



# 计算机科学

COMPUTER SCIENCE

## 基于异构图和指令序列的智能合约字节码漏洞检测方法

宋建华, 曹凯, 张龔

引用本文

宋建华, 曹凯, 张龔. 基于异构图和指令序列的智能合约字节码漏洞检测方法[J]. 计算机科学, 2025, 52(12): 367-373.

SONG Jianhua, CAO Kai, ZHANG Yan. Smart Contract Bytecode Vulnerability Detection Method Based on Heterogeneous Graphs and Instruction Sequences [J]. Computer Science, 2025, 52(12): 367-373.

---

## 相似文章推荐 (请使用火狐或 IE 浏览器查看文章)

Similar articles recommended (Please use Firefox or IE to view the article)

### [基于改进ModernTCN的光伏发电中短期预测](#)

Prediction of Short-and-Medium Term Photovoltaic Power Generation Based on Improved ModernTCN  
计算机科学, 2025, 52(11A): 241000164-7. <https://doi.org/10.11896/jsjcx.241000164>

### [基于知识图谱嵌入的异构图欺诈用户检测](#)

Fraud User Detection Based on Heterogeneous Information Network with Knowledge Graph  
Embedding

计算机科学, 2025, 52(11A): 250400085-7. <https://doi.org/10.11896/jsjcx.250400085>

### [基于消除语义特征的图像篡改定位模型对抗攻击](#)

Attacking Image Manipulation Localization Model by Eliminating Semantic Features  
计算机科学, 2025, 52(11A): 241100104-7. <https://doi.org/10.11896/jsjcx.241100104>

### [自动化软件缺陷定位技术研究](#)

Advances in Automatic Software Defect Location Techniques

计算机科学, 2025, 52(11A): 250200024-14. <https://doi.org/10.11896/jsjcx.250200024>

### [层次时间序列预测方法与应用综述](#)

Comprehensive Review of Hierarchical Time Series Forecasting Methods and Applications

计算机科学, 2025, 52(11A): 241000139-7. <https://doi.org/10.11896/jsjcx.241000139>

# 基于异构图和指令序列的智能合约字节码漏洞检测方法

宋建华<sup>1,3,4</sup> 曹凯<sup>2</sup> 张龔<sup>2,3</sup>

1 湖北大学网络空间安全学院 武汉 430062

2 湖北大学计算机与信息工程学院 武汉 430062

3 智能感知系统与安全教育部重点实验室 武汉 430062

4 智能网联汽车网络安全湖北省工程研究中心 武汉 430062

(sjhhu@126.com)

**摘要** 近年来,智能合约安全问题日益突出,漏洞检测成为关键挑战。在源代码不公开的情况下,字节码检测方法备受关注。然而,现有深度学习方法通常仅基于序列或图结构,难以全面捕捉漏洞特征。为此,提出一种基于异构图和指令序列的智能合约字节码漏洞检测方法 RGCN-ResNet1D(Relational Graph Convolutional Network and ResNet-based 1D Convolutional Network)。该方法将字节码建模为异构图和指令序列,分别利用关系图卷积网络(RGCN)提取结构特征和基于 ResNet 的一维卷积网络(ResNet1D)提取序列特征,并融合两类特征进行漏洞检测。同时,设计了一种基于误分类样本数量动态调整权重的交叉熵损失函数,有效缓解训练集类别不平衡问题。实验结果表明,RGCN-ResNet1D 在检测整数溢出、时间戳依赖和自毁 3 类漏洞时,F1 得分分别为 95.43%,90.67%和 92.31%,显著优于对比方法。

**关键词:** 智能合约字节码;漏洞检测;异构图;字节码指令序列;深度学习

中图分类号 TP309

## Smart Contract Bytecode Vulnerability Detection Method Based on Heterogeneous Graphs and Instruction Sequences

SONG Jianhua<sup>1,3,4</sup>, CAO Kai<sup>2</sup> and ZHANG Yan<sup>2,3</sup>

1 School of Cyber Science and Technology, Hubei University, Wuhan 430062, China

2 School of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China

3 Key Laboratory of Intelligent Sensing System and Security, Ministry of Education, Wuhan 430062, China

4 Hubei Engineering Research Center of Cyber Security for Intelligent Connected Vehicles, Wuhan 430062, China

**Abstract** In recent years, the security issues of smart contracts have become increasingly prominent, and vulnerability detection has become a key challenge. In scenarios where source code is not publicly available, bytecode-based detection methods have attracted significant attention. However, existing deep learning methods typically rely solely on sequences or graph structures, which makes it difficult to fully capture vulnerability features. To address this, this paper proposes a smart contract bytecode vulnerability detection method based on heterogeneous graphs and instruction sequences, called RGCN-ResNet1D(Relational Graph Convolutional Network and ResNet-based 1D Convolutional Network). This method models bytecode as a heterogeneous graph and instruction sequence, using a Relational Graph Convolutional Network(RGCN) to extract structural features and a ResNet-based 1D Convolutional Network(ResNet1D) to extract sequential features, and then fuses the two types of features for vulnerability detection. A cross-entropy loss function is also designed, which dynamically adjusts the weight based on the number of misclassified samples, effectively alleviating the class imbalance problem in the training set. Experimental results show that RGCN-ResNet1D achieves F1 scores of 95.43%, 90.67%, and 92.31% for detecting integer overflow, timestamp dependency, and self-

到稿日期:2024-11-13 返修日期:2025-02-22

基金项目:国家自然科学基金(62377009);湖北省重大攻关项目(JD)(2023BAA018);湖北省重点研发计划重点项目(2021BAA184, 2021BAA188);湖北省高等学校人文社会科学重点研究基地绩效评价信息管理研究中心课题(2020JX01);湖北省科技计划重大科技专项(2024BAA008)

This work was supported by the National Natural Science Foundation of China(62377009), Major Project of Hubei Province(JD)(2023BAA018), Key Project of Hubei Provincial Key R & D Program(2021BAA184, 2021BAA188), Research Center for Performance Evaluation and Information Management of Key Research Bases for Humanities and Social Sciences in Hubei Provincial Colleges and Universities(2020JX01) and Major Science and Technology Special Project of Hubei Science and Technology Plan(2024BAA008).

通信作者:张龔(zhangyan@hubu.edu.cn)

destruct vulnerabilities, respectively, significantly outperforming the comparison methods.

**Keywords** Smart contracts bytecode, Vulnerability detection, Heterogeneous graph, Bytecode instruction sequence, Deep learning

## 1 引言

随着区块链技术的迅速发展,智能合约逐渐成为区块链生态体系中的关键部分。作为一种在去中心化网络中自动执行协议条款的编程合约,智能合约的数字化特征不仅减少了中介过程,还大幅提高了交易效率。智能合约的普及应用促进了去中心化应用程序(DApps)的快速扩展,极大地丰富了区块链的应用场景。然而,智能合约的安全性问题日渐突出,成为阻碍其进一步发展的重要难题。

近年来,由智能合约漏洞引发的安全事件屡见不鲜,严重威胁到区块链生态系统的稳定性以及用户数字资产的安全。由于智能合约涉及大量数字资产且其执行过程不可逆转,代码中的漏洞极易被攻击者利用,可能导致资产损失,甚至带来灾难性的后果。例如,2016年发生的DAO事件中,攻击者利用重入漏洞非法获取了大约360万个以太币,当时市值接近5000万美元<sup>[1]</sup>;随后在2017年6月,攻击者通过Parity钱包的库合约漏洞,盗取了价值约3000万美元的以太币<sup>[2]</sup>;同年11月,在Parity钱包的新版本中再次出现漏洞,致使约1.5亿美元的以太币被永久冻结<sup>[3]</sup>;2021年8月,又有黑客利用系统漏洞窃取了大约6000万美元的加密货币<sup>[4]</sup>。这些事件不仅带来了巨大的经济损失,同时也极大地削弱了公众对区块链技术的信任。因此,如何有效检测智能合约漏洞,已成为智能合约领域的核心研究问题。

智能合约频繁出现漏洞的主要原因在于其独特的执行环境和编程语言的限制。具体来说,智能合约大多使用Solidity等专用编程语言,这类语言在语法和运行环境与传统软件有较大差异。一旦智能合约被部署到去中心化的区块链网络上,其代码将无法更改。因此,任何漏洞都可能带来不可逆的后果。此外,智能合约通常仅提供字节码供分析,源代码往往不公开或难以获取,这增加了分析和理解的难度。相较于源代码,字节码的可读性和可分析性较差,进一步提高了漏洞检测的难度和挑战性。尽管研究人员已提出多种基于字节码的智能合约漏洞检测方法,但是现有的漏洞检测方法多依赖单一的字节码表示方式(如序列或图结构),难以全面捕捉漏洞特征,因此在面对复杂漏洞或类别不平衡的场景时效果不佳。

为了解决上述问题,本文提出了一种基于异构图与指令序列的智能合约字节码漏洞检测方法(Relational Graph Convolutional Network and ResNet-based 1D Convolutional Network, RGCN-ResNet1D)。该方法充分融合了智能合约字节码的结构特征与序列特征,从而实现了高效且准确的漏洞检测。此外,本文设计了一种基于误分类样本数量动态调整权重的交叉熵损失函数,有效缓解了训练集中的类别不平衡问题,进一步提升了RGCN-ResNet1D方法的漏洞检测能力。在整数溢出、时间戳依赖和自毁3种常见漏洞类型的检测中,该方法展现出了较高的性能。

本文的主要贡献如下:

1)提出了一种结合异构图与字节码指令序列的漏洞检测

方法。通过构建包含控制流和数据流信息的异构图,利用关系图卷积网络(RGCN)提取结构特征;同时,使用ResNet1D网络提取字节码指令序列的特征;最终将图特征与指令序列特征进行融合,以实现准确的漏洞检测。

2)设计了动态调整权重的损失函数。针对数据集类别不平衡的问题,设计了一种基于误分类样本数量动态调整权重的交叉熵损失函数,有效提高了RGCN-ResNet1D方法的漏洞检测性能。

3)验证了方法的有效性。在真实智能合约数据集上的实验表明,所提方法在检测整数溢出、时间戳依赖和自毁漏洞时,F1得分分别达到了95.43%,90.67%和92.31%,显著优于对比方法。

## 2 相关工作

针对智能合约字节码的漏洞检测,国内外研究者开展了广泛的探索,并提出了多种不同的检测方法。当前主要有五大类方法,包括形式化验证法、符号执行法、模糊测试法、中间表示法以及深度学习法<sup>[5]</sup>。

**形式化验证法:**通过数学推导和形式规则,确保智能合约满足特定的安全要求。例如,KEVM<sup>[6]</sup>构建了EVM字节码的可执行规范,为程序分析和验证提供支持;Isabelle/HOL<sup>[7]</sup>将字节码分解为基本单元,利用逻辑推理进行验证。然而,这类方法对复杂漏洞的覆盖能力有限,难以识别所有潜在隐患。

**符号执行法:**这一静态分析技术将程序输入符号化,并探索执行路径以识别漏洞。例如,Oyente<sup>[8]</sup>在控制流图上执行符号化分析,检测时间依赖和重入问题;Mythril<sup>[9]</sup>结合符号执行与污点分析,支持多种漏洞类型的检测。然而,符号执行易受路径爆炸问题限制,对复杂合约的分析开销较大。

**模糊测试法:**通过生成随机或变异输入数据,以发现潜在漏洞。例如,ContractFuzzer<sup>[10]</sup>基于ABI规范自动生成测试用例,并分析日志发现漏洞,如重入攻击和异常处理问题。尽管模糊测试灵活性较高,但其误报率较高,且对复杂逻辑漏洞的检测能力有限。

**中间表示法:**通过将智能合约字节码转换为中间表示,以辅助分析。例如,Ethir<sup>[11]</sup>利用控制流图生成基于规则的中间表示,提高分析效率和一致性。然而,中间表示的转换可能丢失部分语义信息,影响检测的准确性。

**深度学习法:**深度学习通过对大量智能合约样本的训练,能够自动化识别漏洞特征,大幅提升检测的准确性和效率。例如,Tann等<sup>[12]</sup>提出基于LSTM的深度模型,通过学习字节码序列检测漏洞;Hu等<sup>[13]</sup>结合GRU与N-gram特征构建漏洞检测模型;Gu等<sup>[14]</sup>提出基于CNN-BiLSTM的模型,从操作码序列中提取漏洞特征以检测未知漏洞。然而,这类方法通常难以保留智能合约的完整结构和逻辑信息。一些研究者进一步尝试引入图结构和多模态以提升检测效果。例如,Wang等<sup>[15]</sup>利用控制流图和图神经网络对字节码进行分析,但图结构的完整性仍有提升空间;Zhang等<sup>[16]</sup>基于异构语义

图和预训练技术,利用合约语法树构建融合控制流和数据流的异构语义图,显著提升了漏洞检测效果,但其方法主要依赖源代码,难以直接应用于字节码场景;Duy 等<sup>[17]</sup>提出的 VulnSense 框架结合操作码序列、源代码和控制流图 3 种特征,采用多模态学习方式,将 BERT, BiLSTM 和 GNN 模型相结合,提升了检测性能。然而,该方法同样依赖源代码,在仅有字节码的场景下,其表现受限。

针对这些不足,本文提出了一种基于异构图和指令序列的智能合约字节码漏洞检测方法。该方法直接分析字节码,保留其图结构特征,并通过字节码指令序列增强漏洞特征的表达能力。与传统方法不同,本文不依赖源代码,而是综合多种字节码特征,旨在提升漏洞检测效果。

### 3 基于异构图和指令序列的智能合约字节码漏洞检测方法

如图 1 所示, RGCN-ResNet1D 方法由特征构建、特征提取、动态权重交叉熵损失函数以及特征融合与分类 4 个阶段组成。本文的创新性主要体现在异构图与字节码指令序列的

结合及动态权重交叉熵损失函数的设计。

该方法以智能合约的字节码为输入,通过以下流程完成漏洞检测。

1) 特征构建:首先解析智能合约字节码,构建包含控制流和数据流的异构图,同时反汇编字节码生成指令序列。此过程为后续的特征提取提供了必要的输入。

2) 特征提取:采用关系图卷积网络(RGCN)从异构图中提取结构特征,同时采用 ResNet1D 网络从字节码指令序列中提取序列特征。这两种特征为后续的漏洞检测提供了更加全面且丰富的特征表示。

3) 动态权重交叉熵损失函数设计:设计了一种基于误分类样本数量动态调整权重的交叉熵损失函数,有效缓解了训练集中类别不平衡问题,提升了模型在分类阶段中对少数类别漏洞的检测性能。

4) 特征融合与分类:将从异构图和指令序列中提取的结构特征与序列特征融合,构建联合特征表示,并输入全连接层进行二分类任务,最终输出智能合约字节码的漏洞检测结果。

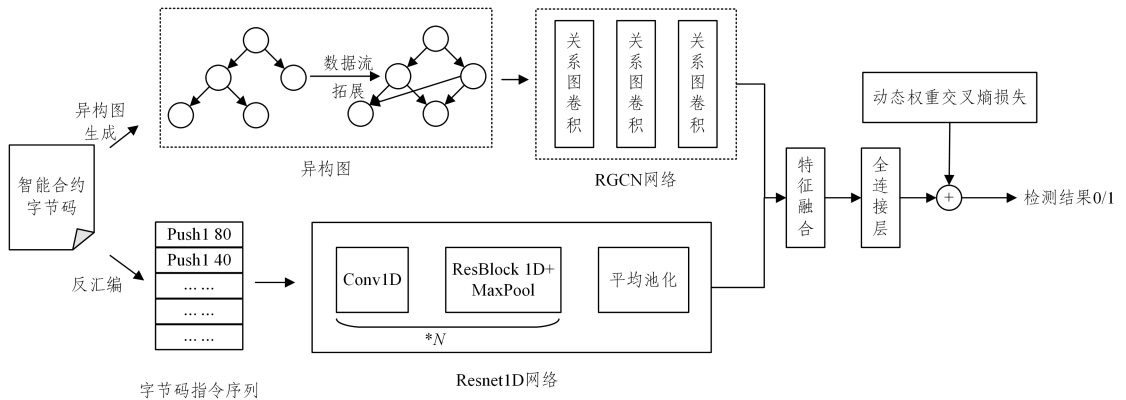


图 1 本文方法整体框架图

Fig. 1 Overall framework diagram of the proposed method

#### 3.1 异构图构建

先前的研究表明,基于控制流的智能合约字节码漏洞检测方法效果良好。在此基础上,本文采用 evm\_cfg\_builder 工具解析智能合约字节码,以构建控制流图。在控制流图中,每个节点对应一个基本块,代表一系列按顺序执行的指令。基本块开始于入口指令,通常为程序的起始点或跳转目标,并终止于跳转指令或结束指令(如 JUMP, JUMPI, STOP 等)。通过分析基本块中的跳转指令,可以将各个基本块进行连接,从而构建出完整的控制流图。在此图中,节点代表基本块,边表示基本块间的跳转关系。

然而,控制流图仅能展示智能合约的执行顺序和条件分支,无法体现合约中的数据传递和依赖关系,在检测依赖数据关系的漏洞(如重入攻击和未初始化变量)时,存在一定的局限性。因此,本文在控制流图的基础上,进一步进行基于基本块的数据流分析,以揭示数据依赖关系。

具体而言,本文首先对每个基本块的指令进行分析,以提取内部数据依赖信息。为此,构建了一个数据依赖字典,包含 read 和 write 两个集合,分别记录基本块中被读取和写入的指令。在分析过程中,通过识别指令特性,例如将 PUSH 指

令作为写操作,将 POP 和算术指令(如 ADD 和 SUB)视为读操作,动态更新字典中的 read 和 write 集合,从而完整记录基本块的内部数据依赖。接着,在此基础上构建基本块之间的数据流关系。遍历每个基本块及其流向的下一个基本块,检查源基本块的写入集合是否与目标基本块的读取集合存在交集。如果有交集,说明源基本块中的数据在目标基本块中被读取,进而在两者之间建立数据流边,以表示数据依赖关系。

通过上述步骤构建的异构图不仅涵盖了控制流信息,还揭示了智能合约内部的数据传递和依赖关系。

为了更加清晰地展示这一过程,本文将包含控制流和数据流异构图的流程以算法形式展现,具体如算法 1 所示。

#### 算法 1 构建包含控制流和数据流的异构图

输入:字节码 B

输出:包含控制流和数据流的异构图 G

1. /\* 控制流和数据流图的构建过程 \*/
2. 初始化:  $E_c \leftarrow \emptyset$ ,  $E_d \leftarrow \emptyset$  /\* 初始化控制流边和数据流边集合 \*/;
3. 使用 evm\_cfg\_builder 从字节码 B 构建控制流图 CFG;
4. 提取控制流图中的基本块集合  $\{B_1, B_2, \dots, B_n\}$ ;

5. for  $i \leftarrow 1$  to  $n$  do
6. /\* 分析基本块的控制流关系 \*/
7. 记录基本块  $B_i$  到后继基本块  $B_j$  的控制流边, 加入  $E_c$ ;
8. /\* 分析基本块的读写依赖关系 \*/
9. 如果  $B_i$  的写入数据在  $B_j$  中被读取, 则记录  $B_i$  到  $B_j$  的数据流边, 加入  $E_d$ ;
10. end for
11. 构建异构图  $G$ ; 节点为  $\{B_1, B_2, \dots, B_n\}$ , 边为  $\{E_c, E_d\}$ ;
12. 输出  $G$ .

### 3.2 异构图特征提取

构建异构图后, 使用 Word2Vec 模型对异构图中每个节点的字节码指令集进行词嵌入处理。然后, 分别计算这些嵌入的平均值、最大值和总和值, 并将计算结果拼接成一个聚合嵌入向量, 作为节点的特征表示, 如图 2 所示。

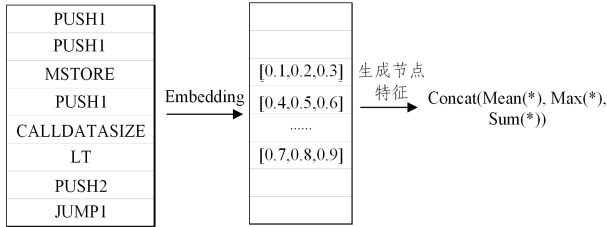


图 2 异构图节点嵌入

Fig. 2 Heterogeneous graph node embedding

完成节点嵌入后, 异构图被定义为  $G=(V, E, R)$ , 其中  $V$  表示节点集合,  $E$  表示边集合,  $R$  表示关系集合。关系集  $r \in R$  包含图中不同类型的边, 包括数据流边和控制流边。

为了高效提取异构图的特征信息, 本文设计了一个包含 3 层关系图卷积层 (RGCNConv) 的关系图卷积网络 (RGCN), 旨在捕捉异构图的复杂结构特征。RGCNConv 通过对不同关系类型分别进行特征传播与聚合操作, 以充分挖掘图结构中的信息。其特征传播公式如下:

$$h_i^{(l+1)} = \sigma \left( \sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{i,r}} \mathbf{W}_r^{(l)} h_j^{(l)} + \mathbf{W}_0^{(l)} h_i^{(l)} \right) \quad (1)$$

其中,  $h_i^l$  表示第  $l$  层中节点  $i$  的特征;  $\mathbf{W}_0^{(l)}$  和  $\mathbf{W}_r^{(l)}$  分别表示节点的自特征权重矩阵和特定关系类型  $r$  的权重矩阵; 邻接节点集合  $N_i^r$  代表与节点  $i$  通过关系  $r$  相连的邻居节点;  $c_{i,r}$  则是用于归一化的系数;  $\sigma$  是激活函数。式 (1) 描述了节点在不同关系类型的邻居影响下的特征更新过程。

### 3.3 字节码指令序列特征提取

为了解决 3.2 节中异构图节点嵌入可能导致的字节码指令顺序信息丢失问题, 并增强对漏洞特征的提取能力, 本文在异构图特征提取的基础上, 额外引入了基于字节码指令序列特征提取模块。首先, 利用 evmdasm 库对智能合约字节码进行反汇编, 提取出完整的指令序列。接着, 使用预训练的 Word2Vec 模型生成每条指令的嵌入向量。具体来说, 对于每条指令的操作码, 检查其是否出现在模型的词汇表中; 若存在, 则直接获取对应的嵌入向量; 若不存在, 则分配一个零向量, 以增强模型的鲁棒性。对于每条指令的操作数, 进行丢失处理, 以去除可能带来的噪声干扰。最终, 将所有指令的嵌入向量整合成一个二维数组, 每一行对应一条指令的嵌入向量, 列数则为嵌入向量的维度。这一过程如图 3 所示。这种基于

指令序列的细粒度表示, 不仅能够有效保留字节码指令的顺序信息, 还能为漏洞检测提供更加丰富的特征, 从而提升检测效果。

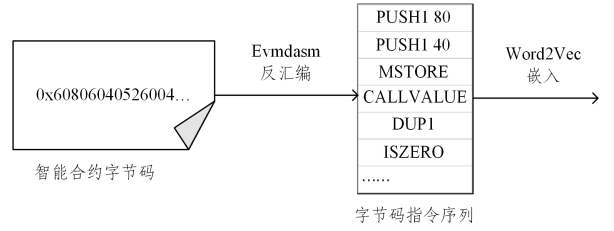


图 3 字节码指令序列嵌入

Fig. 3 Bytecode instruction sequence embedding

在此基础上, 如何有效提取指令序列的特征成为关键问题。本文参考了 Rossini 等<sup>[18]</sup>的研究, 该研究显示, 在智能合约漏洞分类的准确性方面, 一维 ResNet 的卷积神经网络 (CNN) 优于其他模型, 例如一维或二维 CNN 和长短期记忆网络 (LSTM)。因此, 本文采用基于 ResNet 的一维卷积网络 (ResNet1D) 对字节码指令序列进行特征提取。同时, 为了更好地满足双通道漏洞检测任务的需求, 对该网络结构进行了适当的修改和优化, 具体改进如下。

1) 将通道数逐步从 16 增加至 256, 随后逐步减少至 64, 以提升特征表示的丰富性和模型的表达能力。

2) 对池化层进行了调整, 通过减少池化层的数量并减小步长, 以保留更多的时间步信息, 增强细节特征的捕捉能力。

ResNet1D 网络结构由一维卷积层 (Conv1D) 和残差块 (ResBlock1D) 交替组成。卷积层用于调整通道数量, 而残差块通过跳跃连接保留关键特征信息, 有效缓解梯度消失问题, 提升模型的稳定性和泛化能力。在 ResNet1D 网络中, 通道数从 16 逐步扩展到 256, 最后缩减至 64。每次通道数的变化由一维卷积层实现, 随后连接残差块。卷积层与残差块的组合确保了模型的深度、稳定性和特征提取能力。图 4 展示了 ResBlock1D 的结构。

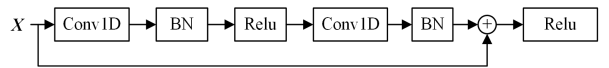


图 4 ResBlock1D 结构

Fig. 4 Structure of ResBlock1D

### 3.4 不平衡数据集处理

使用的数据集中存在类别不平衡问题, 对此, 设计了一种基于误分类样本数量动态调整权重的交叉熵损失函数, 以改进传统的交叉熵损失函数。该函数的核心思想是在训练过程中, 根据模型对各类别样本的误分类情况, 动态调整类别权重, 从而提高模型对易误分类类别的识别能力。

首先, 根据训练集中各类别样本的数量设定初始权重  $initial\_weights_i$ , 如式 (2) 所示:

$$initial\_weights_i = \frac{total\_samples}{count_i} \quad (2)$$

其中,  $total\_samples$  表示样本总数,  $count_i$  表示类别  $i$  的样本数量。

初始权重将用于加权交叉熵损失函数, 以对不同类别的样本施加相应的惩罚。在训练过程中, 每经过  $N$  轮迭代 (本文中取  $N=5$ ), 统计各类别的误分类数量, 并基于此计算新的

动态权重 $current_{weights}$ ,如式(3)所示:

$$current_{weights} = \frac{misclass\_count_i}{total\_misclass} \quad (3)$$

其中,  $total\_misclass$  表示所有类别的总误分类数量,  $misclass\_count_i$  表示类别  $i$  的误分类样本数量。

接下来,通过加权求和的方式结合初始权重和误分类权重,新权重的计算方法如式(4)所示:

$$class_{weights} = \alpha * current_{weights} + (1-\alpha) * initial_{weights} \quad (4)$$

其中,  $\alpha$  表示调整系数(默认值为 0.5),用于平衡初始权重和误分类权重在新权重中的影响。该方法使模型逐步增加对误分类率较高类别的权重,加大对这些类别的惩罚力度,从而提升模型的分类性能。

### 3.5 漏洞检测模块

在提取异构图特征和字节码指令序列特征后,将二者拼接以生成联合特征表示。随后,将该联合特征输入全连接层,执行二分类任务,并通过  $\text{argmax}$  函数选择得分最高的类别,以确定是否存在漏洞。具体流程如式(5)所示:

$$y = \text{argmax}(\mathbf{W}x + \mathbf{b}) \quad (5)$$

其中,  $x$  表示拼接后的联合特征;  $\mathbf{W}$  表示特征权重矩阵;  $\mathbf{b}$  表示偏置向量;  $y$  的取值为 0 或 1,分别对应“无漏洞”与“有漏洞”两种情况。

## 4 实验与结果分析

### 4.1 数据集

本文使用了 Zhen 等<sup>[19]</sup>整理的包含 17 968 个真实智能合约的数据集,包括正常合约和多种漏洞类型合约,具体如下表 1 所列。

表 1 数据集统计  
Table 1 Dataset statistics

数据集	数量
正常合约	6 893
整数溢出漏洞	9 335
时间戳依赖漏洞	1 240
自毁漏洞	500

### 4.2 实验环境及参数设置

实验环境:本实验在 Linux 操作系统上完成,硬件配置为一块具有 16 GB 显存的 Tesla T4 GPU 和 Intel Xeon CPU。软件开发环境使用 Python 3.10.12,并基于 PyTorch 2.5.0 深度学习框架,支持 CUDA 12.1。同时,实验还使用了  $\text{torch\_geometric}$  2.6.1 库以处理图数据。

主要参数:在模型训练中,使用 Adam 优化器,初始学习率为 0.01。模型架构包含 3 层关系图卷积层(RGCNConv),其中异构图节点的嵌入维度为 48,操作码嵌入维度设为 16,RGCN 隐藏层的维度为 64,且采用 0.5 的 Dropout 概率。数据集被分为训练集和测试集,比例为 8:2,批量大小设定为 32。整个训练过程共进行 30 轮迭代。

### 4.3 评价指标

为了评估模型的性能,本文选择了精确率(Precision,P)、召回率(Recall,R)和 F1 分数(F1-score)作为评估指标。精确率表示在预测为正类的样本中,实际为正类的比例,反映模型减少误报的能力。召回率表示在所有实际为正类的样本中,

被正确预测为正类的比例,反映模型识别漏洞的能力。F1 分数用于精确率与召回率之间的平衡。这些指标的定义如式(6)~式(8)所示:

$$P = \frac{TP}{TP + FP} \quad (6)$$

$$R = \frac{TP}{TP + FN} \quad (7)$$

$$F1 = 2 * \frac{P * R}{P + R} \quad (8)$$

其中,  $TP$  表示实际存在漏洞且被正确预测为有漏洞的合约数量,  $TN$  表示实际没有漏洞且被正确预测为没有漏洞的合约数量,  $FP$  表示实际没有漏洞却被错误预测为有漏洞的合约数量,  $FN$  表示实际存在漏洞却被错误预测为没有漏洞的合约数量。

这些评估指标能够全面评估模型在智能合约漏洞检测任务中的表现。

### 4.4 实验结果分析

为了验证本文 RGCN-Resnet1D 方法的有效性,选择了 3 款主流的智能合约漏洞检测工具(SmartCheck, Mythril, Oyente)和 2 种深度学习网络(LSTM 和 GCN)进行对比,具体方法简介如下。

SmartCheck:一种基于中间表示的分析工具,擅长检测智能合约中的潜在漏洞和安全隐患。SmartCheck 能够通过识别合约中的中间表示结构特征,捕捉漏洞模式,并提供相应的安全验证,提高对合约风险的识别准确性。

Mythril:一种基于符号执行技术的漏洞检测工具,能够识别智能合约中的未验证函数调用、整数溢出等安全问题。Mythril 通过符号执行路径分析,检测代码执行的潜在风险,从而有效定位合约中的安全隐患。

Oyente:一种基于符号执行的智能合约检测工具,专注于捕捉代码中的安全漏洞。Oyente 通过路径分析和条件判断,定位代码中的不安全路径,帮助预防执行过程中可能产生的安全风险。

LSTM:一种增强的循环神经网络,擅长学习智能合约代码中的时间序列特征。LSTM 可以识别代码中的顺序关系与依赖,捕捉合约中的漏洞模式特征,从而有效预测可能的安全问题。

GCN:一种处理图结构数据的神经网络。智能合约代码可以表示为节点和连接关系的图结构,GCN 通过学习这种结构特性识别代码中的图形模式,帮助检测可能的漏洞结构,提高合约安全性的预测准确度。

表 2 列出了所有方法在不同智能合约漏洞类型上的检测效果。

从表 2 的结果可知,RGCN-Resnet1D 在漏洞检测任务中的表现显著优于传统工具和其他基线模型。传统漏洞检测工具在多个指标上表现不佳,而 LSTM 通过学习字节码指令的序列特征,性能相比传统工具有所提升。GCN 通过学习智能合约的图结构信息,性能优于单纯的 LSTM 模型,在 3 种类型漏洞检测中的 F1 分数分别达到了 88.51%,65.20% 和 72.00%。然而,值得注意的是,本文 RGCN-Resnet1D 方法通过结合异构图和字节码指令序列特征,实现了更高的检测性

能,其在 3 类漏洞检测中的 F1 分数分别达到 95.43%, 90.67% 和 92.31%。这些结果表明,采用异构图和字节码

指令序列进行智能合约字节码漏洞检测具有巨大的应用潜力。

表 2 RGCN-Resnet1D 方法与其他方法的性能对比结果

Table 2 Performance comparison results of RGCN-ResNet1D method with other methods

方法	整数溢出			时间戳依赖			自毁漏洞		
	P	R	F1	P	R	F1	P	R	F1
SmartCheck	53.21	33.62	41.20	43.11	38.23	40.52	46.32	40.12	42.99
Mythril	41.32	70.69	52.14	58.61	44.59	50.64	58.03	54.74	56.33
Oyente	36.18	54.62	43.52	43.92	23.72	30.80	—	—	—
LSTM	69.48	66.96	68.20	49.59	54.50	51.93	62.86	56.41	59.46
GCN	89.68	87.37	88.51	66.67	63.79	65.20	69.23	75.00	72.00
RGCN-Resnet1D	<b>96.33</b>	<b>94.56</b>	<b>95.43</b>	<b>93.92</b>	<b>87.63</b>	<b>90.67</b>	<b>91.14</b>	<b>93.51</b>	<b>92.31</b>

## 4.5 消融实验

### 4.5.1 数据流边和字节码指令序列的影响

为了验证数据流边扩展和字节码指令序列在漏洞检测中的有效性,本文设计了以下消融实验。

1)控制流图+GCN 方法:将智能合约的字节码转换为控制流图(CFG),并在此基础上构建图卷积网络(GCN)模型,以检测潜在漏洞。

2)异构图+RGCN 方法:在控制流图上增加数据流边,构建异构图,并采用 RGCN 模型,以评估数据流边扩展对漏洞检测性能的提升效果。

3)双通道方法(RGCN+Resnet1D):在异构图 RGCN 模

型的基础上,加入字节码指令序列特征提取的 Resnet1D 通道,形成 RGCN-Resnet1D 双通道网络,进一步验证字节码指令序列特征对漏洞检测的贡献。

实验结果如表 3 所列。与基于控制流图的 GCN 模型相比,基于异构图的 RGCN 模型在 3 种漏洞类型的精确率、召回率和 F1 值方面均表现出显著提升。这表明,加入数据流边能够有效增强漏洞检测的性能。此外,与基于异构图的 RGCN 模型相比,结合异构图和指令序列的 RGCN-Resnet1D 双通道网络在各类漏洞检测指标上表现更优,进一步验证了指令序列特征在漏洞检测中的有效性。

表 3 有无数据流边和字节码指令序列的对比

Table 3 Comparison of with and without data flow edges and bytecode instruction sequences

方法	整数溢出			时间戳依赖			自毁漏洞		
	P	R	F1	P	R	F1	P	R	F1
GCN	89.68	87.37	88.51	66.67	63.79	65.20	69.23	75.00	72.00
RGCN	92.83	91.21	92.02	79.19	81.68	80.41	86.96	85.71	86.33
RGCN-Resnet1D	<b>96.33</b>	<b>94.56</b>	<b>95.43</b>	<b>93.92</b>	<b>87.63</b>	<b>90.67</b>	<b>91.14</b>	<b>93.51</b>	<b>92.31</b>

### 4.5.2 基于误分类样本的动态权重交叉熵损失的影响

为了验证基于误分类样本数量动态调整权重的交叉熵损失函数在处理类别不平衡问题上的有效性,开展了相应的消融实验。选用 3 种不同的交叉熵损失函数,分别为基于误分类样本数量动态调整权重的交叉熵损失函数、标准交叉熵损失函数,以及采用固定权重的交叉熵损失函数。所有实验均在相同的神经网络结构 RGCN-ResNet1D 上进行对比分析,以确保实验条件的一致性。通过对比分析这 3 种损失函数的

表现,可以全面评价动态权重调整策略在处理类别不平衡问题中的优势与实际效果。

实验结果如表 4 所列,采用动态调整权重的损失函数的模型在 3 种漏洞检测任务中的 F1 得分均有所提升,特别是在时间戳依赖漏洞检测任务中,使用动态权重调整模型的 F1 分数相比无权重交叉熵模型提高了 9.27 个百分点。这些结果充分证明了基于误分类样本数量动态调整权重的交叉熵损失函数在解决类别不平衡问题上的有效性。

表 4 不同类型损失函数的对比

Table 4 Comparison of different types of loss functions

损失函数类型	整数溢出			时间戳依赖			自毁漏洞		
	P	R	F1	P	R	F1	P	R	F1
无权重	92.82	91.58	92.19	93.33	72.16	81.40	<b>97.18</b>	77.53	86.25
固定权重	92.88	93.57	93.22	<b>94.30</b>	81.42	87.39	91.67	90.59	91.12
动态权重	<b>96.33</b>	<b>94.56</b>	<b>95.43</b>	93.92	<b>87.63</b>	<b>90.67</b>	91.14	<b>93.51</b>	<b>92.31</b>

**结束语** 本文提出了一种结合异构图和字节码指令序列的智能合约字节码漏洞检测方法 RGCN-ResNet1D。该方法通过解析字节码生成包含控制流和数据流信息的异构图,使用 RGCN 网络提取图结构特征,并采用 ResNet1D 网络提取

字节码指令序列特征,从而显著提高了漏洞检测的准确性。同时,本文设计了基于误分类样本数量动态调整权重的交叉熵损失函数,改善了 RGCN-ResNet1D 方法在少数类别上的检测表现。实验结果显示,该方法在检测准确性和应对数据

不平衡问题上表现出色。

鉴于智能合约领域的快速发展及新型漏洞的不断涌现,未来研究将重点增强对新漏洞的检测能力,并探索引入自动化修复机制,以进一步提升智能合约的安全性。

### 参 考 文 献

- [1] SIEGEL D. Understanding the DAO attack[EB/OL]. <https://www.coindesk.com/understanding-dao-hack-journalists>.
- [2] BlockCAT. On the Parity multi-sig wallet attack[EB/OL]. <https://medium.com/blockcat/on-the-parity-multi-sig-wallet-attack-83fb5e7f4b8c>.
- [3] PRETROV S. Another Parity wallet hack explained[EB/OL]. <https://medium.com/@Pr0Ger/another-parity-wallet-hack-explained-847ca46a2e1c>.
- [4] Wikipedia. Poly network exploit[EB/OL]. [https://en.wikipedia.org/wiki/Poly\\_Network\\_exploit](https://en.wikipedia.org/wiki/Poly_Network_exploit).
- [5] QIAN P, LIU Z G, HE Q M, et al. A Survey of Security Vulnerability Detection Techniques for Smart Contracts [J]. *Journal of Software*, 2022, 33(8): 3059-3085.
- [6] HILDENBRANDT E, SAXENA M, RODRIGUES N, et al. Kevm: A complete formal semantics of the ethereum virtual machine[C] // 2018 IEEE 31st Computer Security Foundations Symposium(CSF). IEEE, 2018.
- [7] AMANI S, BÉGEL M, BORTIN M, et al. Towards verifying ethereum smart contract bytecode in Isabelle/HOL[C] // Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs, 2018: 66-77.
- [8] LUU L, CHU D H, OLICKEL H, et al. Making smart contracts smarter[C] // Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016: 254-269.
- [9] MUELLER B. A framework for bug hunting on the ethereum blockchain[J]. *ConsenSys/mythril*, 2017.
- [10] JIANG B, LIU Y, CHAN W K. Contractfuzzer: Fuzzing smart contracts for vulnerability detection[C] // Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, 2018: 259-269.
- [11] ALBERT E, GORDILLO P, LIVSHITS B, et al. Ethir: A framework for high-level analysis of Ethereum bytecode[C] // Proceedings of International Symposium on Automated Technology for Verification and Analysis. Cham: Springer-Verlag, 2018.
- [12] TANN W J W, HAN X J, GUPTA S S, et al. Towards safer

smart contracts: A sequence learning approach to detecting security threats[J]. *arXiv*:1811.06632, 2018.

- [13] HU H W, XU Y D. SCSSGuard: Deep SCAM detection for Ethereum smart contracts[J]. *arXiv*:2105.10426, 2021.
- [14] GU W Y, WANG G J, LI P Q, et al. Detecting unknown vulnerabilities in smart contracts with the CNN-BiLSTM model[J]. *International Journal of Information Security*, 2025, 24(1): 33.
- [15] WANG Z F, WU W X, ZENG C Y, et al. Graph Neural Networks Enhanced Smart Contract Vulnerability Detection of Educational Blockchain[J]. *arXiv*:2303.04477, 2023.
- [16] ZHANG J, LU G H, YU J. A Smart Contract Vulnerability Detection Method Based on Heterogeneous Contract Semantic Graphs and Pre-Training Techniques [J]. *Electronics*, 2024, 13(18): 3786.
- [17] DUY P T, KHOA N H, QUYUE N H, et al. Vulnsense: efficient vulnerability detection in ethereum smart contracts by multimodal learning with graph neural network and language model[J]. *International Journal of Information Security*, 2025, 24(1): 48.
- [18] ROSSINI M, ZICHICHI M, FERRETTI S. Smart contracts vulnerability classification through deep learning[C] // Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems, 2022: 1229-1230.
- [19] ZHEN Z, ZHAO X, ZHANG J, et al. DA-GNN: A smart contract vulnerability detection method based on Dual Attention Graph Neural Network [J]. *Computer Networks*, 2024, 242: 110238.



**SONG Jianhua**, born in 1973, Ph.D, professor, postgraduate supervisor, is a member of CCF (No. 27785M). Her main research interest is network and information security.



**ZHANG Yan**, born in 1974, Ph.D, professor, postgraduate supervisor. His main research interest is code security.

(责任编辑:何杨)