基于AGA-LVQ神经网络的软件可靠性预测模型研究

乔 辉 周雁舟 邵 楠 高 杨 粟登银
(中国人民解放军信息工程大学电子技术学院 郑州 450004)

摘 要 针对当前大多数软件可靠性预测模型预测准确率不高等问题,利用LVQ神经网络的非线性运算能力和自适应遗传算法(AGA)的参数寻优能力,提出了一种基于AGA-LVQ的软件可靠性预测模型。首先对待预测的数据用主成分分析(PCA)等方法进行预处理以降低维度,去除冗余和错误数据,然后根据自适应遗传算法来计算最优的LVQ神经网络初始权值向量,最后运用LVQ神经网络进行软件可靠性预测实验。通过与传统方法的对比,证明该方法具有较高的预测准确率。

关键词 软件可靠性预测,模式识别,LVQ神经网络,自适应遗传算法,主成分分析

中图法分类号 TP311.5 **文献标识码** A

Research of Software Reliability Prediction Model Based on AGA-LVQ

QIAO Hui ZHOU Yan-zhou SHAO Nan GAO Yang SU Deng-yin
(Institute of Electronic Technology, PLA Information Engineering University, Zhengzhou 450004, China)

Abstract The prediction accuracy of most current software reliability prediction models is not high. This paper put forward a software reliability prediction model based on AGA-LVQ, which takes advantage of non-linear computing power of the learning vector quantization (LVQ) neural network and parameter optimization capability of the adaptive genetic algorithm (AGA). Firstly, principle components analysis (PCA) preprocessing was used to reduce the dimension of the metrics and remove the redundancy and error data. Secondly, AGA was used to calculate the optimal initial vector weights of the LVQ neural network. Lastly, LVQ neural network was used to do the software reliability prediction experiments. The experiment results indicate that the method has a higher prediction precision than the traditional software reliability prediction model.

Keywords Software reliability prediction, Pattern recognition, LVQ neural network, AGA, PCA

1 引言

目前国内外学界提出了很多种不同的软件可靠性预测模型。在现有的各种模型中,基于分类的模型已经被证明可以更好地实现软件可靠性预测[1]。

近年来,用人工神经网络(Artificial Neural Network, ANN)来实现基于分类的软件可靠性预测已经成为研究热点[2]。国际上,Kiran利用小波神经网络对软件可靠性进行预测取得了不错的效果[3]。Khoshgoftaar将神经网络方法应用于Nortel的大规模通信软件的开发过程中,显著地提高了通信软件的可靠性和质量[4]。Tong-Seng Quah使用神经网络进行风险模块的早期识别[5]。国内吴超等人[6]提出了一种利用PCA-BP模糊神经网络的软件可靠性预测模型。胡恒章教授[7]所率领的研究小组将神经网络的方法应用于组合导航软件的可靠性预测。

虽然研究人员提出了很多种基于神经网络的软件可靠性预测模型,但模型在具体的应用中仍然存在许多问题。Kohenen[8]于1990年提出的学习向量量化(Learning Vector Quantization)神经网络可以对所有的聚类中心加以监督学习,其算法结构简单,易于实现,已经广泛应用到各行各业中,但其尚未涉及到软件可靠性的领域,且常用的LVQ神经网络存在神经元未被充分利用以及算法对初值敏感等问题。遗传算法(GA)[9]由美国Michigan大学的Holland教授于1967年提出,具有较好的并行空间搜索能力,可以较快地接近全局最优解,是解决大规模组合优化问题的常用算法。针对简单遗传算法存在着寻优时稳定性不高等缺陷,本文采用了改进的自适应遗传算法(AGA)来解决LVQ算法存在的问题。

为了提高软件可靠性预测模型的预测能力,本文提出了一种基于AGA-LVQ的软件可靠性预测模型,即利用LVQ建立软件可靠性预测模型,用AGA优化其初始权值,并基于

到稿日期:2012-03-25 返修日期:2012-06-18

乔 辉(1988-),男,硕士生,主要研究方向为软件可靠性预测,E-mail:sparlk@126.com;周雁舟(1971-),男,副研究员,硕士生导师,主要研究方向为可信计算、操作系统安全及系统工程;邵 楠(1988-),男,硕士生,主要研究方向为软件测试;高 杨(1987-),男,硕士生,主要研究方向为系统工程;粟登银(1986-),男,硕士生,主要研究方向为软件缺陷仿真。

主成分分析方法进行降维处理，最后利用 AGA-LVQ 算法建立了软件可靠性预测模型。

2 相关研究

2.1 LVQ 神经网络

LVQ 神经网络的基本思想是，对于来自训练集的任一模式向量 X，如果 X 与最近的模板属于同一类，则无需学习；否则，将“惩罚”分类错误的模板，“奖励”其对应正确类别的模板。若经过若干次迭代后，所得到的向量量化不再明显变化，从而实现模式识别，因此简单易行。其 LVQ 结构如图 1 所示。

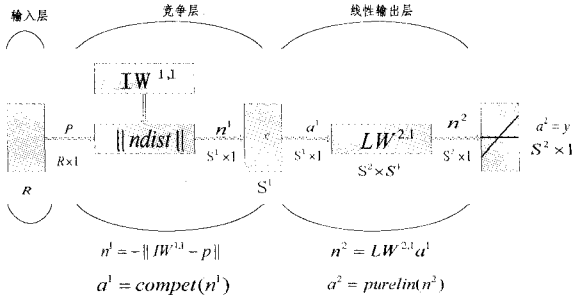

图 1 LVQ 神经网络结构图

在图 1 中，p 为 R 维的输入模式；S^1 为竞争层神经元的个数；$IW^{1,1}$ 为输入层与竞争层之间的连接权系数矩阵；n^1 为竞争层神经元的输入；a^1 为竞争层神经元的输出；$LW^{2,1}$ 为竞争层与线性输出层之间的连接系数矩阵；n^2 为线性输出层神经元的输入；a^2 为线性输出层神经元的输出。竞争层和线性层的每一个神经元的输出都对应一个分类（子分类或目标分类）结果，所以竞争层通过学习可以得到 S^1 类子分类结果；然后，线性层将 S^1 类子分类结果再分成 S^2 类目标分类结果（S^1 始终大于 S^2）。

2.2 自适应遗传算法

20 世纪 80 年代，Holland 教授实现了第一个基于遗传算法的机器学习系统，开创了遗传算法的机器学习的新概念。遗传算法利用生物遗传学的观点，在解空间随机产生多个起始点并同时开始搜索，由适应度函数来指导搜索方向，是一种能够在复杂搜索空间快速寻求全局优化解的搜索技术，目前已在优化、机器学习和并行处理等领域得到越来越广泛的应用。

简单遗传算法存在着寻优时稳定性不高、局部搜索能力不强、收敛速度较慢，且容易产生早熟现象等缺陷。本文利用自适应遗传算法进行 LVQ 神经网络的初始权值向量寻优，使得种群进化更加稳定，避免陷入局部最优。同简单遗传算法相比较，自适应遗传算法主要对交叉和变异操作的概率计算方法加以优化，其详细过程可参考文献[10]。

2.3 主成分分析算法

近年来，许多人对输入向量压缩降维问题进行了深入的研究，取得了一定的成果。常用的变量压缩方法[11]有多元回归与相关分析法、类逐步回归法、主成分分析法、独立成分分析法等。

本文将主成分分析方法（PCA）[12]引入数据的分析过程，主成分分析算法是由 Hotelling 提出的。一般地，在软件度量中，N 个软件模块就是 n 个样品，而考察的 p 个属性对则构成了 p 个变量 $x_1, x_2, \cdots, x_n (n>p)$。这样，原始统计资料整理的原始数据矩阵为

$$X=\begin{bmatrix} x_{11} \cdots x_{1p} \\ x_{n1} \cdots x_{np} \end{bmatrix}$$

为了消除由于量纲的不同可能带来的一些不合理的影响，在进行主成分分析之前先对数据进行归一化处理，即将数据归一到[−1,1]区间内。之后将 $x=(x_1, x_2, \cdots, x_p)'$ 的 p 个变量综合成 p 个新变量，新的综合变量可以由原来的变量 $x_1, x_2, \cdots, x_n (n>p)$ 线性表示，即

$$\begin{cases} y_1=u_{11}x_1+u_{12}x_2+\cdots+u_{1p}x_p \\ y_2=u_{21}x_1+u_{22}x_2+\cdots+u_{2p}x_p \\ \cdots \\ y_p=u_{p1}x_1+u_{p2}x_2+\cdots+u_{pp}x_p \end{cases}$$

并且满足 $u_{k1}^2+u_{k2}^2+\cdots+u_{kp}^2=1$，其中系数 u_{ij} 由下列原则来确定。

(1) y_i 与 $y_j (i\neq j; i,j=1,2,\cdots,p)$ 相互无关；

(2) y_1 是 $x_1, x_2, \cdots, x_p$ 的一切线性组合中方差最大者；y_2 是与 y_1 不相关的 $x_1, x_2, \cdots, x_p$ 的所有线性组合中方差最大者，以此类推。

如此决定的综合变量 $y_1, y_2, \cdots, y_p$ 分别称为原变量中第 1，第 2，…，第 p 个主成分。其中 y_1 在总方差中占的比重最大，第 1 个主成分 y_1 的方差贡献率最大，依次递减。为了简化系统结构，本文只挑选前几个方差贡献率最大的主成分，而不是取所有的 p 个主成分。一般地，前几个方差的累积贡献率超过 85% 即可。

3 软件可靠性预测模型的建立

3.1 软件可靠性预测模型

基于 AGA-LVQ 方法建立的软件可靠性预测模型主要包括以下几个步骤，如图 2 所示。

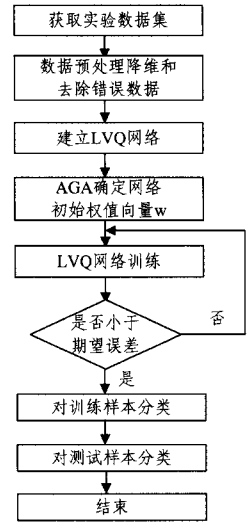

图 2 建立软件可靠性预测模型的流程

其具体步骤如下：

1. 获取软件可靠性度量数据集；

2. 对数据集进行主成分分析来降维，同时消除存在的错误数据，提高预测精度；

3. 用训练样本基于 LVQ 神经网络建立软件可靠性预测模型；

4. 用自适应遗传算法来进行 LVQ 神经网络初始权值的优化，完成网络的构建；

5. 将测试样本输入训练好的 LVQ 神经网络，来进行分类预测；

6. 实验结束，输出软件可靠性分类结果。

3.2 LVQ 神经网络创建

数据采集完成后，利用 LVQ 算法可创建一个 LVQ 神经网络，本文根据自适应遗传算法对网络中的参数进行设定，使其适用于训练和测试的需要。

LVQ 神经网络预测算法的基本思想是：计算距离输入向量最近的竞争层神经元，从而找到与之相连接的线性输出层神经元，若输入向量的类别与线性输出层神经元所对应的类别一致，则对应的竞争层神经元权值沿着输入向量的方向移动；若两者的类别不一致，则对应的竞争层神经元权值沿着输入向量的反方向移动。基本的 LVQ 算法步骤[13]为：

步骤 1　初始化输入层与竞争层之间的权值 w_{ij} 及学习率 $\eta(\eta>0)$。

步骤 2　将输入向量 $X=(x_1,x_2,\cdots,x_R)^T$ 送入到输入层，并根据式(1)计算竞争层神经元与输入向量的距离：

$$d_i=\sqrt{\sum_{j=1}^{R}(x_j-w_{ij})^2} \tag{1}$$

式中，$i=1,2,\cdots,S^1$。

步骤 3　选择与输入向量距离最小的竞争层神经元，若 d_i 最小，则记与之连接的线性输出层神经元的类标签为 C_i。

步骤 4　记输入向量对应的类标签为 C_X，若 $C_i=C_X$，则根据式(2)调整权值；否则，根据式(3)进行权值更新：

$$w_{ij\_new}=w_{ij\_old}+\eta(x-w_{ij\_old}) \tag{2}$$

$$w_{ij\_new}=w_{ij\_old}-\eta(x-w_{ij\_old}) \tag{3}$$

4 基于 AGA-LVQ 神经网络的软件可靠性预测模型实验

4.1 实验环境及实验数据

实验所用算法利用 MATLAB2008a 的 LVQ 神经网络和遗传算法工具箱搭建，程序在配置为 Pentium(R) 4 CPU 3.00GHz 2.99GHz，504MB 内存的电脑上运行。

实验数据来源于 NASA 的 MDP(Metric Data Program) 软件度量项目中的 JM1 数据集，可从网址 http://mdp.ivv.nasa.gov 中下载。JM1 数据集来自于一个地面预测系统的工程，用 C++语言编写代码，包含了 10879 个子程序。

4.2 实验数据预处理

NASA 的软件度量数据集已经被广泛应用于研究中，但数据集本身可能存在的问题却很少有学者直接涉及研究。本文参考英国赫特福德大学计算机科学学院的 David Gray 教授的最新研究成果[14]，其在文献中指出了 NASA 的数据集中存在着大量的“重复数据”和“矛盾数据”，经过研究证明，这些数据已经严重影响了数据的实验结果。

本文首先对 JM1 数据集进行了删除“重复数据”和“矛盾数据”的操作，经过实验，最终将其子程序的个数删减为 8907，将其度量属性由 22 个减少为 13 个。

接着对样本数据运用最小-最大规范化方法进行归一化处理，将数据样本的每个属性的值映射到[0,1]区间，以消除因量纲不同可能引发的影响。

最后对经过归一化操作的数据运用 SPSS 分析工具进行主成分分析降维，以消除样本数据中的相关性，得出的最终实验仿真结果如表 1 所列。

表 1　主成分分析实验结果

Component	Initial Eigenvalues	Extraction Sums of Squared Loadings		
	Cumulative %	Total	% of Variance	Cumulative %
1	64.243	13.491	64.243	64.243
2	72.209	1.673	7.966	72.209
3	88.556	1.663	7.920	88.556

由表 1 的结果可以看出，3 个主要成分的累积贡献率为 88.556%，即保留了原始数据 88.556%的信息，具有显著代表性。F1 主成分权重最大，为 64.243%，是最重要的影响因子。在该文所做的实验中，PCA 方法求出的 3 个主要成分将作为 GA-LVQ 的输入，实验数据维数得到了降低且保留了原始数据 88.556%的信息，并达到了一个比较好的结果。实验数据维数的降低将会简化 LVQ 结构，加快运算速度，并且原始数据的基本信息得到了保留，不会造成实验结果的失真。

4.3 参数寻优

采用 MATLAB2008a 编制自适应遗传算法，种群大小设置为 40，进化代数设置为 200。经过多次实验，获得最好的遗传算法的交叉概率为 $p_{c1}=0.95$，$p_{c2}=0.67$，变异概率为 $p_{m1}=0.05$，$p_{m2}=0.01$，获得的 LVQ 神经网络相关最优参数中期望误差为 0.22，训练速率为 0.12，隐含层神经元为 30 个。

4.4 实验运行

采用 MATLAB2008a 中的 LVQ 神经网络来进行软件可靠性预测实验，运用伪随机数抽样方法[15]选取了 1000 个样本作为实验数据集，其中 700 个为训练样本，300 个为预测样本。

本文实验所采用的评价指标为：

准确度(Accuracy)：预测结果和实际结果相符合的模块个数占整个测试集的比例。

$$accuracy=\frac{TP+TN}{C}$$

查准率(Precision)：预测为易错的模块中实际为易错的模块所占的比例。

$$precision=\frac{TP}{TP+FP}$$

查全率(Recall)：易错模块被正确识别的比例。

$$recall=\frac{TP}{P}$$

F-度量(F-Measure)：Precision 和 Recall 的调和平均数。

$$F1=\frac{2}{\frac{1}{presion}+\frac{1}{recall}}$$

实验运行结果如图 3 所示。

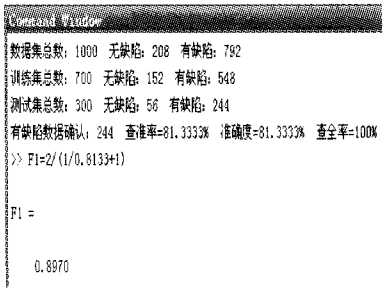

图 3 AGA-LVQ 神经网络软件可靠性预测结果

4.5 实验结果与分析

将基于本文方法(AGA-LVQ)和 Fisher 线性判别方法(LDA)、聚类分析(CA)、BP 神经网络和逻辑回归(LR)等方法建立的软件可靠性预测模型进行比较,结果如表 2 所列。

表 2 本文方法与传统方法对比

预测模型	准确度(%)	查准率(%)	查全率(%)	F1 值(%)
LDA	55.1	56.4	55.7	56.5
CA	72.7	74.6	75.3	75.0
BPNN	80.4	81.2	83.5	82.3
LR	85.3	87.6	90.7	89.3
AGA-LVQ	81.3	81.3	100	89.7

从表 2 可以看出,基于 LDA 方法建立的软件可靠性预测模型效果很差,因为该方法是基于线性分类函数进行分类的,而软件复杂性度量值之间基本都是非线性的;基于 CA 和 BPNN 方法的预测性能明显好于 LDA 方法,但 CA 方法需要很多的先验知识;BPNN 网络结构选择尚无统一的理论指导,且预测效果比 AGA-LVQ 方法仍有不足;基于 LR 方法的预测性能比较好,且一些指标甚至优于 AGA-LVQ,但该方法只有样本容量大时能有很好的预测效果,且预测时间较长。

本文利用 LVQ 神经网络算法结构简单,其竞争层将自动学习对输入向量进行分类,同时 LVQ 神经网络算法建立的决策区是近似最优的等优点进行软件可靠性预测建模;利用主成分分析法对其输入数据进行预处理以减少计算成本,去除冗余数据;利用自适应遗传算法优化 LVQ 神经网络的初始权值,加速了网络收敛速度,提高了网络分类精度,很好地实现了软件可靠性预测。

结束语 本文的贡献是针对软件可靠性预测问题提出了一种新颖的基于 AGA-LVQ 算法的软件可靠性预测模型。LVQ 神经网络在实现模式识别方面简单易行,其已经广泛应用到各行各业中,如故障诊断、性能评价、风险预测等,但还没有应用于软件可靠性预测中。本文很好地将 LVQ 神经网络与软件可靠性预测联系起来;对数据进行的预处理消除了相当一部分"重复数据"和"矛盾数据",这是以往应用 NASA 的 MDP 项目中的数据不被研究者所关注的问题;应用 PCA 方法缩减输入向量维数;应用 AGA 算法迅速获取了最佳的 LVQ 神经网络初始权值向量,从而加速了网络收敛速度,提高了网络分类精度。实验结果表明,该方法比传统方法具有更好的预测速度和预测效果,但其在分类的准确率等方面仍有不足,这是下一步的研究课题。

参 考 文 献

[1] Challagulla V UB, Bastani F B, Yen I-L, et al. Empirical assessment of machine learning based software defect prediction techniques[C]// Proceedings of the 10thIEEE International Workshop on Object-Oriented Real-Time Dependable Systems. Washington DC, USA, 2005: 263-270

[2] Hu Q P, Xie M, Ng S H. Early Software Reliability Prediction with ANN Models[C]// 12th Pacific Rim International Symposium on Dependable Computing(PRDC'06). IEEE, 2006

[3] Kiran N R, Ravi V. Software Reliability Prediction using Wavelet Neural Networks[C]// International Conference on Computational Intelligence and Multimedia Applications. 2007

[4] Khoshgoftaar T M, Allen E B, Hudepohl J P, et al. Application of Neutral Networks to software quality modeling of a very large telecommunications system[J]. IEEE Transactions On Neutral Network, 1997, 8(4): 902-909

[5] Quah T-S, Mie Mie, Thwin T. Application of neural networks for software quality prediction using object-oriented metrics[C]// Proceedings of International Conference on Software Maintenance. 2003: 116-125

[6] 吴超,许建平,陈丽容. 基于生命周期的软件缺陷预测技术[J]. 计算机工程与设计,2009,30(12):2956-2959

[7] 张家海,胡恒章. 组合导航系统可靠性的神经网络静态预测[J]. 哈尔滨工业大学学报,2002:34(4)

[8] Kohenon T. The self-organizing map[J]. IEEE, 1990(78): 1464-1480

[9] 张文修,梁怡. 遗传算法的数据基础[M]. 西安:西安交通大学出版社,2000

[10] 张玲,刘勇,何伟. 自适应遗传算法在车牌定位中的应用[J]. 计算机应用,2008,28(1):185

[11] Menzies T, Ammar K, Nikora A, et al. How Simple is Software Defect Prediction[J]. Journal of Empirical Software Engineering, 2003(10)

[12] 张靖,葛玮,郝克刚. 软件度量中主成分分析方法的研究[J]. 计算机技术与发展,2006,12:144-148

[13] 史峰,王小川,郁磊,等. MATLAB 神经网络 30 个案例分析[M]. 北京:北京航空航天大学出版社,2011

[14] Gray D, Bowes D, Davey N, et al. The Misuse of the NASA Metrics Data Program Data Sets for Automated Software Defect Prediction[OL]. http://uhra.herts.ac.uk/dspace/bitstream/2299/6363/1/905745.pdf

[15] 高书亮,杨东凯,黄智刚. Galileo 系统伪随机序列生成及其 FPGA 实现[J]. 微计算机信息,2008,24(26):124-125