基于机器学习的软件缺陷预测

黃柏曛

2020 年 10 月 5 日

1 摘要

随着虚拟仪器软件系统规模逐渐扩大, 其复杂程度也逐渐提高, 对于缺陷容忍度较低的高风险软件来说发生故障导致的后果很严重, 虚拟仪器软件的性能面临严峻的考验。[1] 软件缺陷预测是软件工程领域中与软件质量保证密切相关的重要的研究课题, 它对提高软件系统质量和优化测试资源分配都有重要意义。在软件工程数据挖掘领域中, 基于机器学习的静态软件缺陷预测根据软件历史仓库数据, 采用缺陷相关的度量对软件代码或开发过程进行分析, 利用机器学习方法来预测软件项目中待测试程序模块的缺陷倾向性或缺陷数量。[2]

2 關鍵字

软件缺陷、机器学习、预测、深度学习

3 前言

IEEE729-1983 对软件缺陷进行了定义：一方面是指系统开发或维护过程中存在的错误、毛病等各种问题；另一方面是指系统没有达到客户需求的某种功能的失效。而软件缺陷的危害是巨大的，在软件开发早期的软件缺陷预测就很重要了。[3] 软件缺陷预测技术是指按照软件的基本属性规模、复杂性、开发方法和过程，以及已知缺陷来预测潜在但还未被发现的

缺陷 [4]。软件缺陷预测技术能帮助测试人员掌握软件失效模式、了解质量状态，并决定软件是否交予用户使用。软件缺陷预测技术可分成动态缺陷预测与静态缺陷预测 [5]。动态缺陷预测技术是以缺陷产生时间为基础，对系统缺陷随时间分布实施预测的技术；该技术利用时间分布统计、挖掘软件的缺陷，寻找缺陷基于软件开发周期的引入与移除规律。静态缺陷预测技术是指采用软件规模、复杂度、开发过程等可度量缺陷的元素及已有缺陷，预测软件潜在但还未暴露的缺陷；该技术以缺陷尽早检测为原则，既可减少缺陷修复成本，又能缩短缺陷修复时间。[6]

4 緒論

4.1 研究背景

作为计算机实现各项功能，辅助人们进行各项活动的载体，在当今世界各行各业中发挥着重要的作用。然而伴随各类软件的开发，软件缺陷的产生是不可避免的。关于软件缺陷 (defect) 的定义，学术界和产业界还有其他与之相关的术语，例如错误 (error)、故障 (fault)、失误 (mistake)、失效 (failure) 等。其中缺陷是软件中已经存在的部分，且可以通过修复进行消除。[7] 错误是一种人为的行为，如开发过程中无意识的技术错误，其后果必定导致软件某部分功能产生缺陷。失效和故障都是软件运行时产生的，失效是软件不能再完成规定的功能，而故障是指软件在运行时没有输出与设计相符合的结果。除人为操作失误以外，软件缺陷正逐渐成为导致计算机系统失效和停机的主要因素。[8] 在某些对可靠性有着严格要求的行业，譬如航空航天、医疗管理、金融系统等，如果潜在的软件缺陷得不到及时的清除，则可能造成很大的损失，例如：1999 年 NASA 在制造火星气候轨道探测器时在程序中使用了英制单位，而不是预定的公制单位，导致探测器的推进系统无法按照预设轨道运作，使其进入大气层的高度有误，最终瓦解碎裂并造成了 3.27 亿美元损失。软件缺陷预测技术术通过对软件开发过程或者软件代码进行分析，以建立相应的预测模型，并对软件中潜在的缺陷进行预测，其结果可以为测试人员提前定位可能产生缺陷的模块或者提前预知模块中所含的缺陷数目，从而帮助和指导决策者分配有限的资源以进行更有针对性的软件测试。软件缺陷预测在软件工程中具有重要

的意义：(1) 使软件开发人员缩短开发周期，节省资源，更快的开发出高可靠性的软件；(2) 使测试过程和资源能更集中地针对易产生缺陷的模块，从而加快缺陷发现和修复的速度，从而降低软件开发的成本；(3) 通过提高测试的过程以提高软件的质量。[9]

4.2 国内外研究现状

软件缺陷预测技术最早是由 Briand 在 1992 年的时候提出的。1993 年，Briand、Basili 和 Hetmanski[10] 等人在 ADA 系统的 146 个组件上应用逻辑回归、分类树以及 OSR 方法进行缺陷预测研究，结果表明，通过使用 OSR 技术之后，预测模型的正确性和完整性达到了 90%。1997 年，Khoshgoftaar 和 Allen[11] 等人在包含 1300 万行代码的电信系统上建立了人工神经网络模型和判别模型，并将这两种模型进行比较，发现人工神经网络模型的预测性能优于判别模型。Evett[12] 等人于 1998 年首次将遗传方法应用于军事通信系统和电信系统的缺陷预测。他们应用了八个级别指标包括 Halstead 科学度量和 McCabe 环路复杂路度量，得到了很高的预测精度。Denaro[13] 在 2000 年通过使用逻辑回归在天线配置软件的 37 个指标上进行了软件缺陷估计，发现静态软件度量和软件缺陷数量具有一定的相关性。Summers[14] 于 2002 年提出的多层感知器（MLP）是一种有效的软件缺陷研究技术。Mahaweerawa 于 2002 年第一次使用模糊聚类来预测软件缺陷，他应用径向基函数（RBF）来预测软件的缺陷，RBF 方法比 MLP 更广泛应用在软件故障预测研究中。Menzies 在 2004 年在 PROMISE 数据集上构建了贝叶斯网络缺陷预测模型，并用 PD 和 PF 作为评估结果的性能指标。Nagappan[15] 等人在 2006 年提出了从度量中提取原子组件并使用这些组件进行缺陷预测，Kim 等人在 2007 年通过获取在软件改变历史数据缓存位置来预测缺陷, 之后 Zimmermann 和 Nagappan[16] 又提出了基于结构测量模型的依赖图，Menzies[17] 等人还在 2010 年建立了基于静态编码属性的缺陷预测模型。Malhotra 在 2014 年建立了神经网络预测模型对软件缺陷进行研究，后续有各种对神经网络的优化模型被提出如：PSO-BP、SA-BP 等。David[18] 在 2015 年将深度学习引入了软件缺陷预测技术研究领域。

5 软件缺陷预测模型及算法

软件缺陷预测模型主要包括基于监督学习的预测模型、基于半监督学习的预测模型和基于无监督学习的预测模型三种。[19] 其中，基于监督学习的预测模型需要足够的度量指标数据和缺陷标记数据作为训练的历史数据; 基于半监督学习的预测模型需要所有模块的度量指标和部分缺陷标记数据作为训练的历史数据; 基于无监督学习的预测模型只需要度量指标数据作为训练的历史数据。[20] 但由于存在数据集有限、收集高质量数据困难、软件数据类不平衡分布等负面问题，在建立软件缺陷预测模型时，必须研究相关内容，设法解决这些负面因素带来的不良影响。[21] 几种常用的算法：

1. 决策树算法，决策树是一种基本的分类与回归方法，用于表示决策和相应的决策结果对应关系的树，揭示数据中的结构化信息，可以归纳为 if-then 规则的集合，主要优点是建模速度快，可读性强。分类与回归树（CART，classification and regression tree）模型由 Breiman 等人 [22] 在 1984 年提出，是应用广泛的决策树学习算法，如果目变量是分类变量，称构建的树为分类树，叶节点代表类别；如果目标变量是连续的，则称此树为回归树，其叶节点代表数值。[23] 而随机森林算法（Random Forest）是一个包含多个决策树的分类器，利用多棵树对样本进行训练并进行分类和回归预测。[24]

2. 回归算法，回归问题研究的是因变量 y 和自变量 x 的关系，本文主要以最简单的线性回归模型为基础，即 y 是 x 的一次函数。[25] 另一类较常用的是逻辑回归，主要针对二分类问题。线性回归得到的函数值是在 $(-\infty, +\infty)$，但对于缺陷预测中的分类问题，预期的结果只有两类：有缺陷或无缺陷，这就需要逻辑回归算法。[26] 逻辑回归是以线性回归为基础的一个非线性模型，通过一个逻辑回归函数将线性回归的值映射到 [0,1] 之间，能够轻松解决二值分类问题。此时因变量 y 的取值为 0,1，0 代表无缺陷，1 代表有缺陷。

3. 贝叶斯网络贝叶斯网络也称为信念网络，是一种概率的图模型，它假定类不是条件独立的，允许表示属性子集之间的依赖关系，这是它与前面所述算法最大的不同。[27] 贝叶斯网络由两部分组成：有向无环图

（DAG）和条件概率表（CPT）的集合。[28] 有向无环图的节点代表离散的或连续的随机变量，这些随机变量可以对应数据集中的实际属性，也可以是隐藏变量、未知参数。若两个节点之间用带箭头的弧线相连，表示这两个节点是具有因果关系的，存在概率依赖而不是条件独立的；反之则这两个节点条件独立。若存在一条弧从节点 B 指向 A，则称 B 是 A 的双亲或直接前驱，而 A 是 B 的后代。[29] 贝叶斯网络提供了一种自然的表示因果关系的方法，用来发现数据间的潜在关系 [30]。由于缺陷预测问题中导致缺陷产生的影响因素繁多，贝叶斯方法以其独特的不确定性知识表达形式、丰富的概率表达形式、综合先验知识的增量学习特性取得了良好的分类、回归效果。

4. 神经网络最初的目的是寻求开发和检验神经的计算模拟。神经网络的计算模型模仿动物的中枢神经系统（尤其是脑），通常呈现为相互连接的“神经元”结构，它可以从输入的值进行机器学习以及模式识别，通过调整内部大量节点之间相互连接的关系，进行分布式并行信息处理。[31] 神经网络的具体模型非常多，其中运用最广泛的是多层感知机（MLP，Multi-layer Perceptron）神经网络，它是使用标准 BP 算法训练的神经网络，是一种非线性统计性数据建模工具，常用来对输入与输出间复杂的关系进行建模，或用来探索数据的模式，是解决大复杂度问题的一种相对简单有效的方法。[32]

6 深度学习的软件缺陷预测算法

Yang X 等人提出了使用深度信念网络来选取特征的方法 [33]。该方法首先针对软件的更改选取 14 个特征，然后对数据进行清理：归一化和重采样。首先通过最小二乘法进行归一化，从而将所有的特征调整到 [0,1] 的范围内。此外，由于在更改过程中没有缺陷的软件模块数量要远远超过有缺陷的软件模块数量，因此在采样过程中随机地删除一部分没有缺陷的模块，直到两种类型的数据数量比较接近。这种方法可以使分类器的分类结果不偏向大多数类，从而提高分类器的性能。[34] 然后，使用由多个受限玻尔兹曼机组成的深度信念网络进行特征选择。

RBM 的可见层神经元之间和隐层神经元之间假定无连接。深神经网

络用层次无监督贪婪预训练方法分层预训练 RBM，将得到的软件缺陷预测结果作为监督学习训练概率模型的初始值，学习性能得到很大改善。无监督特征学习就是将 RBM 的复杂层次结构与大量数据集之间实现统计建模。[35] 通过无监督预训练使网络获得高阶抽象特征，并且提供较好的初始权值，将权值限定在对全局训练有利的范围内，使用层与层之间的局部信息进行逐层训练，注重训练软件代码自身的特性，能够减小对学习目标过拟合的风险，并避免深神经网络中误差累积传递过长的问题。最后，作者在六个大型开源项目的数据集上对预测结果的 F1 和 cost 两项指标进行评估，结果表明，F1 值约为 45%，另外，这种方法可以通过只检查 20% 的代码就检查出超过 50% 的软件缺陷，因此大大减少了预测所需要花费的时间。[36]Dam H K 等人提出了一种抽象语法树与 LSTM 相结合的软件缺陷预测方法 [37]。这种方法的特点是将软件代码本身直接作为神经网络的输入，而不使基于软件度量的特征。首先将代码生成抽象语法树，然后通过词向量方法生成语法树节点之间的关系并对原来的语法树节点进行替换。这种方法的好处使能够捕捉代码本身的语法和不同层次间的语义结构。[38] 然后通过 LSTM 对软件缺陷特征进行学习。首先将一个 LSTM 单元建模成一个函数，这个函数接受一个 AST 节点并输出隐藏状态和它从 AST 中学习到的上下文信息。这是通过聚类来实现的。然后通过递归地调用这些函数完成对整个抽象语法树的解析。[39] 通过在三星数据集上对模型做 10 折交叉验证，并对结果进行分析，预测精度和查全率都远远超过之前表现很好的随机森林和逻辑回归的算法。[40]

7 特征选择技术

7.1 概述

随着样本维数不断增大，由此引发的维数灾难使得构建的软件缺陷预测模型的有效性难以提高，为了解决小样本、高纬度的问题，特征选择被广泛运用于软件缺陷预测研究中。[41] 特征选择可以通过对数据进行降维从而加速学习过程。

7.2 特征选择方法

按子集评估方法划分的特征选择方法主要包含三类：过滤 (Filter) 和包装 (Wrapper) 以及混合型。[42]Filter 特征选择方法与 Wrapper 特征选择方法的不同在于：Filter 特征对于选出的特征子集的评估过程独立于学习算法，主要依赖于评估函数，因此具有较好的通用性以及较低的时间复杂度。[43] 而 Wrapper 特征选择是通过学习算法得到分类性能，通过分类性能评价特征子集的优劣，主要依赖具体的学习算法对子集进行评估，花费的时间多，而且根据训练集上的预测效果选择特征容易导致过拟合的问题。[44] 混合型特征选择方法综合了过滤和包装两种方法，先用 Filter 方法过滤掉小部分特征，再利用 Wrapper 方法选择最佳特征子集。FECAR 特征选择框架基于 Filter 框架使用特征聚集和特征排列，首先将数据的特征空间分为 k 个子集，然后从中选出具有相关性的特征，在评估子集阶段，选取了信息增益、卡方检验值以及 Relief 算法作为评价函数。[45] 实验结果表明利用 FECAR 特征选择框架进行特征选择后能提高软件缺陷预测模型的预测精准率。[46] Baosheng 等人基于 Wrapper 框架提出了 SAFS 特征选择方法，SAFS 在级联问题上最大程度检测每一个子模块，随后利用随机森林算法建立了软件缺陷预测模型，利用模型的预测结果来评估特征选择算法的有效性。[47]

8 性能评价指标

在软件缺陷预测中，需要选取合理的指标来评价预测结果。[48] 其预测过程会产生四种不同的结果，包括真正例 TP、假正例 FP、假负例 FN 和真负例 TN，其中有缺陷为正例，无缺陷为负例。行表示实际类别，列表示预测类别。[49] 精确率 P(Precision) 是指被正确预测为正例数与所有被预测为正例数的比率，反映了预测模型的准确程度，也称为查准率。P=TP/(TP+FP) 召回率 R(Recall) 是指被正确预测为正例数与实际正例数的比率，反映了一个有缺陷模块被正确预测出的概率，也称为查全率。R=TP/(TP+FN) F-Measure 是信息检索领域的一个评价指标，常用的是 F1 度量，即为精确率与召回率的调和平均数。F1=2×P×R/(P+R) 真正例率 TPR(True Positive Rate) 与召回率相同，也是指被正确预测为正例数与

实际正例数的比率。[50]TPR=TP/(TP+FN) 假正例率 FPR(False Positive Rate) 是指被错误预测为正例数与实际负例数的比率。FPT=FP/(FP+TN) 接受者操作特征 (Receiver Operating Characteristic，ROC) 曲线是描述分类模型真正例率 TPR 和假正例率 FPR 之间关系的一种图形化方法 [51]，横坐标表示假正例率，纵坐标表示真正例率。对于一个特定的预测模型和训练数据集，其预测结果对应于 ROC 曲线上的一个点，通过调整该模型的阈值即可得到一条经过 (0, 0) 和 (1, 1) 的曲线，曲线下方的面积即为 AUC(Area Under the Curve) 的值 [52]。其中，AUC 的取值范围为 0 1，当 AUC 为 0.5 时，表示随机猜测模型的性能，AUC 值越大，说明该模型的性能越好 [53]。因此，好的预测模型应尽可能地靠近坐标系的左上角。

9 結論

软件缺陷的预测是一项帮助改善软件质量的活动，经过对软件缺陷的研究发现缺陷存在一定的规律，与软件的代码特性、开发过程、项目属性都有关，因此缺陷预测就是从软件的基本属性中寻找共同点，根据规律对未知的软件进行预测，期望发现其中的软件缺陷或缺陷分布。[54] 现有的软件缺陷预测模型绝大部分是基于机器学习算法，也产生了众多研究成果，尤其是在预测的度量元和预测方法上更是硕果累累。本文笔者在研究相关国内外文献的基础上，总结了基于机器学习和软件缺陷的定义、软件缺陷预测的必要性，软件缺陷预测的步骤、软件缺陷预测的关键技术、模型和算法等。

10 參考文獻

[1] 曾路, 汪浩. 基于机器学习的虚拟仪器软件缺陷预测模型研究 [J]. 自动化与仪器仪表,2020(05):59-62.

[2] 张志武. 基于机器学习的软件缺陷预测方法研究.2018.

[3]Metric-based software reliability prediction approach and its application[J] . Ying Shi,Ming Li,Steven Arndt,Carol Smidts. Empirical Software Engineering . 2017 (4)

[4]Dam H K, Pham T, Ng S W, et al. A deep tree-based model for software defect prediction[J]. arXiv preprint arXiv:1802.00921, 2018.

[5]Kamei Y, Shihab E. Defect prediction: Accomplishments and future challenges[C] Software Analysis, Evolution, and Reengineering (SANER), 2016 IEEE 23rd International Conference on. IEEE, 2016, 5: 33-45.

[6]Fu W, Menzies T. Easy over hard: A case study on deeplearning[C] Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. ACM, 2017: 49-60.

[7]Arar ? F, Ayan K. A feature dependent Naive Bayes approach and its application to the software defect prediction problem[J]. Applied Soft Computing, 2017, 59: 197-209.

[8]Alsawalqah H, Faris H, Aljarah I, et al. Hybrid SMOTE-Ensemble Approach for Software Defect Prediction[M] Software Engineering Trends and Techniques in Intelligent Systems. 2017:355-366.

[9]Tantithamthavorn C, McIntosh S, Hassan AE, et al. The impact of automated parameter optimization on defect prediction models[J]. IEEE Transactions on Software Engineering, 2018.

[10]Bennin K E, Keung J, Phannachitta P, et al. Mahakil: Diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction[J]. IEEE Transactions on Software Engineering, 2017.

[11]Chen X, Zhao Y, Wang Q, et al. MULTI: Multi-objective effort-aware just-in-time software defect prediction[J]. Information and Software Technology, 2018, 93: 1-13.

[12]Young S, Abdou T, Bener A. A replication study: just-in-time defect prediction with ensemble learning[C] Proceedings of the 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering. ACM, 2018: 42-47.

[13]Shepperd M, Bowes D, Hall T. Researcher bias: The use of machine learning in software defect prediction[J]. IEEE Transactions on Software Engineering, 2014, 40(6): 603-616.

[14]Singh M, Salaria D S. Software defect prediction tool based on neural

network[J]. International Journal of Computer Applications, 2013, 70(22).

[15]Li J, He P, Zhu J, et al. Software defect prediction via convolutional neural network[C] Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on. IEEE, 2017: 318-328.

[16]Wang J, Zhang C. Software reliability prediction using a deep learning model based on the RNN encoder–decoder[J]. Reliability Engineering & System Safety, 2018, 170: 73-82.

[17]Malhotra R. A systematic review of machine learning techniques for software fault prediction[J]. Applied Soft Computing, 2015, 27: 504-518.

[18]Yang X, Lo D, Xia X, et al. TLEL: A two-layer ensemble learning approach for just-in-time defect prediction[J]. Information and Software Technology, 2017, 87: 206-220.

[19]Jindal V. Towards an intelligent fault prediction code editor to improve software quality using deep learning[C] Conference Companion of the 2nd International Conference on Art, Science, and Engineering of Programming. ACM, 2018: 222-223.

[20]Malhotra R, Jain A. Fault prediction using statistical and machine learning methods for improving software quality[J]. Journal of Information Processing Systems, 2012, 8(2): 241-262.

[21]Song Q, Jia Z, Shepperd M, et al. A general software defect-proneness prediction framework[J]. IEEE Transactions on Software Engineering, 2011, 37(3): 356-370.

[22]Lessmann S, Baesens B, Mues C, et al. Benchmarking classification models for software defect prediction: A proposed framework and novel findings[J]. IEEE Transactions on Software Engineering, 2008, 34(4): 485-496.

[23]Shanthin A, Chandrasekaran R M. Applying machine learning for fault prediction using software metrics[J]. International Journal of Advanced Research in Computer Science and Software Engineering, 2012, 2(6): 274-284.

[24]Karim S, Warnars H L H S, Gaol F L, et al. Software metrics for fault prediction using machine learning approaches: A literature review with

PROMISE repository dataset[C] Cybernetics and Computational Intelligence (CyberneticsCom), 2017 IEEE International Conference on. IEEE, 2017: 19-23.

[25]Singh P K, Agarwal D, Gupta A. A systematic review on software defect prediction[C] Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on. IEEE, 2015: 1793-1797.

[26]Lanza M, Mocci A, Ponzanelli L. The tragedy of defect prediction,prince of empirical software engineering research[J]. IEEE Software, 2016 (6): 102-105.

[27]Gómez O S, Juristo N, Vegas S. Understanding replication of experiments in software engineering: A classification[J]. Information and Software Technology, 2014, 56(8):1033-1048.

[28]Hall T, Bowes D. The state of machine learning methodology in software fault prediction[C] Machine Learning and Applications (ICMLA), 2012 11th International Conference on. IEEE, 2012, 2: 308-313.

[29]Hall T, Beecham S, Bowes D, et al. A systematic literature review on fault prediction performance in software engineering[J]. IEEE Transactions on Software Engineering, 2012, 38(6): 1276-1304.

[30]Mahmood Z, Bowes D, Hall T, et al. Reproducibility and replicability of software defect prediction studies[J]. Information and Software Technology, 2018.

[31]Yang X, Lo D, Xia X, et al. Deep Learning for Just-in-Time Defect Prediction[C] QRS. 2015: 17-26.

[32]Nam J, Kim S. Clami: Defect prediction on unlabeled datasets (t)[C] Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on. IEEE, 2015: 452-463.

[33] 王铁建, 吴飞, 荆晓远. 基于多核字典学习的软件缺陷预测 [J]. 计算机科学, 2017, 44(12): 131-134.

[34] 张亮. 基于改进 BP 算法的软件缺陷预测模型研究 [D]. 北京理工大学, 2015.

[35] 杨晓琴. 基于改进蝙蝠算法的软件缺陷预测模型 [J/OL]. 计算机技

术与发展,2018(12):1-7[2018-07-18].

[36] 韩宏峰, 罗羿隆, 相克磊, 等. 基于机器学习的软件缺陷识别的必要性 [J]. 电脑知识与技术: 学术交流, 2017, 13(9): 185-186.

[37] 涂威威. 基于机器学习的软件缺陷预测 [D]. 南京大学, 2012.

[38] 于巧. 基于机器学习的软件缺陷预测方法研究 [D]. 中国矿业大学, 2017.

[39] 韦良芬. 基于机器学习的软件缺陷预测技术研究 [J]. 长春大学学报(自然科学版), 2017 (2017 年 05): 7-9, 13.

[40] 刘旸. 基于机器学习的软件缺陷预测研究 [J]. 计算机工程与应用, 2006, 42(28): 49-53.

[41] 程铭, 毋国庆, 袁梦霆. 基于迁移学习的软件缺陷预测 [J]. 电子学报, 2016, 44(1): 115-122.

[42] 毛发贵, 李碧雯, 沈备军. 基于实例迁移的跨项目软件缺陷预测 [J]. 计算机科学与探索, 2016 (1): 43-55.

[43] 程俊, 张雪莹, 李瑞贤. 基于元学习的软件缺陷预测推荐方法 [J]. 中国电子科学研究院学报, 2015, 10(6): 620-627.

[44] 张蕾, 朱义鑫, 徐春, 等. 基于字典学习的软件缺陷检测算法 [J]. 计算机应用, 2016, 36(9): 2486-2491.

[45] 傅艺绮, 董威, 尹良泽, 等. 基于组合机器学习算法的软件缺陷预测模型 [J]. 计算机研究与发展, 2017, 54(3): 633-641.

[46] 陈翔, 王莉萍, 顾庆, 等. 跨项目软件缺陷预测方法研究综述 [J]. 计算机学报, 2018, 41(1): 254-274.

[47] 解维奇, 蔡远文, 程龙, 等. 面向航天型号软件缺陷预测的属性选择方法 [J]. 计算机测量与控制, 2014, 22(10): 3439-3441.

[48] 杨勋姮, 段明璐. 软件缺陷分析技术的研究 [J]. 软件, 2018 (2): 93-101.

[49] 李勇, 黄志球, 王勇, 等. 数据驱动的软件缺陷预测研究综述 [J]. 电子学报, 2017 (4): 982-988.

[50] 何吉元, 孟昭鹏, 陈翔, 等. 一种半监督集成跨项目软件缺陷预测方法 [J]. 软件学报, 2017, 28(6): 1455-1473.

[51] 陈翔, 顾庆, 刘望舒, 等. 静态软件缺陷预测方法研究 [J]. Journal of

Software, 2016, 27(1).

[52] 王男帅, 薛静锋, 胡昌振, 等. 基于遗传优化支持向量机的软件缺陷预测模型 [J]. 中国科技论文, 2015 (2): 159-163.

[53] 黎铭, 霍轩. 半监督软件缺陷挖掘研究综述 [J]. 数据采集与处理, 2016 (2016 年 01): 56-64.

[54] 廖胜平. 基于半监督学习的软件缺陷预测方法研究 [D]. 重庆大学,2016.