

基于连续型深度置信神经网络的软件可靠性预测

亓 慧¹ 史 颖^{1,2} 李灯熬³ 穆晓芳¹ 侯明星¹

1 太原师范学院计算机系 山西 晋中 030619

2 山西大学计算机与信息技术学院 太原 030006

3 太原理工大学大数据学院 山西 晋中 030600

摘 要 为了提高软件可靠性智能预测的精度,采用连续型深度置信神经网络算法用于软件可靠性预测。首先提取影响软件可靠性的核心要素样本,并获取样本要素的关键特征;然后建立连续型深度置信神经网络(Deep Belief Network,DBN)的软件可靠性预测模型,输入待预测样本,通过多个受限波尔兹曼机(Restricted Boltzmann Machine,RBM)层的预处理训练,以及多次反向微调迭代获取 DBN 权重等参数,直到达到最大 RBM 层数和最大反向微调迭代次数;最后获得稳定的软件可靠性预测模型。实验结果证明,通过合理设置 DBN 隐藏层节点数和学习速率,可以获得良好的软件可靠性预测准确率和标准差。与常用的软件可靠性预测算法相比,所提算法的预测准确度高且标准差小,在软件可靠性预测方面的适用度较高。

关键词:深度置信神经网络;软件可靠性;软件失效;学习速率

中图法分类号 TP311.5

Software Reliability Prediction Based on Continuous Deep Confidence Neural Network

QI Hui¹, SHI Ying^{1,2}, LI Deng-ao³, MU Xiao-fang¹ and HOU Ming-xing¹

1 Department of Computer, Taiyuan Normal University, Jinzhong, Shanxi 030619, China

2 School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China

3 College of Data Science, Taiyuan University of Technology, Jinzhong, Shanxi 030600, China

Abstract In order to improve the accuracy of intelligent prediction of software reliability, continuous depth confidence neural network algorithm is used for software reliability prediction. Firstly, the core elements samples that affect software reliability are extracted, and the key features of the sample elements are obtained. Then, a software reliability prediction model based on continuous deep belief neural network (DBN) is established. The samples to be predicted are input, and the parameters such as DBN weight are obtained through pre-processing training of multiple Restricted Boltzmann Machine (RBM) layers and multiple reverse fine-tuning iterations until the maximum number of RBM layers and the maximum number of reverse fine-tuning iterations are reached. Finally, a stable software reliability prediction model is obtained. Experiments show that good software reliability prediction accuracy and standard deviation can be obtained by reasonably setting the number of nodes in the hidden layer of DBN and the learning rate. Compared with commonly used software reliability prediction algorithms, this algorithm has high prediction accuracy, small standard deviation and high applicability in software reliability prediction.

Keywords Deep confidence neural network, Software reliability, Software failure, Learning rate

1 引言

计算机行业快速发展,软件作为计算机的核心支撑,占据着不可替代的地位,随着计算机在多种领域的多样化应用,软

件服务也呈现出了不同特色。软件服务的跨平台、大规模及深度应用需求定制等特点,导致了其可靠性运行受到巨大挑战,特别是重点领域的软件可靠性要求成为了重点关注的问题^[1-2],例如航空航天、军事指挥、金融服务及工业控制等领域

到稿日期:2021-02-05 返修日期:2021-04-03

基金项目:国家重大科研仪器研制项目(6202780085);国家自然科学基金(62076177);山西省关键核心技术和共性技术研发专项(2020xxx007);山西省科技厅重点研发项目(201803D31055)

This work was supported by the National Major Scientific Research Instrument Development Project(6202780085), National Natural Science Foundation of China(62076177), Shanxi Province Key Core Technology and Common Technology R&D Project(2020xxx007) and Key Research and Development Project of Shanxi Province(201803D31055).

通信作者:亓慧(qihui@tynu.edu.cn)

的应用都对软件的可靠性提出了严苛的要求。

软件服务需要适用不同的硬件平台,同时接收不同的网络服务需求,而且还需要防范网络安全问题及自身运行的稳定性问题,这些都是软件可靠性需要直接面对的问题,它们决定了软件可靠性分析的主要要素及关键指标。在对软件可靠性进行分析时,为了尽可能降低软件在整个工程应用中造成的不稳定问题及规避因软件而造成的不必要损失,对软件可靠性进行预测非常关键。软件可靠性预计属于工程领域 RAMS(Reliability, Availability, Maintainability, Safety)体系,其方法、数据来源是有特殊要求的,而可靠性预测则无特殊要求,属于经验估算。通过合理设计算法来实现软件可靠性预测,通过分析预测结果对软件进行优化完善,可以保证软件在其运行周期内正常运行。

目前,关于软件可靠性预测的研究较多。例如,Jain 等^[3]提出了基于遗传算法的软件可靠性评估方法,能够在早期预测软件程序中的故障和错误。Jabeen 等^[4]提出了一种基于高精度误差迭代分析方法的改进软件可靠性预测模型,主要将残差修正与神经网络符号估计器相结合,相比基于遗传算法的预测模型,该方法的效率有一定提高,但是未对精度结果做具体分析。Gong 等^[5]采用最近邻算法对软件可靠性进行预测,预测精度较高,但牺牲了预测效率。Cotroneo 等^[6]采用自适应测试方案来提高软件系统的可靠性,但是需要先验的测试和故障排除预处理,才能获得可靠的连续评估结果。

随着计算性能的提升,基于深度学习的软件可靠性估计方法组建成为了近期研究的热点。例如 Tamura 等^[7]提出了基于深度学习的软件可靠性模型。连续型深度置信网络作为经典 DBN 的一种衍生版本^[8],其性能较为突出,因此本文尝试将连续型 DBN 神经网络用于软件可靠性预测,通过多个 RBM 网络的训练来获得较高的软件可靠性预测精度。

2 软件可靠性

软件可靠性衡量了软件在运行周期内正常运行的概率,软件可靠性与该概率值成正比。一般而言,软件可靠性函数的自变量为时间 t ,软件在其运行周期内有两种工作状态,即正常或失效,根据两者的概率值可以衡量软件运行的可靠性程度。

设软件失效时刻为 t ,在 $[0, t]$ 内程序运行失效的概率为 $Pf(t)$,运行成功的概率为 $Ps(t)$,两者关系为^[7]:

$$Ps(t) + Pf(t) = 1 \quad (1)$$

令 $R(t) = Ps(t)$,则:

$$R(t) = 1 - Pf(t) = 1 - \int_0^t f(x) dx \quad (2)$$

其中, $f(x)$ 表示软件运行失效的概率密度函数。

在研究软件可靠性之前,需要对影响软件可靠性的主要因素进行分析,本文结合文献[8-9]总结了软件可靠性的主要因素,如图 1 所示。

图 1 只是给出了影响软件可靠性的主要因素,还有很多因素未列出,若要获得较高的软件可靠性预测精度,则需要对

这些因素进行量化和维度分析,因此软件可靠性预测是一个复杂问题,需要借助深度学习算法来解决其预测分析问题。在实际的软件可靠性预测建模中,需要充分挖掘影响软件可靠性的核心因素,并对核心因素进行属性提取,最后进行算法设计来完成软件可靠性预测。

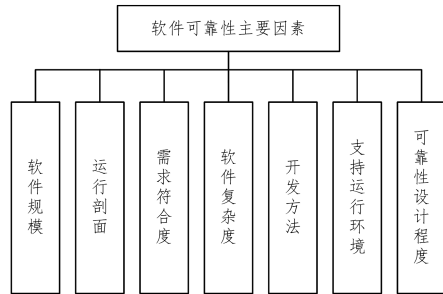


图 1 软件可靠性的主要影响因素

Fig. 1 The main influencing factors of software reliability

3 深度置信神经网络的软件可靠性预测

3.1 连续型深度置信网络模型的构建

设 DBN^[10] 的可视层 $\mathbf{v} = (v_1, v_2, \dots, v_m)$, 隐藏层 $\mathbf{h} = (h_1, h_2, \dots, h_n)$ 。DBN 中 RBM 的确定主要由 $\theta = \{\omega, c, b\}$ 决定, ω 表示 $\mathbf{v}$ 和 $\mathbf{h}$ 层间的权值, c 和 b 表示 $\mathbf{v}$ 和 $\mathbf{h}$ 层的偏移。DBN 的输入和输出均为离散量, 当 $\mathbf{v}$ 和 $\mathbf{h}$ 层由离散值变为连续值时, 则形成了连续型深度置信网络(Continuous Deep Belief Network, CDBN), 其计算方法与 DBN 基本一致。

$\mathbf{v}$ 层相对于 $\mathbf{h}$ 层单节点输出被保留的概率为:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h})} \quad (3)$$

其中, $Z = \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}$ 。 $E(\mathbf{v}, \mathbf{h})$ 为服从伯努利分布的能量函数^[11]:

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{j=1}^m b_j v_j - \sum_{i=1}^n c_i h_i - \sum_{i=1}^n \sum_{j=1}^m \omega_{ij} v_j h_i \quad (4)$$

结合式(3), 可求解 $\mathbf{h}$ 层的 n 个节点作用于 $\mathbf{v}$ 层的概率^[12]:

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} \quad (5)$$

$\mathbf{v}$ 层中 m 个节点作用于 $\mathbf{h}$ 层的概率为:

$$P(\mathbf{h}) = \sum_{\mathbf{v}} P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \sum_{\mathbf{v}} e^{-E(\mathbf{v}, \mathbf{h})} \quad (6)$$

对于第 m 个 $\mathbf{v}$ 层节点, 第 i 个 $\mathbf{h}$ 层节点被激活的概率为:

$$P(h_i = 1 | \mathbf{v}) = \sigma(c_i + \sum_{j=1}^m \omega_{ij} v_j) \quad (7)$$

在 $\mathbf{h}$ 层的 n 个节点的作用下, 第 j 个 $\mathbf{v}$ 层节点被激活的概率为:

$$P(v_j = 1 | \mathbf{h}) = \sigma(b_j + \sum_{i=1}^n \omega_{ji} v_i) \quad (8)$$

σ 函数为 $\sigma(x) = \text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$ 。

当 $\mathbf{v}$ 层 $\mathbf{v} = \{v_0, v_1, \dots, v_N\}$ 中的节点 $v_0, v_1, \dots, v_N$ 独立同分布时, 则有 $P(\mathbf{v}) = \prod_{t=0}^N P(v_t)$, 其似然估计 $L(\theta) = \prod_{t=0}^N P(v_t | \theta)$ 。对 $L(\theta)$ 取自然对数, 再求解 $\ln L(\theta)$ 的最大值^[13]。

$$\hat{\theta} = \arg \max_{\theta} L(\theta) = \arg \max_{\theta} \sum_{t=0}^N \ln P(v_t | \theta)$$

(9)

$$\theta^* = \theta + \eta \frac{\partial \ln P(v)}{\partial \theta}$$

(10)

其中, $\eta(\eta > 0)$ 为学习速率。

$\mathbf{v}$ 层中一个节点 $\mathbf{v}_0 = \{v_{01}, v_{02}, \dots, v_{0m}\}$, 求解其在 $\mathbf{h}$ 层作用下输出被保留的概率。

$$\ln P(\mathbf{v}_0) = \ln \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}_0, \mathbf{h})} = \ln \sum_{\mathbf{h}} e^{-E(\mathbf{v}_0, \mathbf{h})} - \ln \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}$$

(11)

根据式(11)对 $\theta = \{w, c, b\}$ 求偏导, 得:

$$\frac{\partial \ln P(\mathbf{v}_0)}{\partial \theta} = - \sum_{\mathbf{h}} P(\mathbf{h} | \mathbf{v}_0) \frac{\partial E(\mathbf{v}_0, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{v}, \mathbf{h}} P(\mathbf{v}, \mathbf{h}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta}$$

(12)

根据条件概率计算公式有:

$$P(\mathbf{v}, \mathbf{h}) = P(\mathbf{h} | \mathbf{v}) P(\mathbf{v})$$

(13)

式(12)可变为^[14]:

$$\begin{aligned} \frac{\partial \ln P(\mathbf{v}_0)}{\partial \theta} &= - \sum_{\mathbf{h}} P(\mathbf{h} | \mathbf{v}_0) \frac{\partial E(\mathbf{v}_0, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{v}} \sum_{\mathbf{h}} P(\mathbf{v}) P(\mathbf{h} | \mathbf{v}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} \\ &= - \sum_{\mathbf{h}} P(\mathbf{h} | \mathbf{v}_0) \frac{\partial E(\mathbf{v}_0, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{v}} P(\mathbf{v}) \sum_{\mathbf{h}} P(\mathbf{h} | \mathbf{v}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} \end{aligned}$$

(14)

接着, 对 $\theta = \{w, c, b\}$ 中的 w, c, b 分别求偏导:

$$\frac{\partial \ln P(\mathbf{v}_0)}{\partial w_{ij}} = P(h_i = 1 | \mathbf{v}_0) v_{0j} - \sum_{\mathbf{v}} P(\mathbf{v}) P(h_i = 1 | \mathbf{v})$$

(15)

$$\frac{\partial \ln P(\mathbf{v}_0)}{\partial b_j} = v_{0j} - \sum_{\mathbf{v}} P(\mathbf{v})$$

(16)

$$\frac{\partial \ln P(\mathbf{v}_0)}{\partial c_i} = P(h_i = 1 | \mathbf{v}_0) - \sum_{\mathbf{v}} P(\mathbf{v}) P(h_i = 1 | \mathbf{v})$$

(17)

根据上述内容, 可以获得 w, c, b 的更新方法:

$$w_{ij}^* = w_{ij} + \eta \frac{\partial \ln P(\mathbf{v}_0)}{\partial w_{ij}}$$

(18)

$$b_j^* = b_j + \eta \frac{\partial \ln P(\mathbf{v}_0)}{\partial b_j}$$

(19)

$$c_i^* = c_i + \eta \frac{\partial \ln P(\mathbf{v}_0)}{\partial c_i}$$

(20)

采用上述方法不断完成层与层之间的计算, 直到经过了 RBM 的所有层后, 完成 RBM 预训练并得到 $\theta^* = (w_{ij}^*, b_j^*, c_i^*)$ 。

最后, 采用反向传播继续调整 $\theta = \{w, c, b\}$, 当达到最大迭代次数时, 输出 $\theta^* = (w_{ij}^*, b_j^*, c_i^*)$, 从而获得稳定的 DBN 结构。

3.2 基于 CDBN 的软件可靠性预测流程

根据图 1 所示的因素表建立 CDBN 软件可靠性预测样本, 并通过主成分分析 (Principal Component Analysis, PCA) 等方法分析影响因素的核心属性, 然后建立 CDBN 预测模型, 最后通过训练获得稳定的软件可靠性预测模型, 其主要流程如图 2 所示。

在设置 CDBN 的软件可靠性预测算法停止条件时, 可以将软件可靠性预测准确度阈值作为判定条件, 或者将多个条件组合的判定条件作为算法的停止条件。

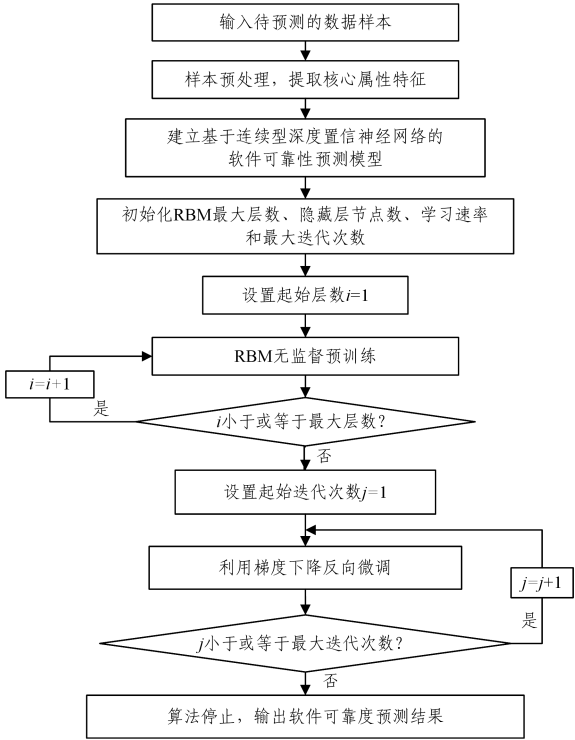


图 2 基于 CDBN 的软件可靠性预测流程

Fig. 2 Software reliability prediction process based on CDBN

4 实例仿真

为了验证连续型深度置信网络在软件可靠性预测中的性能, 分别对 CDBN 的核心参数进行差异化设置, 验证不同参数下的软件可靠性预测性能, 以便能够获得最优的 CDBN 软件可靠性预测模型。此外, 采用常用软件可靠性预测算法和本文算法进行软件可靠性预测对比, 比较多种算法的软件可靠性预测性能。

本文仿真数据的来源为美国宇航局 (NASA) MDP 库¹⁾ 中的 JM, PC 和 KC 数据集, 考虑到样本本身标注的完整性和软件失效错误模块的占比, 选取的 7 组数据集的主要分布如表 1 所列。CDBN 的训练样本和测试样本的比例为 7:3。

表 1 MDP 仿真样本

Table 1 MDP simulation samples

数据集	子集	模块数目	错误模块	错误率/%
JM	JM1	10 876	2 101	19.32
	PC3	1 560	160	10.26
PC	PC4	1 458	178	12.21
	PC5	17 186	516	3
KC	KC1	2 107	325	15.42
	KC2	523	105	20.08
	KC3	458	43	9.39

4.1 CDBN 核心参数对预测性能的影响

CDBN 的训练性能与 RBM 隐藏层节点数 K 和 $\theta = \{w, c, b\}$ 的学习速率 η 有很大的关系, 因此在 CDBN 的软件可靠

¹⁾ <https://blog.csdn.net/valark/>

性预测训练过程中,对这两个参数进行细调,以求得能够获得更高的预测性能的参数。初始化 $\eta=0.1$ 。

4.1.1 不同隐藏层节点数的软件可靠性预测

差异化设置参与 RBM 运算的隐藏层节点数 K ,CDBN 算法预测各数据集的实际软件失效模块,根据表 1 验证预测准确率,结果如图 3 所示。

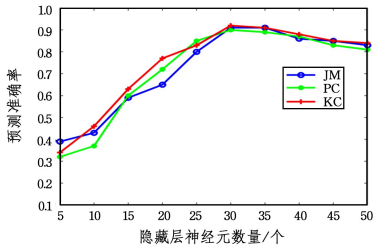


图 3 不同 K 值的预测准确率

Fig. 3 Prediction accuracy of different K values

从图 3 可以看出,JM,PC 和 KC 这 3 种数据集在 K 取值为 $[30,35]$ 时获得了较高的预测准确率;而后不断增加 K 值,可以发现预测准确率出现了轻微下降,这可能是因为 CDBN 网络结构的复杂度增加,造成了模型对测试样本的可靠性预测的准确度降低。下面继续对不同 K 值下的 MSE 进行仿真。

表 2 不同 K 值的预测 MSE 值

Table 2 Predicted MSE values of different K values

数据集	隐藏层神经元数量 K	MSE		
		最大值	最小值	平均值
JM	10	0.0841	0.0708	0.0726
	20	0.0612	0.0512	0.0525
	30	0.0567	0.0476	0.0505
	40	0.0562	0.0471	0.0501
PC	10	0.0791	0.0713	0.0731
	20	0.0632	0.0562	0.0596
	30	0.0581	0.0523	0.0535
	40	0.0583	0.0521	0.0533
KC	10	0.0799	0.0705	0.0749
	20	0.0613	0.0549	0.0585
	30	0.0543	0.0499	0.0524
	40	0.0553	0.0501	0.0523

从表 2 可以看出,随着 K 值的增加,MSE 值不断提升,表明对于 JM,PC 和 KC 这 3 种数据集,本文提出的 CDBN 算法预测的稳定性随着 K 值的增加变得更好。对于 JM 数据集, K 值从 10 增加到 30 时,MSE 均值提升了 43.76%;对于 PC 数据集, K 值从 10 增加到 30 时,MSE 均值提升了 36.64%;对于 KC 数据集, K 值从 10 增加到 30 时,MSE 均值提升了 42.94%。通过对比发现,当 K 为 30 和 40 时,两者的 MSE 值非常接近,但是 K 值增加 10,DBN 将要消耗更多的时间来完成所有数据集的可靠性预测,因此 K 值选择 30 更合适。

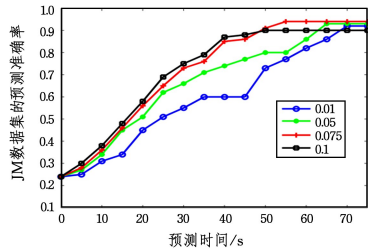
综合对比发现,对于 JM,PC 和 KC 这 3 种数据集,选择 K 值范围为 $[30,35]$ 能够获得最优的预测准确率和 MSE 性能。

4.1.2 不同学习速率的预测性能影响

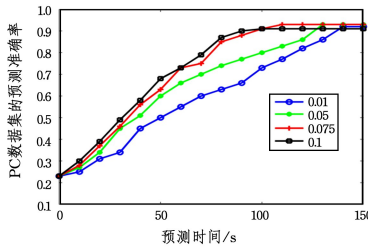
差异化设置 η 值,根据表 1 验证不同 η 值条件下 CDBN 算法的预测准确率。

图 4 给出了设置不同学习速率 η 参数时,所提 CDBN 算

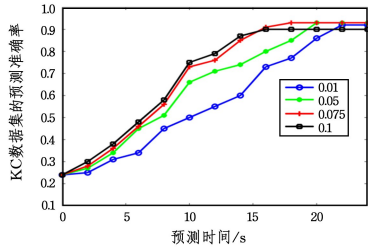
法在 JM,PC 和 KC 这 3 种数据集上的预测性能。通过对比发现,在 JM 数据集上所提 CDBN 算法的预测过程消耗的时间远长于 KC 数据集,这主要是因为 JM 数据集的样本数量比 KC 数据集多。通过对比准确率可以发现, $\eta=0.075$ 时所提 CDBN 算法能够在 3 种数据集中获得最高的准确率。



(a) JM 数据集



(b) PC 数据集



(c) KC 数据集

图 4 不同学习速率下的预测准确率

Fig. 4 Prediction accuracy of different learning rates

4.2 不同预测算法的预测性能

为了进一步验证 CDBN 算法在软件可靠性预测中的性能,分别运用常用的软件可靠性预测算法和相关向量机(RVM)^[15]、自回归滑动平均模型(ARIMA)^[16]、粒子群优化神经网络(PSO-ANN)^[17]、LSTM 神经网络^[18]和 CDBN 算法对 JM,PC 和 KC 这 3 种数据集进行仿真。此外,为了对本文算法进行具体应用软件测试,运用上述 5 种算法在浩辰机械 CAD 设计软件的真实失效数据集上进行了实际测试,不同算法的软件可靠性预测性能如表 3 所列。

表 3 中,SVR 和 ARIMA 属于经典的软件可靠性预计算法,PSO-ANN,LSTM,CDBN 均属于神经网络模型算法。从表 3 可以看出,相比经典的软件可靠性预计算法,神经网络模型算法的预测准确率更高。5 种软件可靠性预测算法在真实失效 CAD 数据集上的结果与在 PC 数据集上的结果几乎一致,验证了本文提出的可靠性预测算法的可行性。

具体来说,CDBN 算法的预测准确率最高,PSO-ANN 和 LSTM 神经网络算法次之,SVR 和 ARIMA 算法最差。但是,在预测时间方面来看,SVR 和 ARIMA 算法更具优势,其他 3 种神经网络模型算法都需要经过多层迭代来求解网络权

重等参数,因此预测时间较长。尽管 PSO-ANN 的 MSE 性能最高,但是其预测时间相对较长,导致综合性能不够理想。因此,综合来看,本文提出的 CDBN 算法的实用性更加突出。

表 3 不同算法的软件可靠性预测性能

Table 3 Software reliability prediction performance of different algorithms

数据集	算法	预测 准确率	MSE	预测 时间/s
JM	RVM	0.7835	0.0425	40.194
	ARIMA	0.8823	0.0461	45.428
	PSO-ANN	0.8912	0.0568	60.253
	LSTM	0.8961	0.0513	56.395
	CDBN	0.9352	0.0506	56.253
PC	RVM	0.8165	0.1142	87.612
	ARIMA	0.8929	0.0512	91.427
	PSO-ANN	0.8862	0.0552	136.254
	LSTM	0.8834	0.0537	121.674
	CDBN	0.9271	0.0533	117.371
KC	RVM	0.8135	0.1346	7.472
	ARIMA	0.8952	0.0617	10.502
	PSO-ANN	0.9108	0.0556	24.025
	LSTM	0.9013	0.0523	19.353
	CDBN	0.9286	0.0525	17.059
真实失效 CAD	RVM	0.8314	0.1236	86.512
	ARIMA	0.9005	0.0546	92.102
	PSO-ANN	0.8721	0.0563	137.552
	LSTM	0.8604	0.0562	120.824
	CDBN	0.9126	0.0572	117.453

结束语 本文采用连续型深度置信神经网络进行软件可靠性预测,通过合理设置 CDBN 算法的隐藏层节点数和学习速率值,能够获得较好的预测性能。与常用的软件可靠性预测算法相比,本文算法能够获得更高的预测准确率和更优的 MSE 值。后续研究将对 CDBN 算法的权重求解进行进一步优化,以提高 CDBN 算法在软件可靠性预测中的适用度。

参 考 文 献

[1] ZHANG D P, LIU G Q, ZHANG K. Software defect prediction model based on GMDH causality [J]. Computer Science, 2016 (7):171-176.

[2] XU L. Software Reliability Model Considering Incomplete Error Removal and Degradation of Detection Rate [J]. Journal of Chongqing University of Technology(Natural Science), 2019, 33 (8):132-137.

[3] JAIN R, SHARMA A. Assessing software reliability using genetic algorithms [J]. Journal of Engineering Research, 2019, 16(1):11-17.

[4] JABEEN G, LUO P, AFZAL W. An improved software reliability prediction model by using high precision error iterative analysis method[J]. Software Testing, Verification and Reliability, 2019(29):67-75.

[5] GONG L, JIANG S, BO L, et al. A Novel Class-Imbalance Learning Approach for Both Within-Project and Cross-Project Defect Prediction[J]. IEEE Transactions on Reliability, 2020, 69(1):40-54.

[6] COTRONEO D, PIETRANTUONO R, RUSSO S. RELAI Tes-

ting: A Technique to Assess and Improve Software Reliability [J]. IEEE Transactions on Software Engineering, 2016, 42(5): 452-475.

[7] TAMURA Y, YAMADA S. Software Reliability Model Selection Based on Deep Learning with Application to the Optimal Release Problem[J]. Journal of Industrial Engineering and Management Science, 2016, 2016(1):43-58.

[8] LIU J, WU N Q, QIAO Y, et al. Short-Term Traffic Flow Forecasting Using Ensemble Approach Based on Deep Belief Networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2020(99):1-14.

[9] QIAO J, PAN G, HAN H. A regularization-reinforced DBN for digital recognition[J]. Natural Computing, 2019, 18(4): 721-733.

[10] YU L, ZHOU R, TANG L, et al. A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data[J]. Applied Soft Computing, 2018(69):192-202.

[11] ABDEL-ZAHER A M, ELDEIB A M. Breast cancer classification using deep belief networks[J]. Expert Systems with Applications, 2016(46):139-144.

[12] WEN C L, LU F Y. Review on Deep Learning Based Fault Diagnosis [J]. Journal of Electronics & Information Technology, 2020, 42(1):234-248.

[13] ZHENG J, FU X, ZHANG G. Research on exchange rate forecasting based on deep belief network[J]. Neural Computing & Applications, 2019, 31(1):573-582.

[14] LIANG H, LIU Y, SHENG G, et al. Fault-cause identification method based on adaptive deep belief network and time-frequency characteristics of travelling wave[J]. Generation, Transmission & Distribution, IET, 2019, 13(5):724-732.

[15] LIANG H T, XU J L. Software Reliability Prediction Model Based on Hybrid Kernels RVM[J]. Computer Science, 2016, 43(11):257-259.

[16] RAY S, BHATTACHARYYA B, PAL S. Statistical Modeling and Forecasting of Food Grain in Effects on Public Distribution System: An application of ARIMA Model[J]. Indian Journal of Economics and Development, 2020, 12(4):739.

[17] PRATIK, ROY, GHANSHAYM, et al. Forecasting of Software Reliability Using Neighborhood Fuzzy Particle Swarm Optimization Based Novel Neural Network[J]. IEEE/CAA Journal of Automatica Sinica, 2019, 6(6):78-96.

[18] CHEN L Q, LIU W, HE X. Reliability prediction of CPS software based on LSTM neural network [J]. Computer Engineering and Design, 2019, 40(5):1343-1348.



Qi Hui, born in 1981, master, associate professor, graduate supervisor. Her main research interests include machine learning and computational intelligence.