文章编号: 1006-2475(2017)02-0036-04

基于 DA-SVM 的软件缺陷预测模型

甘 露,臧 洌,李 航

(南京航空航天大学计算机科学与技术学院,江苏 南京 210016)

摘要: 特征提取是软件缺陷预测技术研究中的重要环节,而现有的特征提取方法无法准确获得特征之间的非线性依赖关系,因而无法提高软件缺陷预测的准确性。针对该问题,本文构建基于降噪编码器和支持向量机的软件缺陷预测模型(Denoising Autoencoder Support Vector Machine,DA-SVM)。首先利用降噪编码器进行特征提取,然后将提取的特征作为支持向量机的输入向量,最后再进行软件缺陷预测。实验结果表明,DA-SVM 提高了软件缺陷预测的准确度,同时降低了历史数据中的噪声,增强了软件预测模型的鲁棒性。

关键词: 特征提取; 软件缺陷预测; 降噪自动编码器; 支持向量机

中图分类号: TP311 文献标识码: A doi: 10.3969/j.issn.1006-2475.2017.02.007

Software Defect Prediction Model Based on DA-SVM

GAN Lu, ZANG Lie, LI Hang

(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: Feature extraction is an important step in software defect prediction technology research. However, the existing feature extraction cannot accurately obtain the nonlinear dependence relations among features, thus these methods are unable to improve the accuracy of software defect prediction model. In this paper, to solve this question we propose a software defect prediction model (Denoising Autoencoder Support Vector Machine, DA-SVM) which is based on denoising autoencoder and Support Vector Machine. Firstly, the model extracts features by using denoising autoencoder, secondly uses these features as input of support vector machine, lastly, uses this model to predict bugs. Experimental results show that DA-SVM not only improves the accuracy of software defect prediction model, but also reduces the noise of history data and enhances the robustness of the software defect prediction model.

Key words: feature extraction; software defect prediction; denoising autoencoder; support vector machine

0 引 言

近 30 年来,软件缺陷预测[1]一直是软件测试领域中重要的研究课题。在软件开发周期中,利用软件缺陷预测技术预测当前开发模块的缺陷情况,从而决定是否可以进入下一阶段的开发。软件缺陷预测技术不仅可以调节软件的开发过程,在测试阶段还可以将有限的资源集中在缺陷较多的模块,合理利用资源,提高软件的安全性能,因此,软件缺陷预测技术的研究具有深远意义。

软件属性对于构建软件缺陷预测模型起着重要的作用。软件缺陷模型的性能和有效性取决于软件模块的特征属性,利用这些特征属性可以快速有效地对软件模块进行预测。用来描述软件的属性有很多,由于某些属性不包含重要的信息或存在不相关的冗余信息,从而会影响软件缺陷预测模型的准确性。如果属性选择不当,构建的缺陷预测模型的预测性能则会降低。因此,为了提高缺陷预测模型的效率和性能,采用合适的特征选择[2-3]算法对于构建软件缺陷预测模型尤为重要。

1 相关工作

软件缺陷预测技术是对历史缺陷数据进行一系列的数据预处理和特征提取,然后将提取的特征通过特定的算法进行训练,最后得到缺陷预测模型,并对软件进行缺陷预测。在这个过程中,由于各种混合组

收稿日期: 2016-06-08

作者简介: 甘露(1991-),女,安徽宁国人,南京航空航天大学计算机科学与技术学院硕士研究生,研究方向: 软件测试; 臧洌(1964-),女,副教授,硕士,研究方向: 网络安全及软件可靠性; 李航(1992-),男,硕士研究生,研究方向: 机器学习。

合而导致的维数灾难是影响软件缺陷预测准确性的一个重要因素,因此,特征选择是进行软件缺陷预测的一个重要环节,它能够有效地解决因特征冗余而造成的预测精度下降问题。简单地说,如果能通过最基本的属性来形成比较容易判断是否为缺陷的特征就能提高分类的准确性,例如在图像识别中,如果能通过最开始的像素来形成比较明显的人的五官这样的特征,则更有利于对这样的图片是否是人进行判断。降噪自动编码器就具有这样的特征提取作用。目前软件缺陷预测领域主要应用的特征选择方法主要分为 2 类: 有监督特征提取方法和无监督特征提取方法。

有监督提取方法主要是 Person[4] 提出的主成分分析(Principal Component Analysis,PCA) 。PCA 是一种可以将多个变量通过线性转换从而选择出少数重要变量的一种统计分析方法。有监督特征提取方法主要是由 Fisher[5] 在 1936 年提出的线性鉴别分析方法(Linear Discriminant Analysis,LDA) 。LDA 可以寻求属性的最优线性组合来保证原始数据的不变性。在此基础上,Wei[6] 等人提出了一种基于生成视图的特征选择方法,可以探索信息之间最佳的联系方式。但是这 2 种特征提取方法都只能通过线性转换进行特征提取,无法对特征进行非线性转换,从而具有一定的局限性。如果将输入属性记为($x_1, x_2, \ldots, x_n$) ,预测结果用 y 表示,那么 PCA 和 LDA 只能得到 $y = a_1x_1 + a_2x_2 + \ldots + a_nx_n$ 的线性关系,而如果某个属性对于是否为缺陷是非线性的关系时,如代码行数和是否为缺陷存在 $y = ax_1^k$ 时,PCA 和 LDA 并不能得到,而且降噪自动编码器[7-8] (Denoising Autoencoder,DA) 可以压缩特征并提取特征间非线性的依赖关系,通过发现并利用属性之间的非线性依赖关系可以极大地提高分类器的预测性能,这是降噪自动编码器与其它特征选择方法最大的不同与优势。同时,降噪自动编码器对于数据中夹杂的噪声具有一定的降噪能力,减少噪声对于软件缺陷模型预测能力的干扰,能够在一定程度上增强软件缺陷模型的鲁棒性。综上所述,本文利用降噪自动编码器网络进行特征提取,利用支持向量机[9-10] 作为分类模型,构建了 DA-SVM 软件缺陷预测模型,并对缺陷数据进行预测。

2　DA-SVM 软件缺陷预测模型

2.1　DA-SVM 框架图

本文提出的 DA-SVM 软件缺陷预测模型主要分为 2 个部分,降噪自动编码器网络和支持向量机。其中,降噪自动编码器主要实现特征提取功能,通过将输入数据 x 进行一层层的同义转换,降低原始输入数据的维度,从而获得更加简洁的输入数据的同义表示,依据该同义表示重构 x,因此,经过自动编码器所获得的这个表示获取了原始输入的主要特征,所以利用降噪自动编码器网络可以进行特征提取。支持向量机[11] (Support Vector Machine,SVM) 在处理小样本、非线性及高维模式识别中表现出许多特有的优势,它的主要功能是对降噪编码器网络获取的同义表示进行学习,通过学习历史数据对 SVM 的参数进行调整,得到分类模型,最后利用该模型对软件模块进行预测,具体过程如图 1 所示。

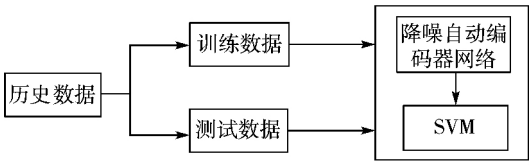

图 1　DA-SVM 框架图

2.2　降噪自动编码器网络

降噪自动编码器[12-13] 是深度学习理论中一种可以寻求输入向量的同义表示算法,是神经网络的另外一种扩展,本质上仍然是一种多层神经网络,神经网络[14] 在解决缺陷预测问题上得到了广泛应用,但是多层神经网络的参数调节仍然存在问题,而深度学习的出现恰好解决了这个问题。Wang 等[15] 提出利用深度信念网(Deep Belief Networks,DBN) 自动学习源代码的语义特征,并利用学习到的特征训练和构建缺陷预测模型。Yang 等[16] 利用 DBN 对 14 个基本特征进行特征提取,然后利用回归算法建立分类模型进行软件缺陷预测。降噪自动编码器与 DBN 相比,不仅具有 DBN 的特征提取能力还增加了对缺失数据的抗干扰性,因此,本文利用降噪自动编码器网络来实现特征提取。为了增强软件缺陷预测模型的鲁棒性,将原始的输入 x 通过一个随机映射 $x' \sim q_D(x \mid x')$ 得到一个损坏的输入版本 x',定义一个联合分布概率:

$$q_0(x, x', y) = q_0(x)\, q_D(x' \mid x)\, \delta_{f_\theta(x')}(y) \tag{1}$$

其中,y 是 x 的确定性函数,因为选择 $L_H(x, z) = H(\beta_x \| \beta_z)$ 作为损失函数,因此随机梯度下降的目标函数为:

$$\theta^*, \theta'^* = \underset{\theta, \theta'}{\operatorname{argmin}} E q_0(x, x') \left[L_H(x, g_{\theta'}(f_\theta(x'))) \right] \tag{2}$$

在具体的实验过程中,对于每一个输入 $x = (x_1, x_2, \ldots, x_n)$ 选取固定的 v_d 个分量,并将其置为 0,同样地,利用 $y = f_\theta(x') = s(W x' + b)$ 以及 $z = g_\theta'(y) = s(W'y + b')$ 重构输入向量,利用 $L_H(x, z) = H(\beta_x \| \beta_z)$ 作为重构损失函数,使得重构的 z 尽可能地靠近 x,具体过程如图 2 所示。

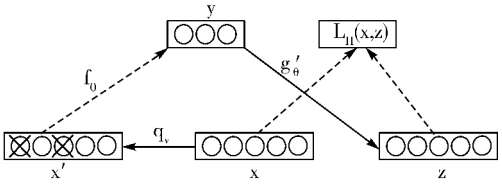

图 2 降噪自动编码器

因此,即使输入向量 x 夹杂一定的噪音,也可以通过已经训练好的降噪自动编码器尽可能地还原原来的输入样本,从而降低噪音的干扰,增强软件缺陷预测模型的准确性。

2.3 SVM 分类方法

SVM 是在 1995 年由 Cortes 和 Vapnik 提出的一种基于学习理论和结构风险最小化原则的统计机器学习算法,它的实质是一种分类算法,在解决小样本、非线性及高维模式识别中表现出许多特有的优势。SVM 的基本思想是求解核函数和二次线性规划问题,可以通过核函数将线性不可分样本映射到高维空间,找到最优的分类平面,从而进行分类计算。SVM 常用的核函数有 3 种: 径向基核函数(Radial Basis Function,RBF) 、多项式函数和线性核函数,其中径向基核函数因为有较宽的收敛范围,被广泛地应用在 SVM 中。所谓最优的分类平面就是求解一个平面 $g(x) = wx + b$,使得这个平面到达两类样本的几何间距达到最大。在这里,x 就是 SVM 的输入,也就是利用降噪编码器网络所得到的最终输出。如果样本线性可分,则可以转化为下列求解问题:

$$\min \frac{1}{2} \| w \|^2 \tag{3}$$

$$\text{S.t. } y_i[(wx_i) + b] - 1 \geq 0,\ i = 1,2,\ldots,L \tag{4}$$

其中 L 是样本数。

根据优化理论,上述问题可以转化为求解它的对偶问题:

$$\min \frac{1}{2} \sum_{i=1}^{L} \sum_{j=1}^{L} y_i y_j (x_i, x_j) a_i a_j - \sum_{j=1}^{L} a_j \tag{5}$$

$$\text{S.t. } \sum_{i=1}^{L} y_i a_i = 0,\ a_i \geq 0,\ i = 1,2,\ldots,L \tag{6}$$

求解上述问题,得到一个向量 $a = (a_1, a_2, \ldots, a_n)^T$,得到的最优超平面的分类函数为:

$$f(x) = \operatorname{sgn}\{(wx) + b\} = \operatorname{sgn}\{\sum_{i=1}^{L} y_i a_i (x_i x_j) + b\} \tag{7}$$

通常,所用的样本是线性不可分的,因此,可以使用核函数来解决这个问题。核函数将输入数据映射到更高维的特征空间,然后样本在这个高维特征空间可分。针对线性不可分的样本的求解问题可以转化为下列公式:

$$\min \frac{1}{2} \sum_{i=1}^{L} \sum_{j=1}^{L} y_i y_j K(x_i, x_j) a_i a_j - \sum_{j=1}^{L} a_j \tag{8}$$

$$\text{S.t. } \sum_{i=1}^{L} y_i a_i = 0,\ C \geq a_i \geq 0,\ i = 1,2,\ldots,L \tag{9}$$

最后,得到的分类函数为:

$$f(x) = \operatorname{sgn}\{\sum_{i=1}^{L} y_i a_i K(x_i x_j) + b\} \tag{10}$$

因此,当将经过降噪自动编码器网络处理得到的 y 作为 SVM 的一个测试样本 x 时,利用式(10) 便可以得到 f(x) 为 0 或 1,即可以判断该样本是否有无缺陷的,于是便可以对一个软件模块是否含有缺陷进行预测,以决定后续工作将如何进行。

3 实验内容与结果分析

3.1 实验内容

DA-SVM 软件缺陷预测模型所使用的自动编码器网络结构为 22-20-10-10,第一层有 22 个可见节点,代表缺陷数据集的 22 个属性,之所以选择这 22 个属性集是因为在 NASA 的子数据集都拥有这 22 个属性,这些属性的具体信息如表 1 所示,20 个隐藏节点第一次特征提取后的属性数量,再将提取的属性进行 2 次提取,最终得到 10 个特征,将 10 个特征作为 SVM 的输出,通过 SVM 得到最终的分类结果。

实验使用 NASA 官网发布的 MDP 软件缺陷数据集,是用 C/C++ 语言开发的飞船仪表、卫星飞行控制、科学数据处理和存储地面数据管理等模块的软件缺陷数据。本实验从该数据集中选取 JM1,KC1,MC1,PC2 和 PC5 这 5 个数据量较多的子数据集。每个子数据集是在采用不同的开发环境网站上收集到的,因此,每一个分类器在每一个缺陷数据集上的分类性能是不同的,详细描述数据在表 2 中给出。

表 1 属性表

Loc_total	Num_unique_operands	Halstead_prog_Time
Loc_bank	Halstead_Length	Cyclomatic_complexity
Loc_code_and_comment	Halstead_volume	Design_complexity
Loc_comments	Halstead_Level	Essntial_complexity
Loc_executable	Halstead_Difficulty	Branch_cout
Num_operators	Halstead_content	Number_of_Lines
Num_operands	Halstead_Effort	
Num_unique_operators	Halstead_Error_Est	

实验采取的对比模型是用 PCA 和 LDA 进行特征提取的支持向量机模型,简称 PCA-SVM 模型和 LDA-SVM 模型,为比较 3 种软件缺陷预测模型的性能,实验依据混淆矩阵采用准确率(accuracy) 、精准率(precision) 、召回率(recall) 和 F1-度量(F1-measure) 这 4 个指标进行统计,在划分训练集和测试集时,实验采取十折交叉法,每组实验重复 10 次,结果取 100 次实验的平均值。

表 2　NASA MDP 数据集

	语言	千行代码数	模块数	缺陷率/%
JM1	C	315	10878	19
KC1	C++	43	2107	15
MC1	C&C++	63	9466	0.7
PC2	C	26	5589	0.4
PC5	C++	164	17186	3

在实验过程中,首先要对每个训练集进行预处理,它包括 3 个部分的内容: 1) 数据的归一化; 2) 移除缺失值; 3) 删除冗余值。之所以进行归一化是因为每个不同属性的取值范围可能会在不同的量级,这样会导致倾斜从而无法比较结果的差异性。对于每个属性的缺失值可由降噪自动编码器进行处理。在某些数据集中存在数据的冗余,因此这些相同的数据只会保存一次,其它的将会被移除,减少时间成本。同样地,对于每个测试集也要进行上述处理过程。在使用 PCA-SVM 模型和 LDA-SVM 模型进行实验时,数据预处理完全与 DA-SVM 模型相同,它与 DA-SVM 模型最大的不同在于: DA-SVM 采用降噪自动编码器进行特征提取。

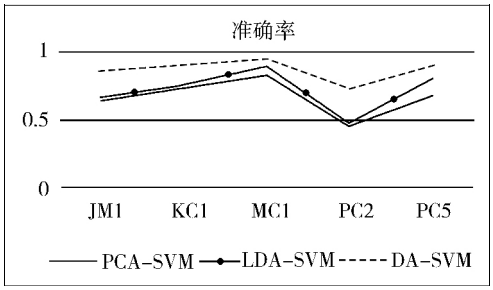

图 3　准确率指标结果

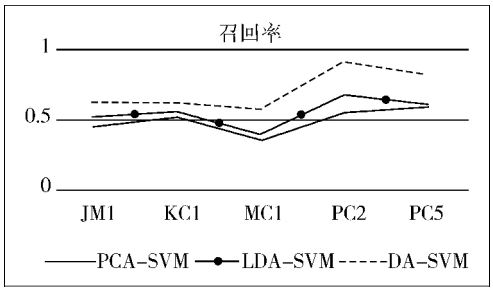

图 4　召回率指标结果

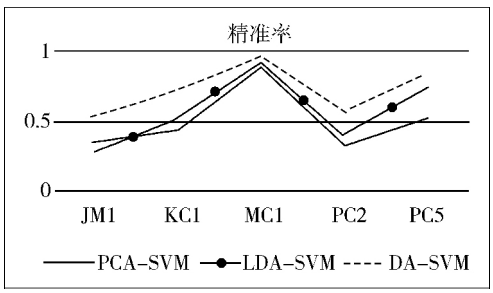

图 5　精准率指标结果

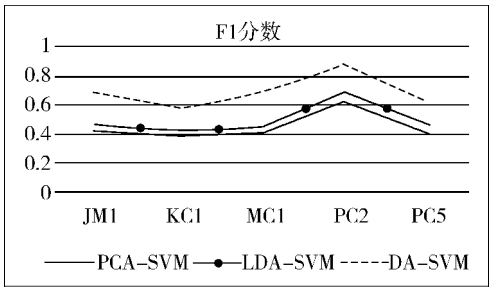

图 6　F1 分数指标结果

3.2　实验结果及分析

实验对 DA-SVM,PCA-SVM 以及 LDA-SVM 预测模型在每个数据集上的准确率、召回率、精准率和 F1 分数 4 个指标进行统计,具体如图 3 ~ 图 6 所示。图 3 ~ 图 6 分别为本文方法与 PCA-SVM 和 LDA-SVM 在 5 个数据集上的预测结果对比。从图中可以看出,基于 PCA 和 LDA 建立的模型的预测结果较差,这是因为这 2 种方法是基于线性关系而进行特征选择,对于特征之间的非线性依赖关系无法转化,所以预测模型的准确度不高,而 DA-SVM 模型不仅对于线性关系可以转化,对于非线性关系也可转化,所以 DA-SVM 在准确率、召回率、精准率和 F1 分数这 4 个指标上普遍比 PCA-SVM 和 LDA-SVM 要高,在各个指标的平均值上比 PCA-SVM 高 20% 左右,比 LDA-SVM 高 15% 左右,因此,DA-SVM 模型的缺陷预测能力比 PCA-SVM 模型和 LDA-SVM 模型要高,预测的结果更加准确。

4　结束语

本文针对现有的特征提取方法无法提取样本的非线性特征的不足,使用降噪编码器网络进行特征提取,并将该方法与 SVM 分类方法相结合,构建 DA-SVM 模型,并在 NASA 的 MDP 数据集上进行实验,将实验结果与 PCA-SVM 预测模型和 LDA-SVM 预测模型的结果进行比较,结果表明,DA-SVM 预测模型具有更高的准确性,在后续的工作中,笔者将进一步研究降噪编码器网络与其它分类算法构成的预测模型的准确性,使得降噪编码器在软件缺陷预测领域能发挥更大的优势。

参考文献:

[1] 陈翔,顾庆,刘望舒,等. 静态软件缺陷预测方法研究[J]. 软件学报, 2016,27(1):1-25.

[2] Mandal P, Ami A S. Selecting best attributes for software defect prediction [C] // 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIEC-ON-ECE). 2015: 110-113.

[3] Khan J I, Gias A U, Siddik M S, et al. An attribute selection process for software defect prediction [C] // 2014 International Conference on Informatics, Electronics & Vision (ICIEV). 2014: 1-4.

(下转第 44 页)

图的技术细节,并减少嵌入式程序代码空间。

参考文献:

[1] 王忠,张晓莉,李忠安,等. 继电保护装置自动测试系统设计[J]. 电力系统保护与控制,2015,43(5):130-135.

[2] 赵志华. 图形化编程与继电保护装置开发[J]. 电力自动化设备,2004,24(2):70-72.

[3] 李金,孙斌,张静. 继电保护装置可视化编程反馈回环问题研究[J]. 电力系统保护与控制,2013,41(21):15-19.

[4] 仲伟,丁宁,吴参林,等. 图形化编程的继电保护软件平台设计[J]. 电力系统保护与控制,2011,39(3):100-104.

[5] 张云,尹秋帆,胡道徐. 继电保护装置开发平台软件系统架构与设计[J]. 电力系统及其自动化学报,2005,17(4):20-23.

[6] 邓秋娥,杜奇壮,卢娟. 可视化编程在微机保护中的实现[J]. 继电器,2008,36(3):1-4.

[7] 蒋泽军,李艳艳,王丽芳. 基于插件技术可视化测控系统的研究[J]. 测控技术,2014,33(7):110-113.

[8] 王黎明,王帼钕,周明媛,等. 程序流程图到代码的自动生成算法[J]. 西安电子科技大学学报(自然科学版),2012,39(6):70-77.

[9] 张春合,余群兵,陆征军,等. 保护测控一体化装置的研制[J]. 电工技术,2012(9):65-66.

[10] 张磊,陈宏君,吴相楠,等. 基于扩展 103 规约的保护装置通信与调试系统设计[J]. 电力系统保护与控制,2015,43(21):126-130.

[11] 陈海宏,张静,万书亭. 基于 GIS 的电力通信线路管理系统开发与应用[J]. 电力系统及其自动化学报,2013,25(2):26-30.

[12] 何鹏飞,何平,张松阳,等. 组件技术在嵌入式系统中的应用[J]. 计算机系统应用,2014,23(6):220-223.

[13] 刘克金,陈宏君,冯亚东,等. 新一代控制保护系统可视化编程软件设计与实现[J]. 工业控制计算机,2014,27(10):82-84.

[14] 陈宏君,刘克金,冯亚东,等. 新一代保护测控装置配套工具软件设计与应用[J]. 电力系统自动化,2013,37(20):92-96.

[15] 熊蕙,张磊,周磊,等. 嵌入式程序辅助调试软件设计[J]. 工业控制计算机,2014,27(6):22-23.

[16] 周磊,陈宏君,张磊,等. 保护控制模块化配置工具软件设计[J]. 工业控制计算机,2015,28(6):20-22.

[17] 莫鹏飞,陈志坚,杨军,等. 嵌入式处理器的在线调试器设计与实现[J]. 计算机应用与软件,2012,29(12):302-305.

(上接第 39 页)

[4] Pearson K. Principal components analysis [J]. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 1901, 6(2): 559.

[5] Fisher R A. The use of multiple measurements in taxonomic problems [J]. Annals of Eugenics, 1936, 7(2): 179-188.

[6] Wei Xiaokai, Cao Bokai, Yu Philip S. Unsupervised feature selection on networks: A generative view [C] // The 30th AAAI Conference on Artificial Intelligence. 2016.

[7] Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders [C] // Proceedings of the 25th ACM International Conference on Machine Learning. 2008: 1096-1103.

[8] Wang Hao, Shi Xingjian, Yeung Dit-yan. Relational stacked denoising autoencoder for tag recommendation [C] // Proceedings of the 29th AAAI Conference on Artificial Intelligence. 2015: 3052-3058.

[9] Elish K O, Elish M O. Predicting defect-prone software modules using support vector machines [J]. Journal of Systems and Software, 2008, 81(5): 649-660.

[10] 姜慧研,宗茂,刘相莹. 基于 ACO-SVM 的软件缺陷预测模型的研究[J]. 计算机学报,2011,34(6):1148-1154.

[11] 王涛,李伟华,刘尊,等. 基于支持向量机的软件缺陷预测模型[J]. 西北工业大学学报,2011,29(6):864-870.

[12] Kamyshanska H, Memisevic R. The potential energy of an autoencoder [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(6): 1261-1273.

[13] Lander S, Shang Y. EvoAE—A new evolutionary method for training autoencoders for deep learning networks [C] // Proceedings of the 39th IEEE Annual Computer Software and Applications Conference (COMPSAC). 2015, 2: 790-795.

[14] Jindal R, Malhotra R, Jain A. Software defect prediction using neural networks [C] // International Conference on Reliability, INFOCOM Technologies and Optimization. 2014: 1-6.

[15] Wang Song, Liu Taiyue, Tan Lin. Automatically learning semantic features for defect prediction [C] // Proceedings of the 38th ACM International Conference on Software Engineering. 2016: 297-308.

[16] Yang Xinlin, Lo David, Xia Xin, et al. Deep learning for just-in-time defect prediction [C] // 2015 IEEE International Conference on Software Quality, Reliability and Security. 2015: 17-26.