人工神经网络 进化设计 遗传算法 (12)

计算机科学2000Vol. 27№. 6

43-45,31

进化设计人工神经网络

Designing ANN with Genetic Algorithm

TP18

王洪燕 杨敬安 蒋 培

(合肥工业大学人工智能研究所 合肥 230009)

Abstract A new method for designing artificial neural network architectures, which is based on L-system and genetic algorithm, is presented in this paper. Production rules and parallel algorithm are used to solve traditional neural network design problems. The experiment results proved that the algorithm can improve network performance and converge speed.

Keywords Genetic algorithm, Artificial neural network, L-system, Parallel algorithm

1 前言

从1985年 Rumelhart 提出 BP 算法以来，神经网络理论发展迅速，多层前向型网络更成为用途最为广泛的网络之一[1]。探索高效的神经网络学习算法和优良的网络结构成为推动神经理论和应用的重要因素。在网络结构设计中经常遇到的两个问题是：所得网络缺少泛化能力和易产生干扰现象[3,7,11]。一个网络具有泛化能力是指网络能正确响应未出现在训练集中的输入。出现这种现象的原因是网络结构存在太大的自由度，虽然大量权系组合成多个函数并能响应训练样本集，但并不能精确反映我们的目标函数。另一相关现象称为过训练，当用问题域的样本子集长时间训练网络时，整个网络就会越来越倾向于反应训练样本子集而不是整个问题域。

当用一个网络来表示多个相关程度小的问题时，网络易出现干扰现象[5]：用新问题的样本集训练网络时，已训练过的其它问题会被遗忘或受干扰，反之亦然。Jacobs 等将干扰分为两类：空域干扰和时域干扰。空域干扰发生在输出单元的输出信息互相冲突时；时域干扰发生在训练样本集不一致时。

为解决上述问题，许多学者致力于研究优化神经网络结构的方法[3,11]，Rumelhart 用 XOR 问题来说明网络结构的重要性：对标准的含有2个输入节点、两个隐节点和一个输出节点的网络，经1650次迭代收敛；若在输入和输出节点间直接附加两个连接权，仅须30次迭代就可获得同样的结果。

人的大脑从结构上可分为几个模块，每个模块负责特定的神经功能[7,8]。这种结构在人出生时就已具雏形，并可直接执行学习任务，最小规模的模块可以看作是人脑的基本计算单元，每个模块包含大约100个神经元细胞，其功能形成与后天生长学习过程中经常处理的信息类型有关。每个基本模块结构和整个大脑以一定的机制进行遗传编码，通过产生越来越多结构类似的基本单元进行大脑的生长。这种重复复制相同单元的生长过程也是一种普通的自然现象。

遗传算法是一种模仿生物进化机制的自适应随机搜索方法，它是由美国 Michigan 大学的 Holland 教授在七十年代提出来的。作为解决大规模优化问题的一种有力的工具，遗传算法已成功地应用于商业、工程及科研领域。一般情况下，遗传算法能在一个合理的时限内求得问题的优化解，但求解大型的结构复杂的问题、对运算时间的需求就会迅速增加，因而为了减少运算时间，提高解的质量，对并行遗传算法的研究已成为当务之急[2]。

已经证明遗传算法收敛到唯一解的时间和群体的规模有很大关系，Goldberg 和 Deb 实验了不同的选择方法，较快的算法的时间复杂度为 $O(S\log S)$，其中 S 为群体中个体的个数。但是较大规模群体能增加遗传算法全局寻优能力和优化解的质量[4]。这一对矛盾是标准遗传算法需要首先解决的问题。

对复杂问题求解的自然而有效的途径是采取分而

王洪燕 博士生，主要研究兴趣为进化计算，人工智能。杨敬安 教授，博士生导师，IEEE 高级会员，纽约科学院 Fellow Academy Member，主要研究兴趣为图像理解，模式识别、计算机视觉，人工智能与机器人以及多传感器融合与集成。蒋 培 博士生，主要研究兴趣为进化计算，机器学习。

洽之的策略，将复杂问题分解为数个简单的子问题，然后分别进行求解[10]。

本文首先模拟大脑的生长过程及其生物神经系统功能和结构特征，利用L-系统[5,9]和遗传算法建立遗传搜索和人工生长模型；然后针对网络结构设计搜索空间大、网络训练时间长等特点，提出一个分层并行遗传算法，新算法首先将一复杂问题分解为多个简单问题，再将各子问题对应的子群体进行分组，模型的处理单元分为主处理单元、辅助处理单元和从处理单元三个层次，该算法在保持通讯开销增长幅度较小的情况下，使用多从处理单元有效地提高算法模式搜索效率。最后通过仿真实验证明了该方法的有效性。

2 L-系统

L-系统是在模拟植物的生长过程中引入的，可以看作是分形的一种[6]，其基本原理是利用了并行串重写机制。经过对一基本单元反复应用这种产生式规则而得到的结果串，可以根据符号的语义存在多种解释[9]，最常见的是将其解释为一棵树的骨架拓扑。为了刻画这种拓扑结构，需要引入一套约束规则对其进行上下文敏感的图重写。应用L-系统生成神经网络结构所引入的约束包括模块化结构约束和自相似约束。

2.1 串编码拓扑

文中对串进行编码的字符集Ω取为{A-I，0-6，(，)}。网络中的节点对应于字符集中的一个字母，并可视作最小的模块。较大的模块可以用括号括起来的节点或较简单的模块来表达。数字表示模块间的连接，例如数字x表示前件模块与向右距离为x的模块有权相连。因为我们仅考虑前向网络，所以节点间的连接是单向的，模块间的连接是指前模块的输出节点和后模块的输入节点相连，其中一个模块的输入节点是指在本模块内不存在任何节点的输出权与其相连；输出节点是指在本模块内不存在任何节点的输入权与其相连。我们认为在局部区域有效的规则在较大区域同样有效，上述编码有利于重复、递归利用好的产生式规则，并最终产生类似于分形的网络结构。

若在字符集Ω允许取任意的连接间隔x，由字符集通过产生式重写即可产生所有可能的网络结构。我们取最大连接间隔为6，一是可以约束网络自由度，二是可以加快网络空间搜索的收敛速度。

2.2 产生式规则

网络描述串是通过对初试串（原子串）反复应用L-系统产生式重写规则而得到的。我们所用的L-系统为双向系统：每一个规则可包括左上下文和右上下文，其一般格式如下：

L〈P〉R→S

式中四个元素均属于字符集Ω，并满足下列条件：1）前件元P可包含数字和模块，并至少包含一个字母；2）S的限制同P，只是可以为空；3）上下文L和R可为空，不允许存在游离的数字，数字须包含在模块中。

与传统L-系统不同，新方法处理上下文是必须考虑连接间隔、上下文的获取不是直接对前件元的左右符号进行匹配，还要考虑所表达的网络的实际结构。上下文是以模块为元素的串，其中的每一模块连接到应用重写规则前的前件元，左上下文即为输出（或部分输出）到前件元的模块（或节点）集，右上下文可同理定义。较传统系统，这里的上下文匹配的范围更广。

对图1中的产生式规则，取A为一原子串，重写规则如下：

A→B0B0B→[CD]0[CD]0C→[CC1]0[CC]0C

利用上述规则得到的网络如图2。重复调用同一规则的调用上限为6。

为了进行上下文匹配计算，每一步的迭代结果都转化为一邻接矩阵，最后再对邻接阵进行剪枝处理去除孤立和无效的节点。

1: A ->B0B0B
2: B>B->[CD]
3: B ->C
4: C<D ->C
5: D>D->C1

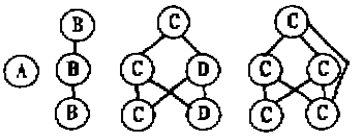

图1 产生式规则　　图2 生成网络

3 并行遗传算法

3.1 遗传编码

搜索有效的网络结构也是应用遗传算法对L-系统的产生式规则进行定义和优化的过程，首先要做的是对产生式（L-系统）进行染色体编码。字符集中的每个字符用一个六位二进制串来编码，总共包括Ω中的18个字符和一个"*"号分割符。由染色体抽取产生式规则的过程如下：

1）以6bits为一单位，读入染色体；
2）通过转换表将每个6位串转化为*号或Ω中的一个字符，从而得到一字符串；
3）查找所有包括5个*号的最小子串（每个子串以*号开头和结尾）；
4）每个子串即为一个产生式规则，两个*号之间的字符子串即为一个产生式规则的元；例如，串"A*BD*C**"对应的产生式规则为A>BD->C；两个相邻的*号表示一个空元素；
5）删除不满足2.2节所规定条件的约束；
6）以第2-5位分别作为起始位，重复1)-5)；
7）从最末位开始，反向读入位流，并重复1)-6)。

3.2 并行进化

利用遗传算法产生多个初始位串，由这些位串（染

色体)组成进化初始群体;算法的解即为解码后的L-系统所产生的网络结构.选定一特定问题,通过上述算法进行训练求解,网络输出的误差平方和经过变换即为遗传算法所需的适应值。

遗传算法的性能对其参数选择有一定的依赖性,对解的质量也会有所影响[12~14]。例如,将训练代数定为100次时所得最优网络结构,在训练代数增加时可能不再是最优的。关于这个问题的解决办法将这些参数进行编码加入进化过程,这显然会增加遗传算法的搜索空间;另一办法是放宽对参数的限制,利用并行算法提高搜索效率。

本文中的分层并行算法主要分为六个步骤:问题分解、编码、初始化、染色体分组、任务分配和迭代运算,主要特点是引入辅助处理单元,构成层次型通讯拓扑结构,提高了算法的处理单元容量。详细算法如下:

1)*问题分解* 根据问题特征,将一复杂问题Q分解为几个较简单问题 q_i,$i=1,2,\cdots,N$。

2)*编码* 对每一子问题 q_i 选择染色体编码方案:$q_i \rightarrow c_i$,$(i=1,2,\cdots,N)$,则原问题的编码即为 $C=c_1 \oplus c_2 \oplus \cdots \oplus c_N$,进一步确定C的适应度函数F(C),⊕为染色体编码串联结操作符。

3)*初始化* 基于子编码 c_i,$(i=1,2\cdots,N)$,随即生成N个初始群体:$PC_i(0)=\{C_{ij}\}$,其中 i(1,2,…,N)为子群体标号,j(1,2,…,k)为子群体内染色体编号。

4)*染色体分组* 将 $PC_i(0)$ 划分为M组:$PS_k(0)$,$(k=1,2,\cdots,M)$,第k组染色体个数为 N_k 个,并有

$$K=N_1+N_2+\cdots+N_M$$

原问题编码又可表示为 $PS(0)=\{PS_k(0),k=1,2,\cdots,M\}$,其中

$$PS_k(0)=PS_{1k}(0)\oplus PS_{2k}(0)\oplus\cdots\oplus PS_{Nk}(0)$$

5)*任务分配* 选定一个主处理单元 U_0,N个辅助处理单元 U_i($i=1,2,\cdots,N$)和M×N个从处理单元 U_{ij}($i=1,2,\cdots,N$;$j=1,2,\cdots,M$),对初始群体按如下方案进行任务分配:

$PC(0)\Rightarrow U_0$。

$PS_k(0)\Rightarrow U_k$,$k=1,2,\cdots M$

$PS_{ij}(0)\Rightarrow U_{ij}$,$i=1,2,\cdots,N$;$j=1,2,\cdots,M$

由 U_i 分别从各自的群体中随机选取一染色体

$$PP_i(0)=PS_{1k(1)}(0)\oplus PS_{2k(1)}(0)\oplus\cdots\oplus PS_{Nk(1)}(0)$$

作为代表染色体发送给 U_i 的从处理单元 U_{ik},$k=1,2,\cdots,M$。

6)*迭代运算* 各处理单元分别对各自的子群体进行遗传算子操作,个体的适应值评估由下式确定(其中g表示进化的代数,$PP_{ij}(g)$表示 U_{ij} 中的代表染色体):

$$f^g(PS_{ij}(g))=F(PP_{1j}(g-1)\oplus PP_{2j}(g-1)\oplus\cdots\oplus PS_{ij}(g)\oplus\cdots\oplus PP_{Nj}(g-1)) \quad (1)$$

①基于标准遗传算法,对各子群体 $PS_{ij}(g)$ 进行遗传操作运算;

②由(1)式计算 $PS_{ij}(g)$ 的适应值,若满足终止条件则结束,否则继续;

③由辅助处理单元 U_k 搜集各自从处理单元的最优个体组:

$$PP_i(g)=\{PP_{1i}(g-1)\oplus PS_{2i}(g-1)\oplus\cdots\oplus PP_{ki}(g)\oplus\cdots\oplus PP_{Ni}(g-1),k=1,2,\cdots,N\}$$

④由主处理单元 U_0 搜集各自辅助处理单元的个体,并基于标准遗传算法进行遗传操作,然后将结果返还给各辅助处理单元,作为新的代表染色体组;

⑤辅助处理单元将新一组代表染色体组转发给各自的从处理单元,转①。

4 仿真结果

我们的仿真设计实例是一个字符识别问题,每个字符由3×3个像素组成,并可旋转90、180或明或270°,如图3所示,每个字符可放置在一个4×4的网格中。单机实验得出的一个优化的网络如图4,其中 G_i 表示含 i 个节点的模块(网络仅含13个输入单元而不是16,多出的三个输入被忽略)。对图4中的网络进行训练,输入32个样本,训练次数为200次,并重复执行上述训练40次,网络正确识别率为95%;作为对比,将同样的样本对一个标准单隐层的BP网络进行训练、隐节点个数从5到20可变,最高识别率仅为60%。上述计算的网络学习参数取为0.5、动量项设为0.85。

我们的网络实验环境选用10Mbits/sec的局域网,工作站为Pentium机型,在这种环境下的通讯时间的经验公式为 $T_c=20+0.00526x$,单位为毫秒,x为消息的字节数。从处理单元数目分别取2、4、8和16,实验结果见表1,从中看出新算法在处理单元个数不大于8时可以实现线性加速;当处理单元个数继续增加时,因为网络通讯开销的增加,算法的整体性能反而会下降。

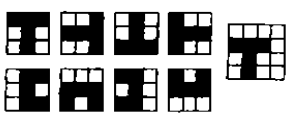

图3 待识别字符集和一个样本输入

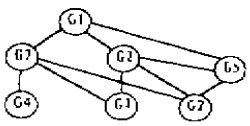

图4 TC识别问题的最优网络

表1 计算时间列表 (单位:秒)

从处理单元个数	0	2	4	8	16
收敛时间	456	289	216	118	201

结论 由实验仿真结果可以看出,利用并行遗传算法和L-系统相结合能有效提高解的收敛速度和解的质量。我们认为这主要得益于如下几点:1)利用产生式方法比对整个权空间进行编码的方法有效地降低了

(下转第31页)

机并绘制出来。

结论 与传统系统相比，基于 Agent 的软件可靠性评估系统具有以下特点：

(1)可靠性模型选择和决策的科学性和合理性。系统为故障数据选择合适的可靠性模型是"二阶段决策"过程：第一阶段是 SRMA 在接收到故障数据时根据数据外部特性来判断模型是否可处理该类数据，在模型运行后根据预测和观测失效行为之间的偏差来客观评价自身的预测准确性；第二阶段是 SREA 在收到所有模型返回的结果后对各模型预测的精确性进行比较，判断哪个模型更适用。SREA 综合以上两个阶段的结果选择出最适用的模型。该决策过程是由请求方和服务方合作共同完成的，比起单方面决策更科学和合理。

(2)系统结构具有较高的灵活性。系统中的 SREA 和 SRMA 能够对其内部状态和行为实施完全自主的控制，具有高度自治性和独立性。从软件开发的角度上看，这种软件系统的模块性、可维护性、可重用性也就更好。在要增加新模型时，只需开发一个新的 SRMA 加入到系统环境中并进行注册，无须对原有系统进行改动，而且能充分重用已有模型。

(3)系统适合在分布式环境中应用，实现了并发和共享。由于 SRMA 具有驻留性、反应性以及独立性等特征，所以它们的行为可以并发地运行。从软件结构上看，它是一个多线程的系统。在网络和分布式环境中，各 SRMA 可以驻留在不同的主机上，能够充分地利用自身资源。多个用户还可以对它们共享使用。

(4)系统通过问题分解简化了求解过程。系统利用了分布式人工智能中合同网的求解方式，把为故障数据选择合适模型并进行可靠性评估的问题分解到 SREA 和 SRMA 中来实现，降低了求解的复杂性。整个系统通过实体之间交互来实现合作、协同和竞争，从而完成整个问题的求解。

总体说来，基于 Agent 的软件可靠性评估系统与传统软件可靠性评估系统相比具有很多优越性，与当前迅速发展的网络与分布式环境结合起来，可以对软件可靠性工程的应用和发展起到很大的促进作用。

参考文献

1 Lyu M R Handbook of Software Reliability Engineering. IEEE Computer Society Presss，1996

2 Wooldridge M，Jennings N R. Intelligent Agents：Theory and Practice. Knowledge Engineering Review，1995，10(2)：115～152

3 Nwana H，Lee L. Coordination in Software Agent Systems. BT Technology Journal，1996，14(4)

4 Musa J D，et al. Software Reliability Measurement，Prediction，Application. McGraw-Hill Book Company，1987

5 Prieto-Diaz. Status Report：Software Reusability. IEEE Software，1993，10(3)，61～66

6 徐仁佐，郑仁杰，等. 软件可靠性模型及应用. 清华大学出版社，广西科技出版社，1994

(上接第45页)

搜索空间的维数；2)解的模块化复制策略，以及对有效的规则的重用，降低了计算的复杂度；3)采用由简单到复杂的设计方法明显优于直接对复杂问题进行优化；4)采用并行处理机制和问题分解处理方法来直接提高算法收敛的速度。

参考文献

1 Rumelhart D E，et al. Learning internal representations by error propagation，Parallel Distributed Processing：Explorations in Microstructures of Cognition，1986，1(1)：318～362

2 Tanese R. Distributed genetic algorithms. In：Schaffer J D，ed. Proc. of the Third Interl. Conf. on Genetic Algorithms. San Mateo，CA：Morgan Kaufmann，1989：434～439

3 Dodd N. Optimization of network structure using genetic algorithms. In：Widrow B，Angeniol B，eds. Proc. of the Interl. Neural Network Conf.，INNC-90-Pairs. Kluwer，Dordrecht，1990：693～696

4 Harik G，et al. The gambler's ruin problem，genetic algorithms，and the sizing of populations. In：Back T，ed. Proc. of the Fourth Interl. Conf. on Evolutionary Computation. New York：IEEE Press，1997：7～12

5 Jacobs R A，et al. Task decomposition through competition in a modular connectionist Architecture，the what and where vision tasks. Cognitive Science，1991(15)：219～250

6 Mandelbrot B B. The fractal geometry of nature. Freeman，San Francisco，1982

7 Murre J M J. Categorization and learning in neural networks. Modelling and implementation in a modular framework. Dissertation，Leiden University，1992

8 Posner M I，et al. Localization of cognitive operations in the human brain. Science，1988(240)：1627～1631

9 Prusinkiewicz P，Hanan J. Lindenmayer systems，fractals and plants. Springer-Verlage，New York，1989

10 Whitley D，Starkweather T. Genitor Ⅱ：A distributed genetic algorithm. Journal of Experimental and Theoretical Artificial Intelligence，1990

11 Solla S A. Learning and generalization in layered neural networks：the contiguity problem. In，Personnas L，Dreyfus G，eds. Neural networks，from models to applications. I. D. S. E. T，Paris，1989：168～177

12 王洪燕，杨敬安. 并行遗传算法研究进展. 计算机科学，1995(6)

13 王洪燕，杨敬安. 并行进化 BP 神经网络. 合肥工业大学学报，1993(3)

14 王洪燕，杨敬安. 关于主从式分层并行遗传算法的研究. 见：第五届全国计算机应用联合学术会议论文集，1999