基于大数据分析的软件缺陷检测与修复优化

梅思雨
江西机电职业技术学院　江西南昌　330000

摘　要: 在软件开发领域中,质量是决定产品成败的一个关键因素,软件缺陷是软件质量中的一大难点。在软件规模越来越大,复杂度越来越高的情况下，传统软件测试方法已经很难满足对缺陷进行高效，精确检测的要求。大数据分析的快速发展给软件缺陷检测带来一种新思路、新方法。因此，文章分析当前软件缺陷产生的原因及缺陷检测方法，探讨媳妇优化策略。

关键词： 大数据分析；软件缺陷；修复优化

在软件应用场景日益扩大的情况下，软件可靠性与稳定性就成为关键考虑。软件缺陷不但会使软件功能不正常，系统瘫痪，而且还会造成昂贵的维护成本以及用户信任危机。以往软件缺陷检测多集中于局部代码审查及简单测试策略上，很难处理软件系统不断增长的复杂性与动态性。大数据时代的到来给软件领域带来全新生机。

1 常见软件缺陷产生的原因

1.1 需求理解偏差

软件项目启动阶段的需求理解偏差很容易造成软件缺陷。开发团队在与需求方交流过程中，受语言表达限制，双方知识背景及认知差异等因素影响，难以规避需求模糊性及不确定性等问题。需求方也许不能明确地阐明业务流程和功能期望，开发团队也许会根据自己的经验来诠释需求，而在这一过程中缺少有效的验证机制从而造成对于需求关键要素的认识错位[1]。与此同时，对需求变更管理不到位也是一个不可忽视的因素，如果在项目过程中对需求动态变化没有得到及时而全面地捕获和消化，就会出现新的需求和现有认识之间的矛盾，会进一步加重需求理解偏差并最终体现在软件功能不符合实际需求的弊端上。

1.2 编码错误

在编码时开发人员会受到很多因素的影响而出错。编程语言的内在复杂性，例如语法规则和内存管理机制等，都增加编码的难度，稍不注意就可能导致语法错误和逻辑漏洞。开发人员的个体能力与经验良莠不齐，初学者对于复杂的算法，设计模式缺乏足够的把握，容易编写低质量的代码；即便是有经验的人，面对一个大型项目的复杂架构、海量代码，都会因为疏于管理而出错。另外，在团队协作开发过程中，由于代码风格的不一致，接口定义的含混不清以及缺乏高效的代码审查机制等原因，使得不同模块的代码在集成过程中存在兼容性的问题，这些编码层面上的误差积累在一起就成软件缺陷产生的一个重要根源。

2 基于大数据分析的软件缺陷检测方法

2.1 基于机器学习的检测方法

基于机器学习软件缺陷检测方法借助于大量已经标记好的软件数据构造出能够识别缺陷模式。数据预处理阶段对软件的代码特征，执行日志和其他数据进行清洗、变换，并提取出诸如代码复杂度和圈复杂度这样的有效特征作为之后建模的铺垫[2]。然后，选择适当的机器学习算法进行研究，常用的算法包括决策树，支持向量机和朴素贝叶斯。决策树是通过建立树状结构，并根据特征属性对数据进行分类的，从根节点到叶节点的路径代表不同的分类规则，这可以直观地展示缺陷分类的逻辑。支持向量机的主要目标是找到一个最佳的超平面，这样可以将不同种类的数据点进行区分，从而精确地区分出有缺陷和无缺陷的数据。在模型训练结束时，将待检测软件中的有关数据进行录入，该模型根据训练和学习得到的规律判断该软件中有无缺陷及缺陷类型，在历史数据丰富、特征明显的场景下，能有效检测出已知类型缺陷。

2.2 基于深度学习的检测方法

深度学习应用于软件缺陷检测时，借助深度神经网络较强的自动特征学习能力。以卷积神经网络（CNN）为研究对象，当处理软件代码的图像化数据或连续数据时，它可以通过卷积层的核心滑动来自动识别数据中的特定特征，池化层再将特征降维以降低计算量和保留关键信息。循环神经网

络（RNN）以及其衍生的长短时记忆网络（LSTM）特别擅长处理带有序列特性的数据，例如程序的执行路径。LSTM 采用门控机制可以有效地捕获远距离依赖关系和记忆程序状态信息的时间较长，这对于发现由于程序过程复杂而造成的缺陷是非常有效的。将深度学习模型预先训练到大规模无标注数据中，然后对软件缺陷检测任务做出微调，可以挖掘传统方法很难检测到的复杂缺陷模式，检测精度高，但是模型训练的计算量较大，对于硬件的要求也比较高。

2.3 基于关联规则挖掘的检测方法

关联规则发掘方法关注软件数据各属性间潜在的联系。由软件中代码元素，模块依赖关系以及测试用例的执行结果数据，采用 Apriori 算法搜索符合最小支持度与最小置信度要求的关联规则。比如，若发现“模块 A 调用模块 B”与“产生内存泄漏缺陷”之间存在较高置信度的关联规则，当检测到软件中有模块 A 调用模块 B 的情况时，就可预警可能出现内存泄漏问题。这种方法能够揭示出隐藏在数据之中的因果关系，并且挖掘出来的关联规则具有较强的可解释性，帮助开发人员了解缺陷存在的原因。然而关联规则可能会非常多，需要合理地设定阈值来筛选出有效的规则，而且对于数据完整性与准确性都有很高的要求，否则容易造成误判。

3 基于大数据分析的软件缺陷修复优化策略

3.1 挖掘缺陷数据关联，精准定位修复点

在软件开发生命周期的今天，缺陷数据就像一个富矿，包含着大量关键信息，其内部的关联关系也就成为精准定位修复点最关键的线索 [3]。随着软件系统规模的指数级扩张和复杂度的不断攀升，以往仅仅凭借开发人员经验和简单规则进行缺陷定位的模式在当今动辄百万行编码的大软件项目面前变得捉襟见肘。数据挖掘技术的兴起为解决这一难题带来转机，在关联规则挖掘算法的帮助下，可以深度剖析上万条缺陷报告，频繁代码变更记录和大量测试用例实现数据。以某电商平台软件项目为例，该项目业务逻辑复杂，经过挖掘后发现商品搜索模块代码经常会在指定时段更改，同时牵涉到商品分类算法的调整，很容易诱发搜索结果不准这一特定类型缺陷并表现出显著的共现关系。图数据库技术的出现，使这些联系得到更加直观和有效地利用。分别以缺陷，相关代码实体和测试用例为节点，将其所具有的联系设为边来构造缺陷关联图。通过应用如深度优先搜索（DFS）和广度优先搜索（BFS）这样的图遍历算法，我们可以在这张复杂的图形结构中迅速地找到与当前问题密切相关的代码部分和测试案例，对修复点的准确定位避免在海量代码库中的盲目搜索，大大提高修复的精度和效率，使开发人员可以有针对性地对缺陷进行修复。

3.2 优化数据分析算法，提升修复效率值

迈入大数据时代的数据分析算法，其性能好坏直接左右软件缺陷修复效率。已有数据分析算法在处理大规模，高维度缺陷数据过程中暴露出很多问题，计算资源耗费大，运行时间长是最突出的问题。以处理一个包含数百万条记录和上千个特征维度的缺陷数据集为例，传统的基于单机计算的数据分析算法可能需要数小时甚至数天的时间，这无疑严重限制缺陷修复的进度。为有效地提高修复效率而引入分布式计算框架，而 Apache Spark 就是其中之一。实现传统数据分析算法的并行化，并以决策树算法为例实现 Spark 集群环境中决策树各分支建设任务可以并行地分配给集群内多个节点同时处理。集群内各节点充分利用其计算资源优势显著减少算法总体运行时间。同时，降维算法也在提高算法效能方面起到至关重要的作用，其中主成分分析（PCA）是一种常用的降维方法。通过预处理高维度缺陷数据，PCA 可以识别和剔除数据冗余特征，最大限度地保留关键信息，同时降低数据维度。比如把原来 1000 维有缺陷的数据减少到 100 维，数据量大大减少使后续算法的处理速度明显提高，加快算法收敛过程。另外，基于深度学习的神经网络算法也为缺陷分析提供一条新途径。构建一个端到端缺陷分析模型，具有较强的自动学习能力并能挖掘大量复杂缺陷数据的深层模式及特征表示。相对于传统的机器学习算法，神经网络模型能更加有效地从大量数据中挖掘出有价值的信息，对缺陷修复给出快速准确的分析结果，从而有力地促进软件缺陷修复效率向一个新的层次发展。

3.3 构建缺陷预测模型，提前布局修复策

预先布局软件缺陷修复策略对降低修复成本和提高软件质量，起着不容忽视的作用，构造缺陷预测模型是实现这一目的的有效手段 [4]。在长期的软件开发过程中积累海量覆盖代码度量指标，开发人员信息和项目进度数据的多维历史数据。以长时间迭代更新移动应用开发项目为例，代码度量指标主要包括代码行数，圈复杂度和代码耦合度，开发人员的信息涵盖开发经验和擅长领域、代码编写风格等等，项目进度数据覆盖各阶段时间点，任务执行情况等等。在这些海

量数据资源的基础上，利用机器学习与深度学习技术建立缺陷预测模型。逻辑回归模型综合代码复杂度，代码行数和变更频率这些关键代码度量指标可以有效地预测软件模块中缺陷发生的趋势。在对一些开源项目代码库进行分析时，通过逻辑回归模型的预测，我们发现代码行数超过 5000 行且复杂度超过 10 的模块，其出现缺陷的概率比其他模块高出 30%。循环神经网络（RNN）以及其衍生的长短时记忆网络（LSTM）在处理时间序列中的缺陷数据时展现出显著的优越性，能够准确地确定缺陷出现的时间趋势和模式。在经过多年研发的企业级软件项目 LSTM 模型中，通过研究历年缺陷数据成功地预测年度具体业务高峰期到来之前，部分核心业务模块因业务功能经常调整而产生缺陷的可能性将明显增大。迁移学习的运用进一步扩大缺陷预测模型应用范围，把类似工程或者模块中训练完成的缺陷预测模型移植至当前工程，可以迅速适应新工程的特征，对缺陷做出预测。开发团队可透过这些多样化的预测模型事先准确地找出可能有瑕疵的代码区域及时段，然后合理地安排资源、制定周密的修复计划并实现软件缺陷前瞻性管理，从而有力地保证软件质量的稳定提高和开发效率的显著改善。

3.4 改进修复流程机制，保障修复高质量

确保软件缺陷修复工作的优质进行，必须有科学、合理的修复流程机制作为支持。在对软件缺陷进行实际修复时，往往会因为没有健全的流程控制以及行之有效的质量保障措施而产生一些难以解决的问题，比如修复不够彻底以及新缺陷的引入。例如在一个大型游戏软件的缺陷修复中，开发人员在未经过严格流程把控的情况下对游戏卡顿问题进行修复，虽然短期内解决卡顿现象，但是在随后的测试过程中，发现修复过程会引入画面闪烁这一新问题，这极大地影响游戏体验。为对修复流程机制进行有效完善，构建一个以工作流为核心的缺陷修复管理系统就成为一个至关重要的措施。该系统对缺陷实现缺陷从检测，上报，分发，修复，核查等全生命周期全方位，严把流程关。修复环节中引入代码审查机制并组织由多位经验丰富和技术过硬的开发人员共同组成的审查小组对修复后的代码进行详细的审核。审核内容涉及代码是否正确并保证代码逻辑准确；代码是否规范并按照工程既定代码编写规范进行编写；核对整体架构兼容性确保修复代码不损害软件整体架构稳定性。采用自动化测试技术对其进行修复后，自动引发系列回归测试，单元测试以及集成测试。回归测试的目的是确认已存在的缺陷是否得到完全地修复。单元测试主要针对修复代码中涉及的最小功能单元进行评估，而集成测试则是为检查修复代码与其他相关模块之间的协同效应，以确保没有新的缺陷被修复。在此基础上，我们结合各种质量监控指标，例如缺陷修复的成功率和修复后再次出现缺陷的可能性等，对整个修复流程进行持续的优化和调整。

4 结语

在软件产业日益繁荣的今天，大数据分析已经被深入地纳入软件缺陷检测和修复的范畴。它的多元方法和策略开拓了一条克服软件质量难题的新途径，从数据的关联挖掘到算法的优化，预测模型的构造以及修复流程的完善等方面全方位地提高软件的可靠性。它既是一次技术革新，也是软件产业向高质量发展新阶段迈进的关键力量，同时也为迎接复杂多样的软件应用场景奠定坚实保证，帮助软件对各个产业起到更加有力和稳定的支持作用。

参考文献：

[1] 赫鹏 . 大数据背景下软件开发与维护对策分析 [J]. 中国新通信 ,2024,26(24):33-35.

[2] 刘易晓 . 基于大数据分析的通信信息安全风险识别技术 [J]. 信息记录材料 ,2024,25(12):98-100.

[3] 常敏 . 基于大数据技术的静态软件缺陷检测系统设计 [J]. 现代电子技术 ,2021,44(17):37-41.8.

[4] 刘钦 , 巨彧龙 , 涂静鑫 . 基于大数据分析的变电设备缺陷数据辅助软件的研究及应用 [J]. 江西电力职业技术学院学报 ,2020,33(01):9-12.