|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 论文题目 | Fault Localization with Code Coverage Representation Learning | | | |
| Paper URL | https://ieeexplore.ieee.org/document/9402079/ | | | |
| Project URL | <https://github.com/Weifeng-Liu/deeprl4fl-2021-icse> | | | |
| 综述/背景介绍 | 发展状况 | 原因 | 意义 | 关键词（速记词汇、信息索引词汇） |
| 提出了一个statement-level和method-level的bug location模型，叫：DEEPRL4FL。对代码覆盖测试和突变测试的code coverage matrix改进，以及引入statement-dependency以及AST信息来综合学习表征代码程序，以更好的突出bug与非bug的特征。模型采用了CNN，因为将这些特征表示成matrix可以输入CNN进行学习，且CNN利用卷积核能够较好的捕获一些邻域信息特征。 | | |  |
| 假设 |  | | |  |
| 方法描述(含图) | DEEPRL4FL运用了三个表示学习的过程，即代码覆盖表示学习、语句依赖表示学习和源代码表示学习。  代码覆盖表示学习的过程是首先重排序CCmatrix，然后对测试用例的输出进行了分析，以定位 the error-exhibiting (EE) lines。【就是当程序执行测试用例崩溃时，会显示出报错行的信息；若程序没崩溃，而用测试用例测试的结果不符合预期，也会显示出导致错误结果的代码行信息。这些就是EE line】这些EEline在CCmatrix中被标记为-1。然后对其进行编码作为CNN输入。  语句依赖表示学习的过程是考虑了函数中代码语句之间的依赖关系，即程序的执行流（execution path）和数据流（data flow graph），然后对其进行编码作为CNN输入。  源代码表示学习的过程是对于源代码的编码，使用long path来表示代码的结构信息，使用了code2vec和ASTNN的方法来提取的，通过ASTNN得到了source code的编码表示，作为CNN的输入。 | | |  |
| 实验设计 | 1. 研究问题 2. 语句级别缺陷定位比较 3. 方法级别缺陷定位比较 4. 不同矩阵增强技术的影响分析 5. 不同表征学习的影响分析 6. 跨项目分析 7. 对C代码的性能 8. 数据集   Defects4J V1.2.0（java）,ManyBugs（C）   1. 评估指标   Recall at Top-K、Mean Average Rank (MAR)、Mean First Rank (MFR) | | |  |
| 数据处理 | 输入 | 筛除特例 | 处理方式 | 关键词（速记词汇、信息索引词汇） |
|  |  |  |  |
| 结论 | 提出了一种基于深度学习的缺陷定位（FL）方法，即DEEPRL4FL，以改进现有的FL方法。其关键思想包括：（1）将FL问题视为图像识别；（2）通过对语句和失败的测试用例之间的关系进行建模来增强代码覆盖率矩阵；（3）将代码覆盖率表示学习与语句依赖性相结合，并对通常的可疑代码进行代码表示学习。我们的实证评估表明，我们的模型比最先进的基线方法更有优势。 | | | |
| 局限性分析 |  | | | |

2.论文总结：

**（论文名）：**用代码覆盖率表示学习进行缺陷定位

**（题目）**：Fault Localization with Code Coverage Representation Learning

**（论文URL）：**https://ieeexplore.ieee.org/document/9402079/

**（项目URL）：**<https://github.com/Weifeng-Liu/deeprl4fl-2021-icse>

**（总结）：**

本文提出了DEEPRL4FL，它是一种基于深度学习的缺陷定位（FL）方法。之前的缺陷定位方法大多数都是利用了不同的计算公式来计算出不同代码行的缺陷得分会带来一些问题，对此前的方法提出了一些改进，第一将FL问题视为图像识别；第二通过对语句和失败的测试用例之间的关系进行建模来增强代码覆盖率矩阵；第三将代码覆盖率表示学习与语句依赖性相结合，并对通常的可疑代码进行代码表示学习。通过实验与其他方法在语句级别、方法级别对比，并通过消融实验、跨项目和跨语言评估项目，我们的模型比最先进的基线方法更有优势。

**（附图）：**

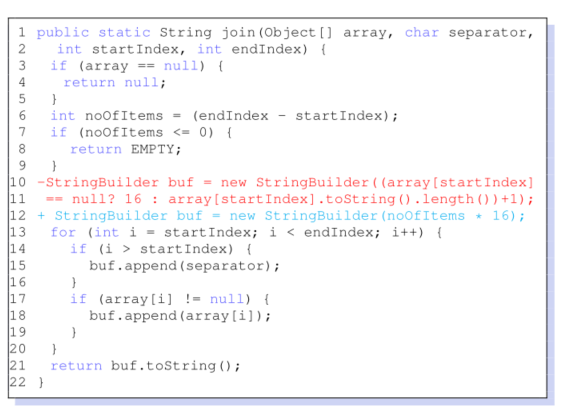


图1：一个有缺陷的声明的例子

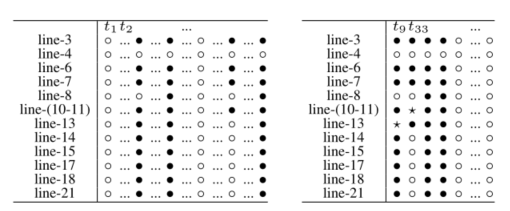


图2：图1的代码覆盖率（注： 表示0,1,-1）

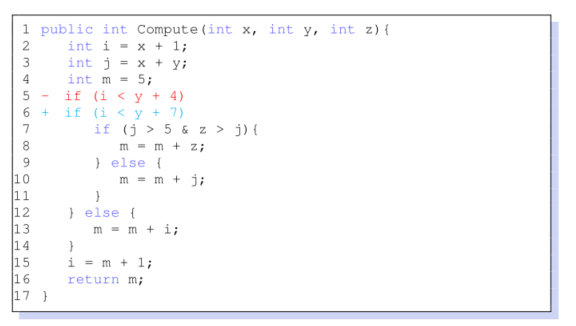


图3：一个有问题的语句和相互依赖的语句

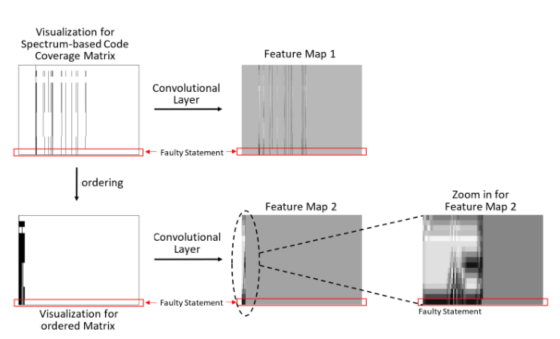


图4：测试案例排序后的特征图

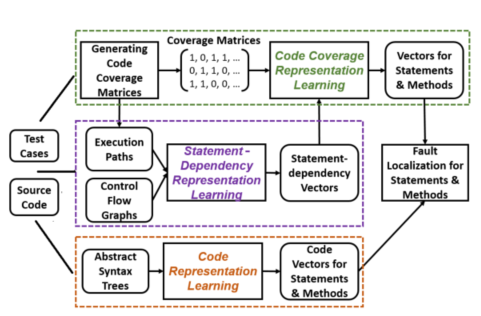


图5：DEEPRL4FL的架构

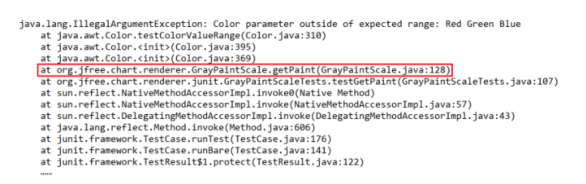
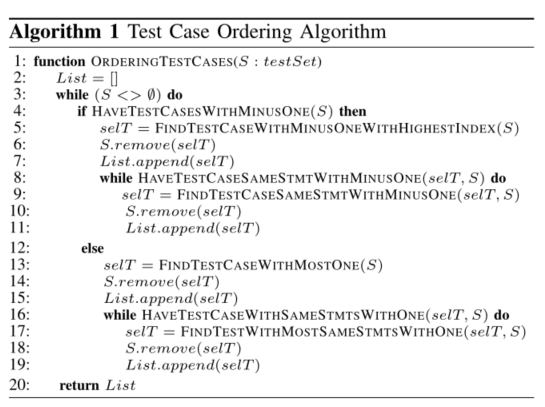
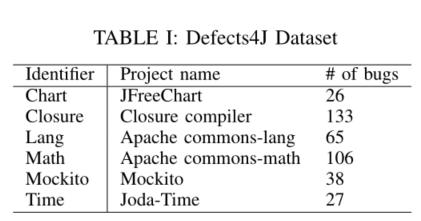
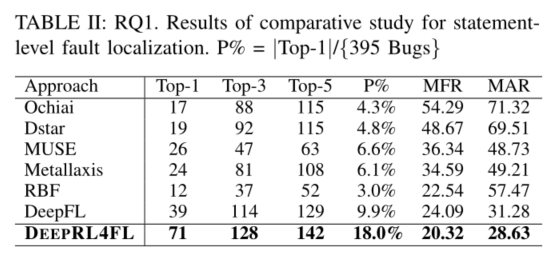


图 6: 错误信息示例







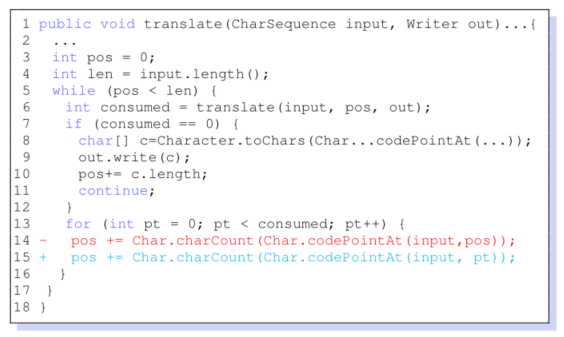
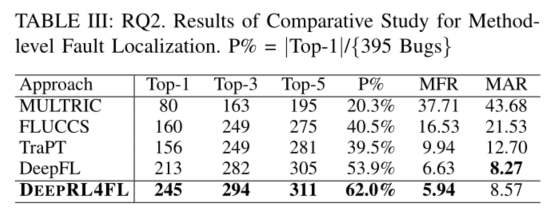


图7：来自Defects4J的一个例子



图8：由CNN为图7制作的特征图。



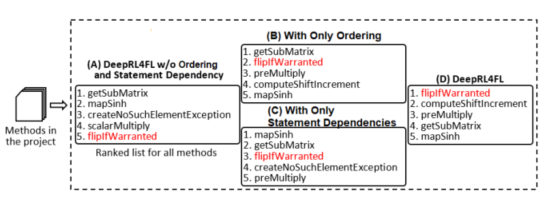
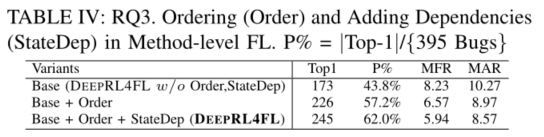


图9：排序和语句的依赖性影响排名



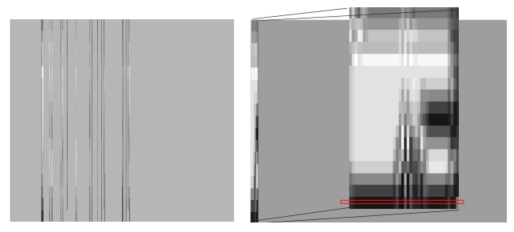


图 10: bug语句周围的视觉上的暗线

