基于GMDH因果关系的软件缺陷预测模型

张德平 刘国强 张 柯
(南京航空航天大学计算机科学与技术学院 南京 210016)

摘 要 软件缺陷预测是软件可靠性研究的一个重要方向。基于自组织数据挖掘(GMDH)网络与因果关系检验理论提出了一种软件缺陷预测模型,借鉴Granger检验思想,利用GMDH网络选择与软件失效具有因果关系的度量指标,建立软件缺陷预测模型。该方法从复杂系统建模角度研究软件度量指标与软件缺陷之间的因果关系,可以检验多变量之间在非线性意义上的因果关系。最后基于两组真实软件失效数据集,将所提出的方法与基于Granger因果检验的软件缺陷预测模型进行比较分析。结果表明,基于GMDH因果关系的软件缺陷预测模型比Granger因果检验方法具有更为显著的预测效果。

关键词 软件缺陷,因果关系,软件度量,GMDH网络,Granger检验
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.7.031

Software Defect Prediction Model Based on GMDH Causal Relationship

ZHANG De-ping LIU Guo-qiang ZHANG Ke
(College of Computer Science and Technology, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, China)

Abstract Software defect prediction is an important aspect in the field of software reliability research. In this paper, we presented a software defect prediction model based on GMDH networks and causal test theory. The model selects the software metrics with the defect causal relationship by learning Granger test ideas and uses the GMDH network which can check the non-linear causality between multiple factors of software defect. Finally, based on two real software failure data sets, we designed an experiment to compare the proposed method with the Granger test software defect prediction model. The experiment results show that the proposed model is more effective and efficient than Granger test software defect prediction model.

Keywords Software defect, Causal relationship, Software metrics, GMDH network, Granger test

1 引言

随着计算机技术在航空航天、国防、金融、能源和通信等诸多领域中的广泛应用,软件质量日益受到普遍关注。软件缺陷预测作为保证软件质量的重要方面,已成为当前软件工程研究领域的热点[1,14]。软件缺陷预测的主要目的是预测软件还存留的缺陷,根据软件的基本属性(规模、复杂度、开发方法、过程等)、软件已经发现的缺陷来预测软件可能还遗留的、尚未发现的缺陷。合理预测软件缺陷可以统计尚未发现但仍存在的软件缺陷数目及软件缺陷的分布,这样可以有效地帮助测试人员快速、准确地定位并纠正软件缺陷,客观评价软件测试结果,可显著减少软件开发成本并提高软件可信性,为保证和提高软件质量起着非常重要的作用。

一般地,软件缺陷预测技术可分为静态和动态两种缺陷预测技术[1-3]。典型软件缺陷预测模型包括基于软件规模、软件复杂度、多维软件度量元等因素的缺陷预测模型[4],基于线性判别分析(LDA)、布尔差别函数(BDF)、K近邻分类回归(K-NN)、支持向量回归机(SVM)等方法的缺陷分布预测模型,以及基于软件质量估算、缺陷移除率、Bayesian网络和"捕捉"估算等方法对软件缺陷进行预测的模型等。这些模型的实质都是基于度量指标与软件缺陷之间的关系进行分析,如相关性[5,6]、相依性[7,8]、一致性[9]、因果关系[10-13]等,利用统计分析技术建立输入、输出变量之间的函数关系,建立相应的预测模型进行软件缺陷预测。在这些软件缺陷预测技术中,基于因果关系分析的软件缺陷预测是使用最为广泛的一类预测技术。由于所采用的预测模型各不相同,其因果关系定义也有所不同,其中最典型的是用条件概率来描述软件缺陷与度量指标之间因果关系的Bayesian因果关系[10]。为了解决描述Bayesian因果关系的条件概率难于计算的不足,Couto等[12,13]基于Granger因果检验定义了一种新的软件缺陷与软件度量指标之间的因果关系,利用自回归模型建立软件缺陷预测模型。然而,使用Granger因果关系检验来研究软件缺

到稿日期:2015-05-19 返修日期:2015-08-22 本文受中央高校基本科研业务费专项资金(NS2014072)资助。
张德平(1973-),男,博士,主要研究领域为软件测试与软件可靠性建模,E-mail:depingzhang@nuaa.edu.cn;刘国强(1989-),男,硕士生,主要研究领域为软件测试与软件可靠性建模,E-mail:963856778@qq.com;张 柯(1984-),男,硕士生,主要研究领域为软件测试与软件可靠性建模,E-mail:18956841@qq.com。

陷与软件度量元之间的因果关系时具有以下几点不足：1)Granger因果关系检验仅适用于两个因素之间的因果关系检验，因此在检验过程中需要分别对每一个因素与软件缺陷进行检验分析，并假设不存在其它因素的影响，最终将所有具有因果关系的因素统一分析，但实际情况是影响软件缺陷的因素往往不止两个，而各因素之间同样具有因果关系；2)Granger因果关系检验对序列的严格假设——平稳序列，需要将原始软件度量指标转化为平稳序列，对原始软件度量指标序列进行差分、消除趋势、Box-Jenkins"预白化"等处理，扭曲了原始软件度量指标与软件缺陷序列之间的因果关系；3)Granger因果关系检验适用于大样本问题，对小样本问题检验结果不够理想；4)Granger 因果关系检验对滞后阶数选择非常敏感，不能揭示出各原始软件度量指标与软件缺陷之间因果关系的强弱，其因果关系检验结果推广到其它软件系统的能力较弱等。

针对 Granger 因果关系检验在软件失效预测中的不足，本文提出了一种基于自组织数据挖掘(GMDH)因果关系的失效预测模型。该模型基于 Granger 因果关系定义中根据检验模型预测能力是否显著变化来判断因果关系是否存在的基本思想，给出了用于描述多变量之间因果关系的 GMDH 因果关系，利用 GMDH 数据划分和内、外准则的配合使用，克服影响软件缺陷的因素多、样本少的困难，从而解决 Granger 因果关系检验结果推广能力弱的问题，使之能较好地应用于软件失效与多个软件度量指标之间的因果关系分析问题。

2 基本概念及相关算法

2.1 GMDH 因果关系

Granger 因果检验是利用时间序列的关系来鉴定两个因素是否具有因果关系[15]，其基本原理是，假设系统是由变量 X 和 Y 组成，如果变量 X 有助于预测 Y，即根据 Y 的过去值对 Y 进行回归时，再加上 X 的过去值，能够显著地增强模型的回归解释能力，则称 X 是 Y 的 Granger 原因；如果用 X 的现在值和 X，Y 的过去值来预测 Y，比只用 X 和 Y 的过去值要好，就称 X 是 Y 的瞬时原因，否则称为非 Granger 原因。为了更好地描述软件失效与软件度量指标之间的因果关系，Chow[16]采用对 Granger 因果关系的定义：假设给定信息集 A_t，其中至少包含 (X_t, Y_t)，记 $\overline{A}_t=\{A_s \mid s\leqslant t-1\}$，定义 $\sigma^2(Y_t|\overline{A}_t)$ 为 Y_t 关于 $\overline{A}_t$ 的最优预测方差，则有如下定义。

定义 1 如果 $\sigma^2(Y_t|\overline{A}_t)<\sigma^2(Y_t|\overline{A}_t-X_t)$，则称 X_t 是 Y_t 的 Granger 原因。

借鉴 Granger 因果关系定义，可通过检验模型预测能力是否显著变化来判断各个因素之间是否存在因果关系，即如果变量 Y 的现在值用 X 和 Y 的过去值预测比只用 Y 的过去值预测要好，就称 X 是 Y 的原因。基于此，本文基于 GMDH 模型给出如下 GMDH 因果关系定义。

假定容量为 N 的样本数据可划分为 A、B、C 3 个子集，并且定义 $A\cup B=W$，则样本数据可表示为：

$$y=\begin{bmatrix}y_A\\y_B\\y_C\end{bmatrix},X=\begin{bmatrix}X_A\\X_B\\X_C\end{bmatrix},N_A+N_B+N_C=N \tag{1}$$

定义 2 对于给定的多变量系统，设输入向量(时间序列)为 $X_{l\times m}=(X_1,X_2,\cdots,X_m)^{\mathrm{T}}$，$X_i(i=1,2,\cdots,m)$ 表示第 i 个输入变量所形成的时间序列，样本数据为 W，输出向量(时间序列)为 $Y_{l\times 1}$，其可能的原因信息全部包含在 $X_{l\times m}$ 中。令外准则的一般表达为：

$$\Delta^2_{Y=f(X)}(H)=\Delta^2(H|G)=\|Y_H-X_H\hat{a}_G\|^2 \tag{2}$$

其中，$\hat{a}_G=(X_G^{\mathrm{T}}X_G)^{-1}X_G^{\mathrm{T}}Y_G$ 表示在数据集 G 上估计得到的系数，$G=A,B,W$，$W=A\cup B$，$H=A,B,H\cap G=\varnothing$。记 $X_{it_i}=\{X_{is}\mid s\leqslant t_i\}$，$A_t=\{X_{it_i}\mid i=1,2,\cdots,m\}$。如果

$$\Delta^2(Y_t|A_t)<\Delta^2(Y_t|A_t-X_{it_i}) \tag{3}$$

则称 X_i 是 Y 的 GMDH 原因的构成要素；如果

$$\Delta^2(Y_t|A_t,X_{it_i})<\Delta^2(Y_t|A_t) \tag{4}$$

则称 X_i 是 Y 的 GMDH 瞬时原因的构成要素。

GMDH 因果关系可检验复杂系统中多维变量间因果关系及其关联方式和强弱程度，识别出复杂系统中同时存在的多个因果关系及其相互关联关系，进而从整体上了解系统变量的层次、结构和功能。

2.2 GMDH 算法

GMDH 是自组织数据挖掘的核心技术，以 GMDH 为基础发展出了多种自组织数据挖掘算法。自组织将数据分为训练集和测试集，在训练集上使用内准则进行参数估计得到中间待选模型，而在测试集上使用外准则进行中间候选模型的选择，这个过程不断重复直到外准则值不能再改善才停止。这样的停止法则可以保证在一定噪声水平下得到数据拟合精度和预测能力实现最优平衡的最优复杂度模型。GMDH 产生最优模型的过程如图 1 所示。

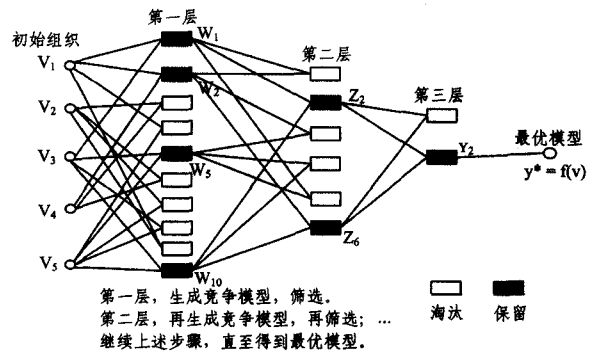

图 1 GMDH 产生最优模型的过程

GMDH 算法的实现步骤为[17]：

(1)将数据样本集(N 个数据样本)分为训练集 A(training set)和检测集 B(testing set)($N_W=N_A+N_B$，$W=A\cup B$)。若建立预测模型，则将数据样本集分为学习集 A、检测集 B 和预测集 C(checking set)，$N_W=N_A+N_B+N_C$，$W=A\cup B\cup C$。

(2)建立因变量(输出)和自变量(输入)之间的一般关系作为"转换函数"，一般常用 K-G 多项式。对于三输入单输出系统，例如可取二次 K-G 多项式

$$f(x_1,x_2,x_3)=a_0+a_1x_1+a_2x_2+a_3x_3+a_4x_1^2+a_5x_2^2+a_6x_3^2+a_7x_1x_2+a_8x_1x_3+a_9x_3x_2$$

为转换函数，并以它的子项作为建模网络结构中的 m 个初始模型：

$$v_1=a_0, v_2=a_1x_1, v_3=a_2x_2, \cdots, v_{10}=a_9x_2x_3$$

此处 $m=10$。

(3)从具有外补充性质的选择准则中选出一个或若干个作为目标函数，即选择外准则。

(4)产生第一层中间模型，如图 1 所示。传递函数 $y_k=f_k(v_i,v_j)(k=1,2,\cdots,m)$ 为第一层中间模型，它们由自组织过程自适应产生，且因所含变量个数、函数结构而彼此不同。同时，在训练集 A 上估计 y_k 的参数。

(5)第一层中间模型筛选。根据外准则，在检测集 B 上对第一层中间模型进行筛选，选出的中间模型 $w_k(k=2,3,6,7,9)$ 将作为网络第二层的输入变量。

(6)形成最优复杂度模型网络结构。重复步骤(4)、(5)，可依次产生第二层，第三层，…，等中间模型，最终形成可用于分析的显式最优分析模型(见图 1)，这里以第三层后的状态为例。在模型 y^* 中变量 v_i 的数目小于或等于 4，而网络初始变量的个数为 5，初始变量 v_2 在筛选中被自动淘汰。

从上面建模过程可以看出，利用 GMDH 方法具有以下特点：1)不需要特定的先验知识和前提假设；2)因为 GMDH 算法的递归性质，用 GMDH 建模需要的数据样本少；3)能有效抵抗噪声干扰；4)能有效避免模型过拟合，模型的预测能力好；5)得到的最终模型容易解释。

3 基于 GMDH 因果关系的软件失效预测模型

与已有的对 Granger 因果关系检验方法[12,13]不同，基于 GMDH 的因果关系检验从复杂系统建模角度研究变量间的关系，其优点在于：对于单输出(结果)多维系统，不需 Granger因果关系检验的诸多限制条件；可以检验变量之间在非线性意义上的因果关系；用外准则值作为推断因果关系的依据可以避免用 F 统计量时容易出现的错误；能从整体上分析多个变量与“结果”变量的关系，同时还能反映变量影响的强弱程度，因而有利于揭示导致软件失效的原因。

基于 GMDH 因果关系检验的软件失效预测分析分为 3 个阶段：初步定性分析、GMDH 因果关系检验和软件失效预测，其流程如图 2 所示。

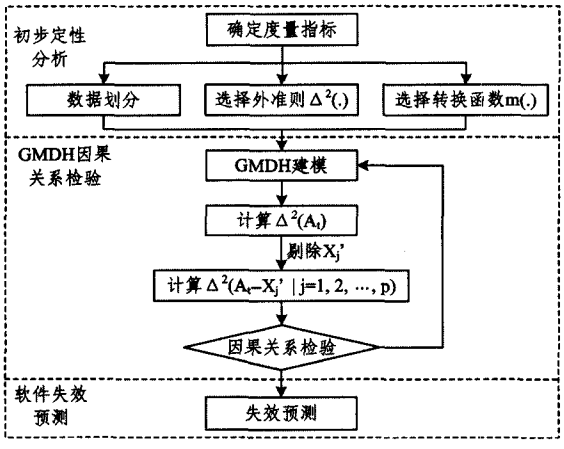

图 2 基于 GMDH 因果关系检验的软件失效预测模型

3.1 初步定性分析

初步定性分析主要包括选择确定进行软件失效预测的各个软件度量指标，对于数据集样本进行数据划分，为 GMDH 模型确定相应的外准则和转换函数。

(1)选择确定软件度量指标

影响软件失效的因素很多，可以出现在整个软件生命周期的各个阶段，主要包括外部因素与软件本身内部因素。根据专家知识、经验以及相关研究确定可能影响软件失效(结果变量)Y 的软件度量指标(因素)集合$\{X_1,X_2,\cdots,X_k\}$。

(2)数据划分

将数据样本集(N 个数据样本)分为训练集 A(training set)和检测集 B(testing set)($N_W=N_A+N_B, W=A\cup B$)。若建立预测模型，则将数据样本集分为学习集 A、检测集 B 和预测集 C (checking set)，$N_W=N_A+N_B+N_C, W=A\cup B\cup C$。

(3)外准则选择

GMDH 模型的变量筛选准则和停止准则一般采用外准则，一般可分为精度准则和相容性准则两大类。其中精度准则是以所建立模型的拟合精度作为算法停止的条件，如正则化(Regularity)准则、稳定(Stability)准则、预测(Prediction)准则和相关性准则；相容性准则是考察在不同样本数据集上对同一系统所建模型的一致性，如最小偏差准则(Minimum bias criterion，又称为无偏准则)。为了强调模型的预测能力，本文采用预测准则作为 GMDH 模型的外准则，具体定义如下：

$$\begin{cases}\Delta^2(A)=\sum_{t\in C}(y_t-y_t^m(A))^2\\ \Delta^2(B)=\sum_{t\in C}(y_t-y_t^m(B))^2\\ \Delta^2(W)=\sum_{t\in C}(y_t-y_t^m(W))^2\end{cases}\tag{5}$$

其中，$y_t, y_t^m(\cdot)$ 分别表示实际观测值和模型在不同数据集(训练集 A、测试集 B 和样本数据集 W)上的预测值。

(4)选择转换函数

在最优 GMHD 模型中，建立因变量(输出)和自变量(输入)之间的一般关系作为转换函数，常用 K-G 多项式[20]。

3.2 GMDH 因果关系检验

(1)GMDH 建模

利用改进的 GMDH 算法，建立以 Y 为输出、$\{X_1,X_2,\cdots,X_k\}$ 为输入的 GMDH 模型。记经过 GMDH 筛选保留在最优模型中的原始变量集为 $A=\{X_1',X_2',\cdots,X_p'\}\subseteq\{X_1,X_2,\cdots,X_k\}, p\leqslant k$。记 $X'_{l_l}=\{X'_{ls}\mid s\leqslant t_l\}(l=1,2,\cdots,p)$ 表示保留在最优模型中 X_l' 的所有滞后变量，t_l 为 X_l' 的最大滞后阶数。令 $A_t=\{X'_{l_l}\mid l=1,2,\cdots,p\}$。

(2)GMDH 因果关系检验

由式(5)计算最优 GMDH 模型的预测准则值 $\Delta^2(A_t)$。依次剔除变量 $X_i\in\{X_1',X_2',\cdots,X_p'\}$ 及其滞后变量，并依次重新建立 GMDH 模型。分别计算这 p 个模型的预测准则值。记只剔除 X_i 本身后新模型的预测准则值为 $\Delta^2(A_t-X_j'\mid j=1,2,\cdots,p)$，剔除 X_i 及其滞后变量后得到的预测准则值为 $\Delta^2(A_t-X'_{jt_j}\mid j=1,2,\cdots,p)$。

利用 Diebold-Mariano 检验方法分别检验预测准则值 $\Delta^2(A_t)$ 与 $\Delta^2(A_t-X_j'\mid j=1,2,\cdots,p)$、$\Delta^2(A_t)$ 与 $\Delta^2(A_t-$

$X'_{jt_j}|j=1,2,\cdots,p)$差异的显著性。Diebold-Mariano 检验具体为：令

$$d_t=g(\Delta^2(A_t))-g(\Delta^2(A_t-X'_{jt_j}|j=1,2,\cdots,p)) \quad (6)$$

其中，$g(\cdot)$是损失函数，可以取 $g(u)=u^2$，则Diebold-Mariano 检验统计量为：

$$DM=[\hat{V}(\bar{d})]^{1/2}\bar{d} \quad (7)$$

其中，$\bar{d}$是 d_t 的样本均值，$\hat{V}(\bar{d})$是 Newy-West 异方差，是$\bar{d}$的样本方差的自回归一致性估计。根据 Diebold-Mariano 假定，检验统计 DM 服从均值为 0 的正态分布，通过计算其在正态分布假定下的 p 值，来判断各类模型在预测准确性上是否一致。

3.3 软件失效预测

根据 GMDH 因果关系检验获得与软件失效有因果关系的软件度量指标，重新建立 GMDH 软件失效预测模型，对软件进行失效预测。

4 实例分析

本文选取两个真实的软件失效数据集 Eclipse JDT 和 Eclipse PDE[21] 对 GMDH 方法进行短期预测验证，其中，数据集 Eclipse JDT 包含 182 个失效观察值，数据集 Eclipse PDE 包含 179 个失效观察值。观察值一共包含 8 个测量指标，分别是 fanIn，fanOut，lackOfCohesionInMethod，numberOfAttributes，numberOfLinesOfCode，numberOfMethods，weigh-tedMethodCount以及 defects，本文利用 defects 作为因变量来评估软件失效预测效果。观察值为一时间序列，记录了当前观察与上次观察的各测量指标缺陷数的增减情况。这两个数据集均保留最后 10 个数据观察值作为验证集，其余数据作为训练集。

为了更好地验证 GMDH 模型的有效性，本文将其与利用 Granger 因果检验的软件缺陷预测模型作性能对比。其中，预测模型选为反向传播神经网络预测模型(BPN)与基因表达式编程预测模型(GEP)，分别记为 Granger-BPN 以及 Granger-GEP。

在实验中采用以下 4 种性能评价指标进行比较。

(1)均值误差平方和(mean square error，MSE)

$$MSE=\frac{1}{n}\sum_{i=1}^{n}(y_i-y_i')^2 \quad (8)$$

(2)回归曲线方程的相关指数(R-Square 或 R 值)

$$R\text{-}Square=1-\frac{\sum_{i=1}^{n}(y_i-y_i')^2}{\sum_{i=1}^{n}(y_i-y_{ave})^2} \quad (9)$$

(3)均值误差(average error，AE)

$$AE=\frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i-y_i'}{y_i}\right|\times 100 \quad (10)$$

(4)均方百分比误差(MSPE)

$$MSPE=\frac{1}{n}\sqrt{\sum_{i=1}^{n}\left(\frac{y_i-y_i'}{y_i}\right)^2} \quad (11)$$

式(8)－式(11)中，y_i 表示数据的实际值，y_i'表示数据的预测值，y_{ave}表示观测数据 y_i 的均值。显然，AE 值、MSE 值和 $MSPE$ 值越小，$R\text{-}Square$ 值越接近 1，这表明预测值与实际值越接近，模型拟合或预测性能越好。

实例 1　本文选取的实验数据是 Eclipse JDT 数据集，各测量指标如图 3 所示。

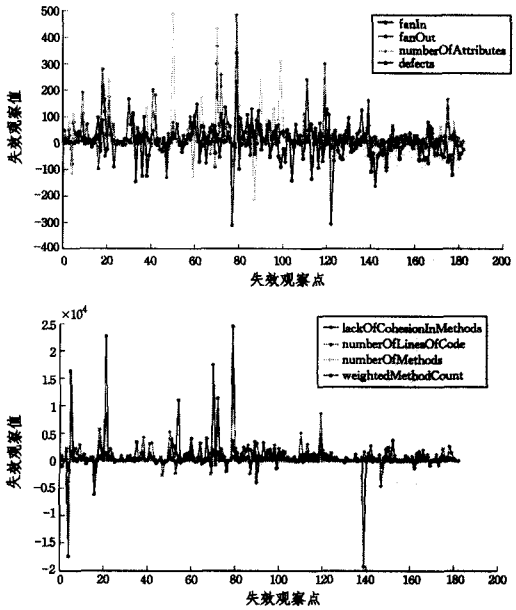

图 3　Eclipse JDT 数据集

将测量指标 defects 作为因变量，其余测量指标作为自变量，无需进行 Granger 因果检验，直接导入到 GMDH 因果关系模型中。图 4 为 GMDH 模型、Granger-BPN 模型以及 Granger-GEP 模型所得预测结果的每个观察值相对误差值结果，每个点的误差值越接近于 0，代表模型在这个点上的预测精度越高。

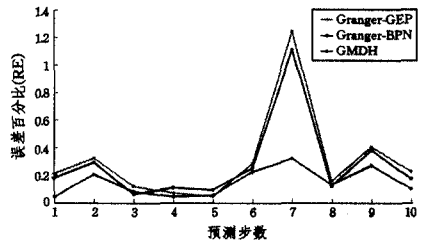

图 4　JDT 数据集在各模型中预测误差百分比对比

表 1 给出了分别利用 GMDH、Granger-BPN 以及 Granger-GEP 方法对 JDT 数据进行预测的结果及相应评价指标值。

表 1　JDT 数据集在各模型的预测性能指标

预测性能指标	模型		
	Granger-BPN	Granger-GEP	GMDH
R-Square	0.9587	0.9216	**0.9816**
AE	27.8309	50.9395	**14.7607**
MSPE	0.1279	0.3321	**0.0558**
MSE	74.3364	91.2477	**33.2158**

由图 4 及表 1 可知，利用 GMDH 算法得到的结果中，R^2 值较 Granger-BPN 以及 Granger-GEP 方法的大，接近于 1；AE 值、$MSPE$ 以及 MSE 值均要比 Granger-BPN 以及 Granger-GEP 方法的小。可以看出，GMDH 较其它方法更优。

实例 2　本文选取的实验数据是 Eclipse PDE 数据集，各

测量指标如图 5 所示。

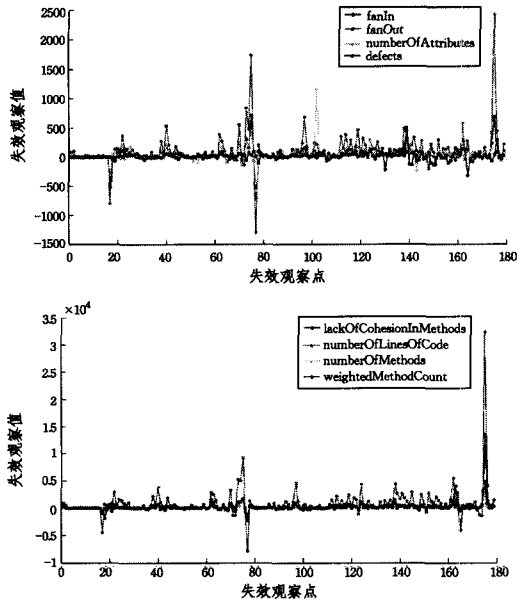

图 5　Eclipse PDE 数据集

同样将测量指标 defects 作为因变量，其余测量指标作为自变量，代入到 GMDH 因果关系模型中进行计算。图 6 示出 GMDH 模型、Granger-BPN 模型以及 Granger-GEP 模型所得预测结果的每个观察值相对误差值结果。

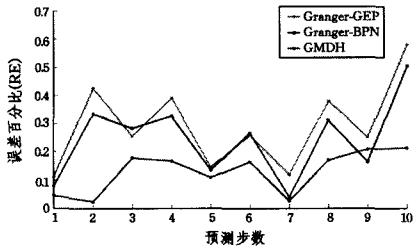

图 6　PDE 数据集在各模型中预测误差百分比对比

表 2 给出了分别利用 GMDH、Granger-BPN 以及 Granger-GEP方法对 PDE 数据进行预测的结果及相应评价指标值。

表 2　PDE 数据集在各模型的预测性能指标

预测性能指标	模型		
	Granger-BPN	Granger-GEP	GMDH
R-Square	0.9946	0.9814	**0.9972**
AE	24.3967	29.1578	**12.9921**
MSPE	0.0879	0.1029	**0.0468**
MSE	48.9004	66.904	**24.8275**

由图 6 及表 2 可知，利用 GMDH 算法得到的结果中，R^2 值较 Granger-BPN 以及 Granger-GEP 方法的大，接近于 1；*AE* 值、*MSPE* 以及 *MSE* 值均要比 Granger-BPN 以及 Granger-GER 方法的小。可以看出，GMDH 较其它方法更优。

结束语　本文基于 GMDH 网络与 Granger 检验提出了一种软件失效预测模型，借鉴 Granger 检验思想，利用 GMDH 网络选择与软件失效具有因果的软件度量指标，建立软件失效预测模型。基于 GMDH 的因果关系检验从复杂系统建模角度研究变量间的关系，其优点在于对于单输出（结果）多维系统，不需 Granger 因果关系检验的诸多限制条件，可以检验变量之间在非线性意义上的因果关系。实验结果表明：该方法在预测时效果较好，与一些常用预测算法相比，效果有一定改善。

本模型由于使用 GMDH 网络对数据进行训练，因此数据的训练集应尽可能大，以使 GMDH 网络得到充分训练，这样才能达到更好的预测效果。另外，本文算法是进行单步预测，每预测一步后，再把其加入训练集来预测下一个点，随着预测点数的增加，在后面点数的预测中误差会越来越大，所以此模型更适合于进行短期预测。

参 考 文 献

[1] Wang Qing, Wu Shu-jian, Li Ming-shu. Software defect prediction technology [J]. Journal of Software, 2008, 19(7): 1565-1580(in Chinese)
王青，伍书剑，李明树. 软件缺陷预测技术[J]. 软件学报，2008，19(7)：1565-1580

[2] Nagappan N, Ball T, Zeller A. Mining metrics to predict component failures[C]// 28th International Conference on Software Engineering (ICSE). 2006: 452-461

[3] Couto C, Araujo J E, Silva C, et al. Static correspondence and correlation between field defects and warnings reported by a bug finding tool[J]. Software Quality Journal, 2013, 21(2): 241-257

[4] Catal C. Software fault prediction: A literature review and current trends[J]. Expert Systems with Applications, 2011, 38(4): 4626-4636

[5] Singh Y, Kaur A, Malhotra R. Empirical validation of object-oriented metrics for predicting fault proneness models[J]. Software Quality Journal, 2010, 18(1): 3-35

[6] Couto C, Montandon J E, Silva C, et al. Static correspondence and correlation between field defects and warnings reported by a bug finding tool[J]. Software Quality Journal, 2013, 21(2): 241-257

[7] Zimmermann T, Nagappan N. Predicting defects using network analysis on dependency graphs[C]// Proceedings of the 30th International Conference on Software Engineering. ACM, 2008: 531-540

[8] Nagappn N, Ball T. Using software dependencies and churn metrics to predict field failures: an empirical case study[C]// First International Symposisum on Empirical Software Engineering and Measurement. 2007: 364-373

[9] Lee H J, Naish L, Ramamohanarao K. Study of the relationship of bug consistency with respect to performance of spectra metrics[C]// 2nd IEEE International Conference on Computer Science and Information Technology, 2009 (ICCSIT 2009). 2009: 501-508

[10] Okutan A, Yildiz O T. Software defect prediction using Bayesian networks[J]. Empir Software Eng., 2014, 19(3): 154-181

[11] Lehtinen T O A, Mäntylä M V, Vanhanen J, et al. Perceived causes of software project failures-An analysis of their relationships[J]. Information and Software Technology, 2014, 56(6): 623-643

[12] Couto C, Silva C, Valente M T, et al. Uncovering causal relationships between software metrics and bugs[C]// 17th European Conference on Software Maintenance and Reengineering

(CSMR). 2012:223-232

[13] Couto C, Pires P, Valente M T, et al. Predicting software defects with causality tests[J]. Journal of Systems and Software, 2014, 93(6):24-41

[14] D'Ambros M, Lanza M, Robbes R. An extensive comparison of bug prediction approaches[C]// 7th IEEE Working Conference on Mining Software Repositories (MSR), 2010. 2010:31-41

[15] Granger C. Investigating causal relations by econometric models and cross-spectral methods[J]. Econometrica, 1969, 37(3):424-438

[16] Chow G C. Econometrics[M]. New York: McGraw Hill, 1983

[17] Zhang M Z, He C Z, Gu X, et al. D-GMDH: A novel inductive modelling approach in the forecasting of the industrial economy [J]. Economic Modelling, 2013, 30(2):514-520

[18] Tamura H, Kondo T. Heuristics Free Group Method of Data Handling Algorithm of Generating Optimal Partial Polynomials with Application to Air Pollution Prediction[J]. International Journal of Systems Science, 1980, 11(9):1095-1011

[19] Liu W, Tian S B. An Improved GMSM Method and its Application[J]. ACTA Automatic Sinica, 1993, 19(4):468-471

[20] Ramakanta M, Ravi V. Software Reliability Prediction Using Group Method of Data Handling[C]// 12th International Conference of Rough Sets, Fuzzy Sets, Data Mining and Granular Computing. 2009:13-19

[21] Couto C, Piresa P. Predicting software defects with causality tests[J]. Empirical Software Engineering, 2014, 19:154-181

(上接第 152 页)

(1)指纹采集阶段的安全性分析

对指纹数据的攻击,是指用户指纹数据被采集并传输到安全操作系统内部之前,被潜在的恶意程序截获。为了避免此种攻击,本文采用通用的加密算法对指纹数据进行加密,以确保本阶段指纹数据的安全,而用于指纹数据加密的密钥则保存在受 TrustZone 保护的硬件隔离区域,该密钥的获取则是通过指纹安全通道进行传输。

(2)指纹信息传输过程的安全性分析

针对指纹信息传输过程的攻击,主要发生在把指纹数据从 Android 环境传输到 T-OS 系统阶段。由于 TrustZone 驱动模块会在内核层申请内存空间,拷贝用户层内存空间的指纹数据,并且整个阶段指纹以密文形式存在,从而保证了本过程的安全性。

(3)指纹模板库的安全安全性分析

指纹模板库作为指纹匹配操作的模板资源,是整个指纹识别过程的基准和依据。指纹模板库由于存储在安全隔离区域中,因此可以阻隔来自非安全环境的安全威胁。此外,为防止 T-OS 内的其他安全服务获取该指纹模板库,通过使用非对称加密算法并结合存储在隔离区域内的密钥,完成对指纹模板库的加密保护。

(4)指纹识别处理程序的安全性分析

针对指纹识别过程的攻击,主要发生在指纹预处理、特征值提取和匹配等阶段,保证该流程的安全可靠是指纹识别服务的关键。本文通过借助于 ARM TrustZone 技术所搭建的指纹识别安全框架,确保了整个识别过程运行在安全环境内部,从而有效地避免了恶意程序对该过程的攻击。

结束语 目前,将指纹识别技术大量地应用于便携设备上,不管是手机厂商,还是互联网企业、移动运营商、银联等,都将会极大地推进移动支付业的发展,而作为安全支付关键技术的指纹识别技术的重要性是毋庸置疑的。保障用户指纹的机密性和指纹处理的安全性是本文研究的目标,本文借助于 TrustZone 的硬件隔离保护机制以及安全加密和安全存储等安全技术实现了对敏感资源操作的安全性保护。其中包括以下 3 个方面的安全:用户指纹数据的安全加密避免了 Android 恶意程序截获用户指纹信息的风险;Android 与 T-OS 间的安全通道确保了指纹密文数据的安全传输;T-OS 内部的指纹预处理、特征值提取与安全存储、指纹匹配操作确保了指纹识别服务过程的安全。通过保障每个环节的安全,保证了整个指纹识别过程的安全。而随着指纹识别技术发展的不断深入,智能终端支付的便捷性和安全性有着一定的研究意义和应用价值。

参 考 文 献

[1] ARM. Building a Secure System using TrustZone Technology [M]. 2009

[2] Wang Dong, Fan Jian-ying. Application Study of Fingerprint Identification Based on Linux Embedded System[D]. Harbin: Harbin Engineering University, 2009(in Chinese)
王东. 嵌入式 Linux 系统在指纹识别中的应用研究[D]. 哈尔滨:哈尔滨理工大学,2009

[3] Luo Fan, Pei Yi-jian. Research and design of embedded fingerprint recognition system based on ARM+Linux[D]. Kunming: Yunnan University, 2014(in Chinese)
罗凡,裴以建. 基于 ARM+Linux 的嵌入式指纹识别系统研究与设计[D]. 昆明:云南大学,2014

[4] Shen Yong, Zhu Wen-jing. Design and Implementation of Security-enhanced Scheme for Embedded Database[J]. Modern Electronics Technique, 2010(14):21-24(in Chinese)
沈勇,朱文静. 一种嵌入式数据库安全增强方案的设计与实现[J]. 现代电子技术,2010(14):21-24

[5] Luo Jing, Yang Xia, et al. Design and Implementation of Security OS based on the TrustZone[C]// ICEMI. IEEE Press, 2013: 1027-1032

[6] Global Platform Device Technology. TEE Internal API Specification[EB/OL]. [2011-01]. http://www.globalplatform.org/specificationsdevice.asp

[7] Yang Xia, Luo Jing, et al. Trust-E: A Trusted Embedded Operating System Based on the ARM Trustzone[C]// UIC-ATC-ScalCom. 2014

[8] Wang Jin-xiang. Gabor Filter based Fingerprint Image Enhancement[C]// International Society for Optics and Photonics. 2013

[9] Wang W, Li J, Huang F, et al. Design and implementation of Log-Gabor filter in fingerprint image enhancement[J]. Pattern Recognition Letters, 2008, 29(3):301-308