)+\beta\frac{p_j^*-p_j}{C}. \tag{1}$$

其中 β 为学习步长,t 为学习次数,C 为激活的单元数.

2　利用联想特性提高输出精度的算法

2.1　联想插补算法

由 CMAC 的原理可知,当 CMAC 用于映射非线性函数时,其输入必须量化. 提高其量化级数,虽然精度会有所提高,但所占用的存储器容量就会很大,而且学习速度变慢,增加量化间隔则使精度下降. 如何解决这一矛盾呢?考虑到人脑具有联想推测功能,当接受未经学习过的信息时,会根据过去学习过的相近的知识或经验作出较准确的推断. 根据这一特征,能否根据训练样本提供的信息,通过联想而较准确地推测出非样本输入的响应呢?差分插值原理恰恰具有这样的特性.

考虑一维函数 $y=f(x)$,设 n 个训练样本为等距取点,即 $(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)$,其中 $x_k=x_0+kH$,式中 $k=1,2,\ldots,n$ 为节点序数,H 为节点距或步长.

对于任意输入 x_i,若有 $x_i>x_k$,则有如下差分插值多项式(舍去高次项):

$$\begin{aligned}y_i&=y_k+mh\\&=y_k+m\Delta y_k+\frac{m(m-1)}{2!}\Delta^2y_k+\frac{m(m-1)(m-2)}{3!}\Delta^3y_k+\frac{m(m-1)(m-2)(m-3)}{4!}\Delta^4y_k,\end{aligned}\tag{2a}$$

其中 $m=\dfrac{x_i-x_k}{H}$,$\Delta y_k=y_{k+1}-y_k$,$\Delta^2y_k=y_{k+1}-\Delta y_k$,$\Delta^3y_k=\Delta^2y_{k+1}-\Delta^2y_k$,$\Delta^4y_k=\Delta^3y_{k+1}-\Delta^3y_k$.

若 $x_i<x_k$,则有

$$\begin{aligned}y_i&=y_k-mh\\&=y_k-m\Delta y_k+\frac{m(m-1)}{2!}\Delta^2y_k-\frac{m(m-1)(m-2)}{3!}\Delta^3y_k+\frac{m(m-1)(m-2)(m-3)}{4!}\Delta^4y_k,\end{aligned}\tag{2b}$$

其中 $m=\frac{X_k-X_i}{H}$,$\Delta y_k=y_k-y_{k-1}$,$\Delta^2 y_k=\Delta y_k-\Delta y_{k-1}$,$\Delta^3 y_k=\Delta^2 y_k-\Delta^2 y_{k-1}$,$\Delta^4 y_k=\Delta^3 y_k-\Delta^3 y_{k-1}$.

在一般情况下,多维函数 $y=f(s_1,s_2,\ldots,s_m)$ 的插值算法十分复杂.设某 m 维的函数在 $[R_1,R_2]$ 上有界,我们在 $[R_1,R_2]$ 区间给出有限个学习训练样本,即 $Y^*=f(S_1^*,S_2^*,\ldots,S_i^*,\ldots,S_m^*)$,$S_i^*$ $(i=1,\ldots,m)$ 为给定的输入,Y^* 为期望的输出.对应于每个输入变量 S_i^* 有 n 个输入元素,即 $S_i^*=(s_{i1},s_{i2},\ldots,s_{ij},\ldots,s_{in})$,$s_{ij}\in[R_1,R_2]$.在 $[s_{ik},s_{ik+1}]$ 这样小的取值区间内,函数的输入、输出的微分可近似为线性关系,故有

$$\delta Y=\frac{\partial f}{\partial s_1}\delta s_1+\frac{\partial f}{\partial s_2}\delta s_2+\ldots+\frac{\partial f}{\partial s_m}\delta s_m. \tag{3}$$

因此,可以把多维函数的插值问题分解成一维函数的插值问题.当各变量输入值均介于两样本值之间时,可用式(2)和式(3)进行分维叠加插值.

2.2 实现插补的方法

CMAC 的学习方法可以根据学习或问题要求,事先设计出期望值,让 CMAC 的输出直接达到或接近期望值,这种情况属于无教师学习情况.例如,期望 CMAC 产生的函数为 F,那么对于输入空间内的每一点,$P^*=F(S)$ 就是输出向量的期望值.对任意输入 S,计算该点的函数值 $P=F(S)$,当两者误差大于规定的误差 ε_j 时,用式(1)来调权值,重复这一过程,直到各元素误差都小于 ε_j 为止.在此基础上,可以进行插补运算,其方法如下.

方法 1. 用式(1)学习调权完毕后,若 $P_j>P_j^*$,用式(2a)插补,反之用式(2b)插补.插补过程是以 P_j^* 作为结点数据的,可以根据输入量化间隔及输入值,临时计算出插补所需的相关的 P_j^*,P_{j+1}^*,...(用式(2a)插补时)或 P_j^*,P_{j-1}^*,...(用式(2b)插补时).这种方法的优点是不占额外的内存.也可把有关的期望值事先存于一个表格中,根据表中存储的期望值(结点值)进行插补.为了不占内存,可以把表格存于一个数据文件中,通过文件指针来查找有关的结点值.

有时,在用 CMAC 映射某一函数时,是先用该函数的部分样本对 CMAC 进行训练,训练好后,再让 CMAC 映射该函数,这种情况即为有教师学习.这时,当用式(1)进行逐一学习时,由于 CMAC 相近的输入存在重叠的权,存在着相互间的学习干扰,尤其当给定的误差限 ε_j 较小时,需经反复学习,学习的次数变多,速度变慢,甚至导致不收敛.联想插补方法不仅可以提高 CMAC 的学习精度,而且可以有效地解决这一缺陷.其方法有两种,一种是加大 ε_j;另一种是设定一个学习次数计数值 J,并设定允许的学习次数 M,当用式(3)学习的次数 $J\geqslant M$ 时,即退出本次学习过程,再转到上述插补过程.这两种方法,其权都只进行了粗略的调整,其精度是通过插补来保证的,其学习速度可提高数倍至数十倍,精度也大大提高.在实用中,当用训练样本对网络进行训练时,经过用式(1)进行调权后,即达到了对网络的训练目的.这时可以直接存储样本点的输入及其所对应的期望输出,存储的样本输入点用于判别任意输入的结点位置,其对应的输出作为插补结点.这样,对非样本点用式(1)调权,并用式(2)进行插补,即可得到更高的输出精度.

方法 2. 根据 CMAC 是一种查表技术这一事实,改变输入映射算法,使训练样本通过某种编码,使其权形成类似多维向量数组的存储形式,这些权值形成可供插补用的结点数据,对于任意输入,即可进行分维叠加插补运算.

需要指出的是,联想插补方法是根据已有数据的信息推测出数据的变化趋势,从而实现插补,因此,特别适于变化缓慢的单调函数.对于多值函数,在其拐点处,其插值精度会受到影响,这时,可以采用对这些点进行修正或在这些拐点位置附近增加样本密度等方法来解决.

3 仿真应用研究

在实际应用中,需要 CMAC 学习的函数不一定有明确的解析式,但可以通过给定一组输入信号,根据题目要求,设计出一组期望的输出作为教师信号,经 CMAC 学习后,即可以由 CMAC 根据任意输入求解输出.其中,输入可以是由传感器获取的信号,也可以是输入等其他来源.

3.1 对多维非线性函数的学习与求解

3.1.1 求解的题目

为说明问题,我们选择小脑运动控制中的多自由度的关节控制所表示的坐标变换问题[6]作为我们求解的题

目.以 $X(x,y,z)$ 表示肢体要达到的空间坐标,以 $(\theta_1,\theta_2,\ldots,\theta_n)$ 表示各关节应转过的角度,则有 $X=f(\theta)$.我们的问题就是,已知 θ 求 X(或已知 X 求 θ).设空间某点的坐标为 (x,y,z),三自由度手臂的角度为 $(\theta_1,\theta_2,\theta_3)$,手臂长度为 L,则有

$$\begin{cases} x = L[\cos\theta_1 - \cos(\theta_1 - \theta_3)]\cos\theta_2 \\ y = L[\cos\theta_1 - \cos(\theta_1 - \theta_3)]\sin\theta_2 \\ z = L[\sin\theta_1 - \sin(\theta_1 - \theta_3)] \end{cases}. \tag{4}$$

我们的任务就是已知 $(\theta_1,\theta_2,\theta_3)$,求解 (x,y,z).

3.1.2　样本集的确定与输入变换

因为对 x,y,z 的求解方法相同,故仅给出 x 的求解过程,并且为了说明问题简便而不失一般性,令 $(\theta_1,\theta_2,\theta_3)$ 的取值区间为 $[0°,90°]$.设学习训练元素的取值间隔 $h=7.5°$,则 $\theta_1,\theta_2,\theta_3$ 分别在区间 $[0°,90°]$ 的取值数为 $M=13$.令

$$i=int(\theta_1/h),\quad j=int(\theta_2/h),\quad k=int(\theta_3/h).$$

经过上述变换后,$i,j,k\in[0,12]$.相应的教师信号由式(4)变换后给出.

$$x=L(\cos(\pi*i*h/180)-\cos(\pi*i*h/180-\pi*k*h/180))\cos(\pi*j*h/180).$$

因为 C 语言中的 cos 函数为弧度表示,故乘以 $\pi/180$ 将角度变为弧度,并设 $L=30$.这样,根据给定的学习样本,对网络进行训练并存储后,对输入区间内的任意值,用式(1)学习并经插补后,网络即可输出对应的解,即 x.

3.1.3　仿真实验结果

按上述样本取值对网络进行训练后,再对网络精度进行检验,其仿真结果见表 1 中第 1 行.理论上,检验网络的数据应遍布整个输入空间,但实际上是困难的,为此,取检验网络的数据如下:令 $\theta_1=0,1,2,\ldots,i,\ldots,90$;$\theta_2=0,1,2,\ldots,j,\ldots,90$;$\theta_3=0,1,2,\ldots,k,\ldots,90$.组合成 90^3 个输入组合,则有

$$e_i=fabs(x_{1i}-x_{2i}),\quad aver=Q/p=\sum e_i/p,\quad \max=\mathrm{MAX}\{e_i\},\quad \min=\mathrm{MIN}\{e_i\},\quad i\in[0,90^3],$$

其中 x_{1i} 为由式(4)计算出的每个输入组合的理论值;x_{2i} 为 CMAC 实际的输出值;e_i 为误差的绝对值;$P=90^3$,为误差的个数;$aver$ 为平均误差.

Table 1　Errors analysis and comparison (in x coordinate)

表 1　误差分析与比较(在 x 坐标)

	Maximal error①	Minimal error②	Average error③
The results with algorithm of this paper④	0.2772	0.001	0.0566
The results with algorithm of original CMAC⑤	3.649	0.001	0.677

① 最大误差,② 最小误差,③ 平均误差,④ 本文的算法结果,⑤ 原 CMAC 算法结果.

由于数据处理量巨大,为此,我们用 C 语言编写了一个检验函数来完成这一工作.表 1 中第 1 行的数据由该函数按本文算法完成,其中的最大误差、最小误差和平均误差即为 max,min 和 vear.第 2 行的数据是文献[6]用 Albus 的 CMAC 算法对本题目进行仿真且其训练元素 θ 的取值区间和取值间隔与本文相同时得到的结果.

从表 1 中可以看出,本文提出的插补算法与 Albus's CMAC 算法相比,对非线性函数的映射精度提高了 10 倍,满足实用要求,解决了 CMAC 精度低的问题.很显然,本文提出的插补算法的性能远远优于对输入量化间隔的自适应算法,与加权回归方法相比,本文提出的算法精度更高,算法更简单,学习速度更快,更易于实现.

4　结　语

微机信息查新表明,提高 CMAC 的精度是对 CMAC 进行改进的一项重要的研究内容,围绕这一课题有多种改进方案,但本文提出的对 CMAC 的输出进行联想插补尚属首次.理论和仿真实验都证明了这种算法是提高 CMAC 输出精度的有效方法.经过插补,CMAC 输出精度提高了 10 倍,能够满足大多数应用的要求.

参考文献

1　Albus J S. A new approach to manipulator control: the cerebella model articulation controller(CMAC). Transactions of the

ASME, Journal of Dynamic System, Measurement and Control, 1975,97:220～227
2 Hirashima Y, Iiguni Y, Adachi N. An adaptive control system design using a memory based learning system. International Journal of Control, 1997,68(5):1085～1102
3 Lee Chau-jhy, Lin Wei-song. A method of clustering quantization for bettertraining of CMAC. Journal of the Chinese Institute of Engineers, 1996,19(3):309～320
4 Lin Chun-shin, Chiang Ching-tsan. Integration of CMAC technique and weighted regression for efficient learning and output differentiability. IEEE Transactions on Systems, Man and Cybernetics, Part B, 1998,28(2):231～237
5 Chiang Ching-tsan, Lin Chun-shin. CMAC with general basis functions. Neural Networks, 1996,9:1199～211
6 Ou-Yang Kai, Chen Hui, Zhou Ping *et al*. Generalization of neural network model (CMAC) for coordinatetransformation in neural computation. Acta Automatica Sinica, 1997,23(4):475～481
(欧阳楷,陈卉,周萍等.神经计算中坐标变换的网络模型(CMAC)的泛化特性.自动化学报,1997,23(4):475～481)

An Algorithm for Improving the Accuracy of Cerebella Model Articulation Controller Neural Networks and Simulation Application

ZHU Qing-bao CHEN Zhen

(*School of Mathematics Science and Computer Science Nanjing Normal University Nanjing* 210097)

Abstract The local structure of CMAC (cerebella model articulation controller) neural networks results in faster learning of nonlinear functions. However, the learning accuracy of CMAC is too low to meet the requirements of application in many fields. Hence, an associative interpolation algorithm is proposed in this paper for improving the learning accuracy of CMAC. Meanwhile, a simulation experiment is described. Its result shows that the learning accuracy of the improved CMAC is ten times higher than that of the original CMAC, and the learning convergence is also faster.

Key words Cerebella model articulation controller (CMAC), neural network, associative interpolation, simulation, accuracy, algorithm.