基于漏洞检测的深度学习模型研究：

An Empirical Study of Deep Learning Models for Vulnerability

论文主要研究内容

取得的成果

创新点

自己对论文的思考

自己的改进意见

摘要：

1.1研究背景：

· 深度学习（DL）模型在代码漏洞检测方面取得了显著进展。

· 与静态分析工具相比，DL模型在某些情况下表现更优。

· 尽管提出了许多优秀的模型，但对这些模型的理解还不够深入，限制了模型的鲁棒性、调试和部署的进一步提升。

静态分析工具：

依赖预定义的规则、模式和编码标准来检测代码中的问题，不执行代码

DL模型：

通过训练数据中的示例来学习识别漏洞的特征

1.2研究目的：

· 论文重现并评估了9种最新的深度学习模型在两个常用漏洞检测数据集（Devign和MSR）上的表现。

· 通过6个研究问题（RQ）调查模型能力、训练数据和模型解释

1.3研究内容：

1. 在常用的故障检测数据集Devign和MSR上研究并复现了9种最流行的DL模型。通过6个研究了模型能力，训练数据和模型解释性三个方面；

2. 通过实验证明了模型的不同运行与不同模型输出之间的低一致性之间的可变性。

3. 研究比较了针对特定类型的漏洞训练的模型和一次所有漏洞训练的模型。

4. 探索了 DL 可能认为难处理的程序类型。

5. 研究了训练数据大小和训练数据组合与模型性能的关系。

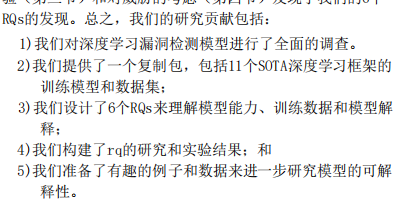
6. 研究了模型解释并分析了用于进行预测的重要特征。发现可以帮助更好地理解模型结果，为准备训练数据提供指导，提高模型的鲁棒性。

1.4研究结果与贡献：

· 提供了对模型结果的更好理解。

· 为准备训练数据提供了指导。

· 改进模型的鲁棒性。



Part1.漏洞检测：

QR1:模型对漏洞的检测结果是否一致？模型在不同运行和不同模型之间的变量是什么

不一致

QR2:不同类型的漏洞检测难易程度是否相同？要建立通用还是专用的检测模型

QR3：是否存在难以被当前模型准确预测的代码程序？有的话这些代码特性是？

Part2.训练数据：

QR4:增加数据集大小是否有助于模型检测

QR5:训练集中数据构成对模型的影响

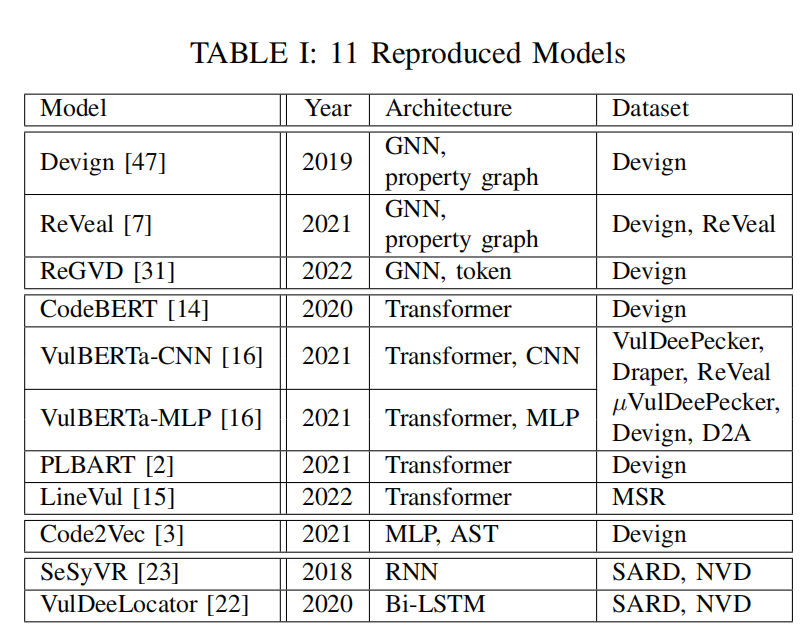
Part3.模型解释性：

QR6：用于预测的模型中有什么源代码信息？这些模型对重要特征是否一致？

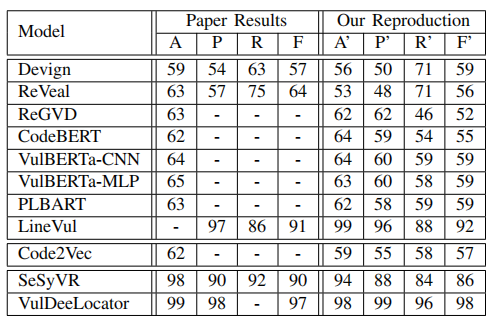
研究方法：

调查了SOTA的深度学习模型，在原始数据集上复刻了11个模型；

选取9个模型在MSR和Devign两个流行数据集上进行对比评估；



复现的11个模型



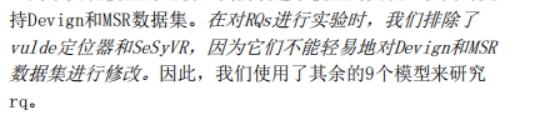
复现结果

A 准确性

P 精度

R 查全率

F F1



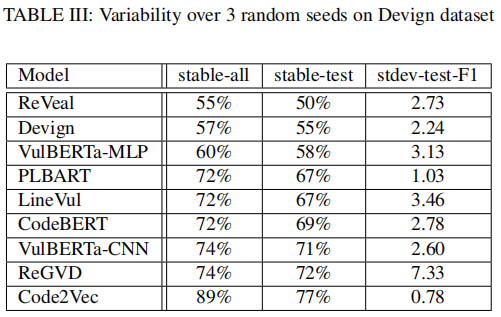
选取的数据集是Devign和MSR原因有三：1.两个数据集都包含真实项目和缺陷；2.本文中大多数模型都用Devign数据集来评估和调整；3.MSR数据集包含 310 个项目，其数据具有漏洞类型的注释，这是研究RQ 所必需的。

QR1：模型对漏洞的检测结果是否一致？模型在不同运行和不同模型之间的变量是什么

Motivation：使用不同随机种子时，深度学习模型的性能会有所不同

在Devign数据集的相同训练/有效/测试集上使用3个不同的随机种子来训练模型

在 全稳定 和 测试集稳定 的测试下稳定输入的百分比

结果：

结果显示：

平均34.9%的测试数据（30.6%的总数据），对于训练中使用不同的种子有不同的预测结果；

而F1测试分数变化不大，即对于大多数模型来说，对于不同的种子，95%的性能会和平均性能有低于5.8%的误差

只有7%的测试数据(和7%的总数据)被所有模型同意。3个GNN模型在20%的测试示例(和25%的总数)上达成一致，而3个表现最好的Transformer模型(LineVul, PLBART和VulBERTa-CNN)在34%的测试数据(和44%的总数)上达成一致。但是，当我们比较所有5Transformer模型，只有22%的测试实例(总共29%)是一致的。

****不同模型之间的低一致性意味着，当没有基础真值标签时，作为常用方法的跨模型比较的差分测试方法可能用途有限。****

****QR2：****不同类型的漏洞检测难易程度是否相同？要建立通用还是专用的检测模型

****Motivation：传统的软件保护技术中常使用不同的算法来检测不同的漏洞****

****使用MSR数据集 用CWE对示例进行注释，将漏洞分为5类：****

****缓冲区溢出/值错误/资源错误/输入验证错误/权限错误****

****8:1:1 训练 有效 测试集****



结论：

不同模型对于哪种漏洞类型最容易检测的看法并不总是一致，不同漏洞类型在不同模型中表现出不同的F1得分。

输入验证和资源错误通常在不同模型中表现较差，这可能与这些漏洞类型的复杂性有关，因为它们需要考虑变量值和循环等因素。

缓冲区溢出和值错误通常相对容易检测，这可能与它们的特性有关，这些漏洞类型在程序中的表现更加明显。

使用GNN架构的模型（Devign和ReVeal）在性能上表现相似，这可能是因为它们在属性图上进行了相似性建模。

对于资源错误（Resource errors），模型的性能相对较低，这可能是因为这些错误在代码中的位置分散，长距离依赖关系难以捕捉，或者训练数据不足以识别它们的模式。

综合模型通常在性能上不如特定漏洞类型的模型，但在某些情况下，综合模型可以表现更好，尤其是在处理输入验证和资源错误这两种漏洞类型时。

在对跨漏洞类型的性能进行分析时，研究发现在大多数情况下，不同漏洞类型之间的检测性能较差，除了 LineVul。这表明不同漏洞类型对应不同的数据分布，因此在检测与训练数据分布不同的漏洞时性能较低。特别值得注意的是，LineVul在处理值错误方面表现出色。当将训练于其他漏洞类型的模型应用于值错误的检测时，通常表现出比相同漏洞类型性能更高的结果。这表明LineVul在跨不同漏洞类型中有效地捕获和识别值错误，即使这些模型并未专门为值错误进行训练。

QR3：是否存在难以被当前模型准确预测的代码程序？有的话这些代码特性是？

创新点：

1. 选用Devign和MSR两个数据集

(1).两个数据集都包含真实项目和缺陷；(2).本文中大多数模型都用Devign数据集来评估和调整；(3).MSR数据集包含 310 个项目，其数据具有漏洞类型的注释，这是研究RQ 所必需的。

1. 首次对模型难以预测的程序和代码特征进行研究