特约综述

张小松,电子科技大学计算机科学与工程学院(网络空间安全学院)教授、博士生导师、四川省学术和技术带头人、国家网络安全十大优秀人才(2017 年)、全国创新争先奖获得者;入选十三五国家自然基金项目优秀成果,以第一完成人获得 2019 年国家科技进步一等奖、2012 年国家科技进步二等奖;2023 年获得第三届"四川杰出人才奖".张小松教授长期致力于信息技术与安全领域的基础研究、技术攻关和人才培养,围绕软件与网络安全、大数据及区块链安全及应用中的核心问题,原创提出以"智感""透析""活现"为核心要素的主动网络安全模型,在威胁感知、检测防御、追踪溯源等方面开展系统性、开拓性的创新工作.张小松教授在 CCF A 类顶级学术会议与期刊 CCS,NDSS,TSE,TIFS 等发表学术论文 70 余篇,一作出版《区块链安全技术及应用》《人工智能算法安全与安全应用》《卫星互联网安全概论》《物联网信息安全技术》专著.张小松教授近年带领团队在区块链安全领域开展了卓有成效的研究工作,取得重要学术成果.在区块链智能合约性能优化方面的研究成果分别在 SANER 2017 (CCF B 类学术会议)和 ICSE 2018(CCF A,软件工程顶级学术会议)发表;在区块链的交易分析和安全应用方面研究成果发表在 INFOCOM 2018 (CCF A 类学术会议,并获 INFOCOM 最佳论文奖项);在区块链 EVM 安全性保护方面的研究成果发表在 ISPEC 2017,并拿到了唯一最佳论文奖项等.

基于深度学习的智能合约漏洞检测方法综述

张小松[1,2,3], 牛伟纳[1], 黄世平[1], 孙裕俨[1], 贺哲远[1]
(1.电子科技大学计算机科学与工程学院, 成都 611731; 2.交子金融科技中心, 成都 610095;
3.区块链安全与平台技术教育部工程研究中心, 成都 611731)

摘　要: 智能合约是区块链三大特点之一,也是区块链具有应用价值和灵活性的领域.本质上,智能合约是一段用特定脚本语言实现的代码,不可避免地存在安全漏洞风险.如何及时准确地检查出各种智能合约的漏洞,就成为区块链安全研究的重点和热点.为了检测智能合约漏洞,研究者提出了各种分析方法,包括符号执行、形式化验证和模糊测试等.随着人工智能技术的快速发展,越来越多基于深度学习的方法被提出,并且在多个研究领域取得了很好的效果.目前,针对基于深度学习的智能合约漏洞检测方法并没有被详细地调查和分析.本文首先简要介绍了智能合约的概念以及智能合约漏洞相关的安全事件;然后对基于深度学习的方法中常用的智能合约特征进行分析;同时对智能合约漏洞检测中常用的深度学习模型进行描述.此外,为了进一步推动基于深度学习的智能合约漏洞检测方法的研究,本文将近年来基于深度学习的智能合约漏洞检测方法根据其特征提取形式进行了总结分类,从文本处理、静态分析和图像处理 3 个角度进行了分析介绍;最后,总结了该领域面临的挑战和未来的研究方向.

关键词: 区块链; 以太坊; 智能合约; 漏洞检测; 深度学习
中图分类号: TP311　　**文献标识码**: A　　DOI: 10.19907/j.0490-6756.2023.020001

收稿日期: 2022-12-09
基金项目: 国家自然科学基金联合基金(U19A2066); 四川省自然科学基金(2022NSFSC0871); 深圳市杰出人才培养经费资助
通讯作者: 张小松. E-mail: johnsonzxs@uestc.edu.cn

A survey of smart contract vulnerability detection methods based on deep learning

ZHANG Xiao-Song[1,2,3], *NIU Wei-Na*[1], *HUANG Shi-Ping*[1], *SUN Yu-Yan*[1], *HE Zhe-Yuan*[1]

(1. School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; 2. Jiaozi Fintech Center, Chengdu 610095, China; 3. Engineering Research Center of Blockchain Security and Platform Technology, Ministry of Education, Chengdu 611731, China)

Abstract: Smart contract is one of the three major characteristics of blockchain and an area where blockchain has application value and flexibility. In essence, a smart contract is a piece of code implemented in a specific scripting language, which inevitably has the risk of security vulnerabilities. How to accurately and timely detect the vulnerabilities of various smart contracts has become the focus and hotspot of blockchain security research. To detect vulnerabilities in smart contracts, researchers have proposed various analysis methods, including symbolic execution, formal verification and fuzzing. With the rapid development of artificial intelligence technology, more and more deep learning-based methods have been proposed and have achieved good results in several research areas. Currently, deep learning-based smart contract vulnerability detection methods have not been investigated and analyzed in detail. This paper introduces the concept of smart contracts and security events related to smart contract vulnerabilities, then outlines commonly used smart contract features and describes the deep learning models commonly used in smart contract vulnerability detection. To promote the research of deep learning-based smart contract vulnerability detection methods, this paper summarizes and classifies recent deep learning-based smart contract vulnerability detection methods according to their feature extraction forms and analyzes them from three perspectives: text processing, static analysis, and image processing. Finally, the paper summarizes the challenges and future research directions in this field.

Keywords: Blockchain; Ethereum; Smart contract; Vulnerability detection; Deep learning

1　引　言

区块链技术[1]最早可以追溯到 2008 年中本聪比特币白皮书[2]的出版. 紧接着 2009 年比特币正式发行,使区块链技术展现出广阔的应用前景,吸引了学术界和业界的广泛关注,拉开了区块链 1.0 的序幕. 随着对区块链技术的不断深入研究,区块链技术不再局限于单纯的去中心化货币交易. 智能合约在区块链上的实现,加速了区块链技术在加密货币之外其他领域的应用. 而第一个支持开发智能合约的区块链即以太坊[3]的发布,标志着区块链 2.0 时代的开始. 通过区块链的去中心化共识机制,智能合约允许相互不信任的用户在不需要任何第三方可信机构的情况下完成数据交换或交易,是区块链最具有应用价值的领域. 随后出现了越来越多支持智能合约的区块链平台. 截至目前,仅以太坊上就存在超过 5000 万份智能合约,而以太坊本身的市值也超过 1500 亿美元,是最广泛使用的支持智能合约的区块链平台. 作为在区块链中运行的程序,控制着区块链上大量资金的智能合约不可避免地存在由程序缺陷导致的安全漏洞[4,5]. 因此,如何准确及时地检查出各种智能合约的漏洞,成为区块链安全研究的重点和热点.

为了确保智能合约的安全性,越来越多的研究人员开始研究智能合约的漏洞检测技术,开发了各种工具来检测和预防现有的智能合约漏洞[6]. 采用的技术主要有符号执行[7-10]、形式化验证[11,12]和模糊测试[13,14]. 符号执行是应用最广泛的方法. 它是指在程序执行过程中将不确定的输入转化为符号值,通常结合约束求解器对程序执行路径进行求解. 符号执行可以实现更准确、更全面的程序分析,但由于程序分支和循环的影响,通常会面临路径爆炸等问题. 形式化验证主要是采用严格的可演绎的描述语言或逻辑来描述程序的属性和特征,并使用数学逻辑证明和推理来构建形式化规范,以确定安全属性设置是否符合预期. 但是形式化验证方法需

要较强的逻辑推理能力，且自动化程度较低. 模糊测试使用随机生成的测试样本作为智能合约的输入，通过监控智能合约的执行过程判断是否触发程序漏洞或其他异常行为. 模糊测试方法能够有效检测合约漏洞，但需要事先获取智能合约的源码和 ABI 接口信息.

近年来，基于深度学习的程序分析方法在安全检测领域逐渐流行[15]. 深度学习方法自动化程度高，可以从大量的数据中提取出程序的隐藏特征，突破了基于规则的传统漏洞检测方法的局限. 本文通过调研近几年基于深度学习的智能合约漏洞检测方法，总结深度学习在智能合约漏洞检测领域的应用现状，讨论目前工作中存在的问题和挑战，并针对不足之处探讨未来的研究方向和思路.

2　智能合约

2.1　智能合约概述

智能合约的概念在 1997 年由 Szabo 首次提出[16]. Szabo 将智能合约定义为一套以数字形式定义的承诺，以及合约参与方可以在上面执行这些承诺的协议. 直到 2015 年以太坊的出现，智能合约这一概念才得以付诸实践.

Solidity[17]是专门用来编写智能合约的图灵完备的编程语言. 以太坊智能合约主要通过 Solidity 编写，由函数、事件和状态变量等组成. 编译完成后，智能合约代码被转换为以太坊虚拟机 EVM 字节码，然后通过合约创建交易部署在以太坊区块链上. 智能合约成功创建后，会有一个唯一的地址标识，生成一个智能合约账户.

以太坊智能合约账户由可执行代码、合约地址、私有状态变量和以太币余额组成. 一个包含目标智能合约地址和调用参数的交易可以实现对智能合约的调用. 以太坊虚拟机（Ethereum Virtual Machine，EVM）是一个基于堆栈的虚拟机，提供了一个与网络隔离的运行时环境来执行智能合约代码.

以太坊的蓬勃发展让更多人意识到智能合约的重要价值，因此出现了越来越多的支持智能合约运行的区块链，基于智能合约实现的区块链应用数量也井喷式增加[18].

2.2　智能合约应用

区块链技术和智能合约催化了去中心化应用（Decentralized Application，DApp）市场的发展. DApp 是可以自主运行的应用程序，通过使用智能合约，在区块链系统上执行. 与传统应用程序一样，DApp 为用户提供一些功能，前端可以由已有的技术构建. 但是与传统应用程序不同的是，由于后端基于区块链系统实现，DApp 无需人工干预即可执行，也不属于任何一个组织或者实体，具有去中心化属性. DApp 无需信任和透明的性质推动了去中心化金融（Decentralized Finance，DeFi）的发展，越来越多的组织和开发者开始在区块链上开发自己的 DApp. 根据 Dune 网站上的数据统计[19]，区块链上智能合约的数量逐年持续增长. 如图 1 所示，目前以太坊上部署的智能合约已经超过了 5000 万份. CoinMarketCap 的数据显示，以智能合约为技术支持的 DeFi 市场的市值超过 430 亿美元[20]. 这表明智能合约具有巨大的市场潜力. 与此同时，在医疗保健[21,22]、供应链[23,24]和物联网[25,26]等非金融领域也出现了大量基于区块链的应用.

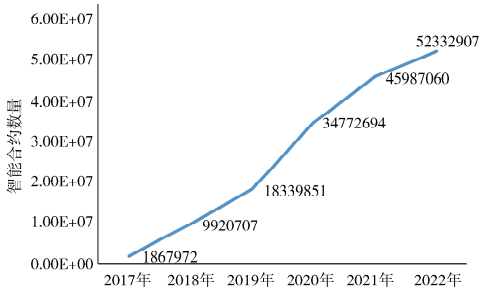

图 1　以太坊智能合约数量统计

Fig. 1　Statistical of the number of Ethereum smart contracts

2.3　智能合约安全

近年来智能合约攻击事件层出不穷. 2016 年 6 月，黑客利用去中心化自治组织（Decentralized Autonomous Organization）合约的重入漏洞窃取了价值约 6000 万美元的以太币[27]. 2017 年 7 月，由于 Parity Multi-Sig Wallet 合约的 delegatecall 漏洞[28]，近 3 亿美元的以太币被冻结. 2018 年 4 月，恶意攻击者借助 Beauty Chain 合约的整数溢出漏洞[29]，发布了无限数量的 BEC 令牌，使得 BEC 市值蒸发为零. 2019 年 5 月，Binance Exchange 被黑客入侵，导致 7000 多个比特币被盗[30]. 2022 年 10 月 7 日，由加密货币交易所币安孵化，全球最活跃的公链之一的 BNB Chain 被黑客攻击，黑客利用跨链桥（可以帮助实现不同区块链之间的资产流动）漏洞分两次共获取 200 万枚 BNB，价值约 5.66 亿美元[31]. 如图 2 所示，根据成都链安安全团队历年数据统计，区块链上因安全事件导致的经济损失呈逐年上升趋势. 相关分析表

明,超过 80%的安全事件都是由智能合约漏洞引起的[32]. 区块链安全形势愈发严峻.

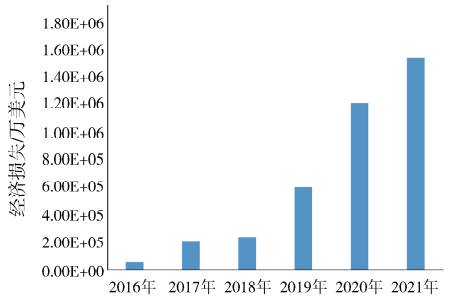

图 2　区块链安全事件经济损失

Fig. 2　Economic loss of blockchain security incidents

3　深度学习概述

深度学习属于人工智能领域,被广泛应用于计算机视觉、语音识别和自然语言处理等领域. 将深度学习应用于智能合约漏洞检测已经成为一个趋势.

典型的深度学习是一个包含多个隐藏层的神经网络,每一层包含不同的神经元,每个神经元具有不同的权重和激活函数. 深度学习最显著的价值在于能够自动提取和抽象数据的高阶特征,避免了繁琐的人工特征提取过程.

3.1　深度学习模型

通过总结智能合约漏洞检测中使用的深度学习模型发现,目前的检测模型主要有卷积神经网络(Convolutional Neural Network,CNN)、循环神经网络(Recurrent Neural Network,RNN)和图卷积神经网络(Graph Convolutional Network,GCN).

(1) 卷积神经网络. 卷积神经网络是一种用于处理图像数据等具有网格结构数据的神经网络. 其一般结构包括卷积层、池化层、全连接层和输出层. 与其他深度学习模型相比,卷积神经网络在图像和语音识别方面可以给出更好的结果. 卷积神经网络使用卷积核来共享参数,显著减少了参数数量并使用了局部数据信息.

(2) 循环神经网络. 循环神经网络是一种用于处理序列数据的神经网络. 循环神经网络已成功应用于许多时间序列问题,例如自然语言处理、语音识别和机器翻译. RNN 对序列中的每个元素和之前计算的输出执行相同的操作. 因此,RNN 由两个输入(当前状态和前一个状态)结合起来确定网络的最终输出.

(3) 图卷积神经网络. 图卷积神经网络是一种用于图结构数据的神经网络. GCN 广泛应用于许多与图结构数据相关的问题,如蛋白质结构预测、化合物稳定性预测和社交网络分析等. GCN 可以看作一种消息传递网络,图中每个节点根据自己的特征生成消息向量,消息通过边传递给邻居节点,然后每个节点通过聚合邻居节点的消息更新自身的特征向量.

3.2　用于深度学习的智能合约特征

基于深度学习的智能合约漏洞检测方法需要从智能合约中提取漏洞相关特征,智能合约特征的结构决定了深度学习模型可以从中学习到的漏洞相关信息. 下面以图 3 所示的智能合约代码为例,分别介绍深度学习方法中常用的智能合约特征.

(1) 字节码和操作码. 智能合约最终被编译为字节码才能被 EVM 执行,字节码以十六进制表示. 操作码是人类可读的字节码表示,图 3 所示智能合约对应的操作码如图 4 所示.

```
pragma solidity >=0.5.16;
contract Example {
    uint256 number;
    function store(uint256 num) public {
        number = num;
    }
}
```

图 3　智能合约示例代码

Fig. 3　Example code of smart contract

```
0x0: PUSH1 0x80
0x2: PUSH1 0x40
0x4: MSTORE
0x5: CALLVALUE
0x6: DUP1
0x7: ISZERO
0x8: PUSH1 0x0f
0xa: JUMPI
0xb: PUSH1 0x00
0xd: DUP1
0xe: REVERT
0xf: JUMPDEST
0x10: POP
0x11: PUSH1 0x04
0x13: CALLDATASIZE
0x14: LT
0x15: PUSH1 0x28
0x17: JUMPI
0x18: PUSH1 0x00
0x1a: CALLDATALOAD
0x1b: PUSH1 0xe0
0x1d: SHR
0x1e: DUP1
0x1f: PUSH4 0x6057361d
0x24: EQ
0x25: PUSH1 0x2d
0x27: JUMPI
0x28: JUMPDEST
0x29: PUSH1 0x00
0x2b: DUP1
0x2c: REVERT
0x2d: JUMPDEST
0x2e: PUSH1 0x56
0x30: PUSH1 0x04
0x32: DUP1
0x33: CALLDATASIZE
0x34: SUB
0x35: PUSH1 0x20
0x37: DUP2
0x38: LT
0x39: ISZERO
0x3a: PUSH1 0x41
0x3c: JUMPI
0x3d: PUSH1 0x00
0x3f: DUP1
0x40: REVERT
0x41: JUMPDEST
0x42: DUP2
0x43: ADD
0x44: SWAP1
0x45: DUP1
0x46: DUP1
0x47: CALLDATALOAD
0x48: SWAP1
0x49: PUSH1 0x20
0x4b: ADD
0x4c: SWAP1
0x4d: SWAP3
0x4e: SWAP2
0x4f: SWAP1
0x50: POP
0x51: POP
0x52: POP
0x53: PUSH1 0x58
0x55: JUMP
0x56: JUMPDEST
0x57: STOP
0x58: JUMPDEST
0x59: DUP1
0x5a: PUSH1 0x00
0x5c: DUP2
0x5d: SWAP1
0x5e: SSTORE
0x5f: POP
0x60: POP
0x61: JUMP
```

图 4　智能合约操作码

Fig. 4　Opcodes of smart contract

(2) 控制流图. 控制流图(Control Flow Graph,CFG)是一种用于表示程序可以处理的所有可能的执行路径的图. 一般使用智能合约的操作码构建

CFG. CFG 中，每个节点代表一个基本块，每个基本块只有一个入口和一个出口，基本块之间通过程序跳转连接. 图 3 智能合约生成的控制流图如图 5 所示.

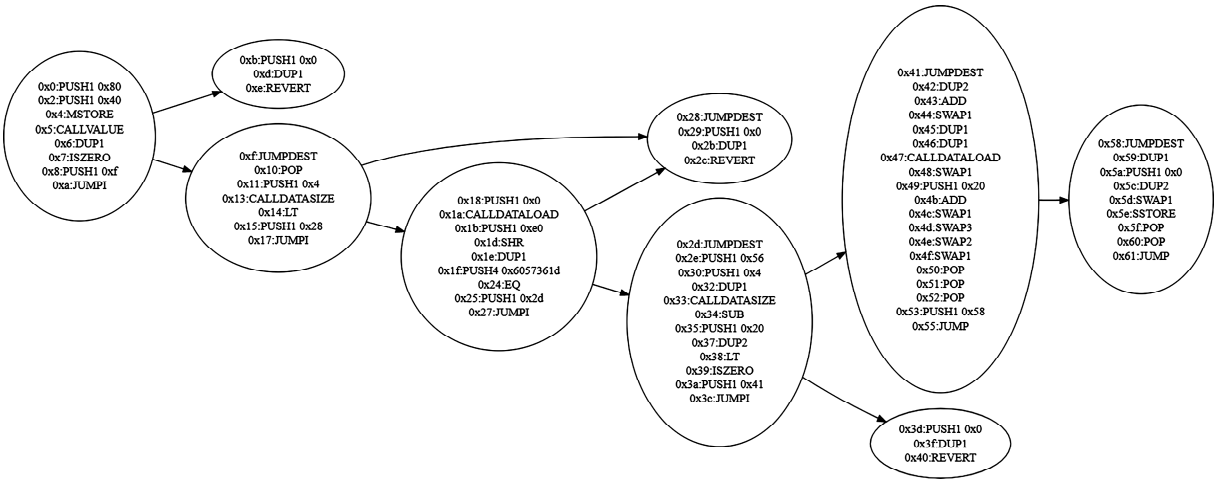

图 5　智能合约控制流图
Fig. 5　CFG of smart contract

(3) 抽象语法树. 抽象语法树(Abstract Syntax Tree, AST)是一种用于推理书面语言的数据结构. 使用编译器将源代码转换为数据结构树，其中每个节点代表源代码的一个句法元素，树形图显示程序中每个元素的流向. 在不破坏结构化信息的情况下，AST 可以提供源代码的详细信息. 图 3 所示智能合约的抽象语法树如图 6 所示.

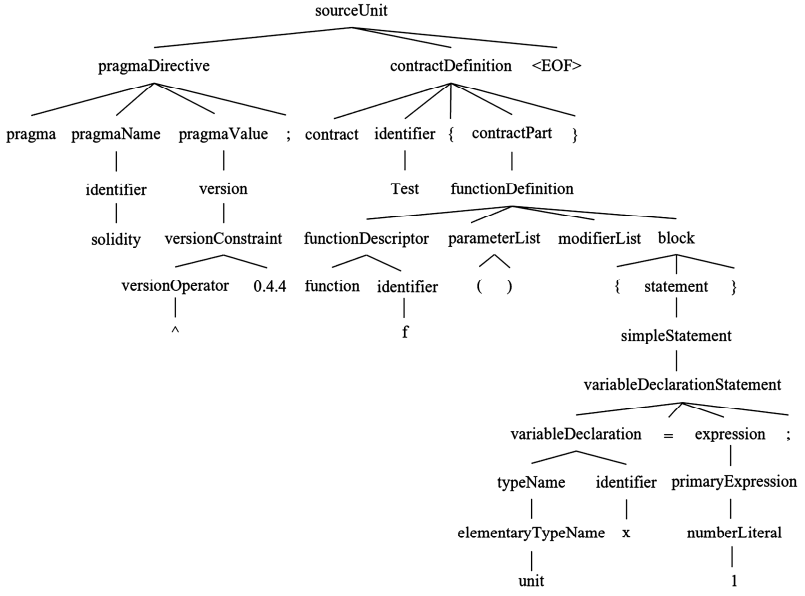

图 6　智能合约抽象语法树
Fig. 6　AST of smart contract

4　基于深度学习的智能合约漏洞检测

如表 1 所示，根据其特征提取方法，将基于深度学习的智能合约漏洞检测模型分为三类，分别是基于文本处理、基于静态分析和基于图像处理的漏洞检测.

基于文本处理的方法将智能合约源代码、字节码和操作码当作连续的文本序列，使用自然语言处理方法，提取智能合约漏洞的语义信息和特征. 基于静态分析的方法在使用静态分析的基础上，利用分析结果如 AST、CFG 等进一步提取智能合约的结构特征，丰富漏洞相关信息. 基于图像处理的方

法,通过将智能合约字节码或操作码序列进行切片组合,构建灰度图矩阵,然后应用图像处理的方法提取特征并构建模型.

4.1　基于文本处理的漏洞检测

文献[33]使用序列学习的方法检测智能合约漏洞.将操作码序列中的操作码转换为 one-hot 向量表示,通过嵌入算法转换为更低维度的向量,同时捕获操作码之间的序列关系,然后将得到的向量序列输入到 LSTM[34]模型中.在包含 62 万份以太坊智能合约的数据集上对模型进行训练,实验结果表明,序列学习的方法优于基于符号执行的分析工具.训练的模型准确率和 F_1 分数分别为 99.57%和 86.04%.文献[35]将预训练方法与 AWD-LSTM[36]模型相结合,提出了一种基于预训练编码器的智能合约漏洞检测方法.该方法首先使用相同长度的输入输出向量训练常规架构的 AWD-LSTM 模型,得到包含操作码语义信息的预训练编码器,然后将常规架构中的解码器替换成用于多分类的全连接层网络,实现智能合约多漏洞检测.

表 1　基于深度学习的智能合约漏洞检测方法

Tab. 1　Smart contract vulnerability detection method based on deep learning

分类	文献	特征	模型
基于文本处理	[33]	操作码	LSTM
	[35]	操作码	AWD-LSTM
	[37]	操作码	GRU+DNN
	[39]	操作码	Attention+CNN
	[40]	源码	CNN+BiGRU
	[44]	AST+字节码	PV-DM
	[46]	操作码	Ensemble Learning
基于静态分析	[48]	CFG	DNN
	[49]	源码	TMP、DR-GCN
	[50]	源码+专家模式	TMP+MLP+Attention
	[51]	源码+安全模式	TMP+CNN
	[52]	CFG+操作码序列	GCN+Attention
	[53]	操作码	Graph2Vec
	[55]	AST+DFG	GraphCodeBERT
	[57]	源码+漏洞相关切片	Bi-LSTM、LSTM-Attention
	[58]	CFG	GCN
基于图像处理	[59]	操作码切片矩阵	CNN、RF
	[60]	字节码切片矩阵	CNN
	[61]	字节码序列	CNN without Stride

文献[37]提出了一个智能合约漏洞检测框架 ESCORT.该框架分为两个部分.第一个部分是一个通用的特征提取器,应用 GRU[38]模型提取智能合约字节码序列的特征;第二个部分是一个多分支结构,每一个分支针对一个漏洞类型进行检测.该框架通过迁移学习快速支持新漏洞的检测,当一个新的漏洞被识别时,构建一个该漏洞的数据集,然后在模型上增加一个新的分支,冻结通用特征提取器和其他分支的参数,利用构建的新数据集对新分支进行训练,从而克服了传统模型的可扩展性和泛化性限制.

文献[39]提出了一种基于多任务学习的智能合约漏洞检测模型,由共享层和特定任务层组成.该模型构建了基于注意力机制的共享层网络,用于学习智能合约操作码序列的关键特征.特定任务层分为检测和识别两部分,检测任务用于检测智能合约是否存在漏洞,使用 CNN 构建二元分类网络;识别任务用于识别漏洞类别,使用 CNN 构建多标签分类网络.特定任务层接收共享层提取的特征作为输入,然后针对自身的特定任务进行训练.为了缓解智能合约数据集的类不平衡问题,采取了对有漏洞样本进行过采样、对无漏洞样本进行欠采样的方法.实验表明,多任务学习可以在多个任务之间共享特征,提高执行速度.

CBGRU[40]是一种混合深度学习模型.混合模型结合了不同深度学习模型的优点,更有利于分类任务.CBGRU 模型由嵌入层、特征提取层和分类层组成.嵌入层主要将字符级的智能合约源码处理成符合神经网络输入的矩阵表示.该模型分别使用 Word2Vec[41]和 FastText[42]进行词嵌入,作为特征提取层的两个输入分支.其中,Word2Vec 的嵌入结果作为 CNN 模型的输入;FastText 的嵌入结果作为 BiGRU 模型的输入.最后在分类层将两个分支提取的特征通过连接层进行特征融合,使用 Softmax 层进行分类.该模型在包含 47 587 个真实且唯一的智能合约数据集 SmartBugs Dataset-wild[43]上进行训练.实验结果表明,CBGRU 模型在针对智能合约漏洞的检测任务中具有更高的准确性,对重入漏洞、时间戳依赖漏洞和无限循环漏洞的检测准确率分别为 93.30%、93.02%和 93.16%.

当智能合约代码被重写后,漏洞检测将会变得困难.针对这一问题,文献[44]扩展了自然语言处理的 PV-DM[45]模型,实现智能合约代码的向量化.提出了一个智能合约漏洞检测静态分析工具

Eth2Vec. Eth2Vec 由 EVM 提取器和 PV-DM 模型两个模块组成. EVM 提取器对 EVM 字节码进行语法分析，为每一个合约创建指令级、块级、函数级和合约级的多级结构数据；PV-DM 模型将该数据作为输入，为每个智能合约生成对应的向量，然后通过计算与已知漏洞合约的向量相似性来识别测试合约中的漏洞. 实验表明，Eth2Vec 的检测能力高于基于 SVM 的方法，且对于代码重写具有很好的鲁棒性.

SCVDIE[46]是一个基于信息图和集成学习的智能合约漏洞检测方法. 该方法从收集的智能合约操作码中计算操作码共现频率，构建操作码共现矩阵. 通过预定义规则识别智能合约的关键操作码序列，基于共现关系构建信息图来表示智能合约的漏洞模式信息. 信息图中节点由操作码组成，操作码之间根据共现频率和出现顺序进行连接. 为了将节点转换为向量矩阵表示，提取每个智能合约信息图的所有路径，构建操作码序列集合，使用该集合训练 Word2Vec 等嵌入模型. 得到操作码序列的向量表示后，将其输入到集成学习模型中进行模型训练. 集成学习模型由两部分组成. 第一部分包括 CNN、RNN、RCNN、DNN、GRU 和 BiGRU，用于提取操作码序列的局部特征；第二部分是 Transformer[47]模型，用于提取全局特征，最后基于权重值将两种特征进行集成输出智能合约漏洞的预测概率. 实验结果表明，该方法明显优于静态分析工具，且集成学习模型能够有效地整合全局和局部知识，进而提高检测精度.

上述智能合约漏洞检测方法，将智能合约源码以及字节码、操作码序列看作自然语言进行处理，可以很方便地将自然语言处理领域的模型迁移到智能合约漏洞检测中，并且取得不错的效果. 但是编程语言与自然语言不同，编程语言是结构化的，信息之间具有明确的结构关系. 例如函数调用、数据之间的依赖等. 基于文本处理的方法仅仅关注代码中的序列关系，忽略了最重要的结构信息，无法学习到代码的结构特征.

4.2　基于静态分析的漏洞检测

文献[48]中提出了一种结合静态分析的智能合约漏洞检测方法. 由于智能合约的源码大部分无法获取，该方法收集智能合约的字节码序列，并将其反编译为 EVM 操作码序列. 然后利用静态分析方法通过操作码序列构建控制流图 CFG，在 CFG 上进行深度优先遍历得到一个新的具有智能合约执行顺序信息的操作码序列. 最后，应用 N-gram 和 TFIDF 模型将操作码序列转换为向量表示，将向量输入到神经网络中进行训练和分类.

文献[49]首次提出了基于图神经网络的智能合约漏洞检测方法. 该方法包括图生成、图归一化和图神经网络训练三个部分. 在图生成阶段，将智能合约源码转换为智能合约图，智能合约图包含三类节点和四类边. 三类节点包括主节点、次级节点和回调函数节点. 主节点表示合约自定义函数或内置函数. 这些函数是与特定漏洞相关的. 如针对重入漏洞，主节点表示 call. value( ) 函数以及调用 call. value() 的函数；次级节点表示主要节点函数调用中涉及到的关键变量；回调函数节点表示外部智能合约的回调函数. 四类边分别是控制流边、数据流边、顺序执行边和回调函数调用边，用于表示不同节点间的执行和调用关系. 在图归一化阶段，为了降低计算复杂度，只保留智能合约图中的主节点，然后将与主节点相关联的次级节点和回调函数节点的特征融合到主节点特征中. 图神经网络训练阶段，区别于传统的消息传播机制，提出了一种按照时间顺序沿着边更新传递信息的新型消息传播机制 TMP. TMP 更加符合智能合约执行的顺序特征. 实验表明，新型消息传播机制比传统消息传播机制具有更好的检测效果.

文献[50,51]对文献[49]的工作进行了扩展，提出了一个结合专家模式的自动漏洞检测系统. 该系统分别为重入、时间戳依赖和无限循环三种漏洞设计了相应的专家模式. 它使用多层感知机将提取的专家模式编码为特征向量，然后使用注意力编码器网络将专家模式特征和 TMP 图神经网络提取的图特征进行融合. 注意力编码器使用注意力机制计算出智能合约图特征和每个专家模式特征之间的权重，突出特征的重要性，使专家模式特征的融合具有可解释性. 实验表明，融合专家模式的方法与最先进的技术相比有显著的改进.

文献[52]提出了一种双注意力图卷积神经网络(DA-GCN)来检测智能合约的漏洞. 该模型利用智能合约的 CFG 提取特征，考虑到 CFG 的基本块级别无法获取某些细粒度的信息，进一步按照基本块出现的顺序构建操作码序列. DA-GCN 由图卷积模块、双注意力模块和分类模块组成. 图卷积模块将 CFG 作为输入，提取 CFG 级别的特征；双注意力模块分别提取 CFG 级别和操作码级别的注意力系数，赋予与漏洞最相关的部分更大的权重；最

后在分类模块中将图卷积模块和双注意力模块中提取的特征进行拼接,送入分类器中进行漏洞检测.实验表明,DA-GCN 在重入和时间戳依赖两种漏洞的检测上具有很好的检测性能.

文献[53]提出了一种用于智能合约漏洞检测的基于图嵌入的字节码匹配方法.该方法将与漏洞相关的数据分为四类.第一类是 CALLDATALOAD、CALLER、CALLVALUE 等指令调用的交易数据,第二类是 BLOCKHASH、TIMESTAMP 等指令调用的区块数据,第三类是 SLOAD 指令调用的存储数据,第四类是 CALL 等指令的调用返回值.该方法模拟执行操作码构造控制流图,同时跟踪上述四类数据的数据流和控制流,结合 REVERT、INVALID 等异常终止指令,对操作码序列进行切分,得到操作码序列切片.由于 Solidity 编译器版本的不同,相同的源代码可以转换为具有不同的 DUP/SWAP 指令的操作码序列.为了减少这种差异,在切片中忽略了这些指令.根据上述的切分方法,得到每一个目标智能合约的切片和已知漏洞的切片.针对每一个切片,将每一个操作码看作一个图节点,利用图嵌入方法 Graph2Vec[54]将切片转换成向量表示,然后计算候选切片与已知漏洞切片的余弦距离来测量切片间的相似性,值越高.说明候选切片有漏洞的可能性越大.

文献[55]提出了一种基于关键数据流图和预训练模型的智能合约漏洞检测框架 Peculiar.该框架利用修改后的 tree-sitter 工具将智能合约源代码解析为 AST,在 AST 中识别变量序列,然后根据变量之间的数据流向关系构数据流图 DFG,从 DFG 提取与关键变量相关的数据节点构成的子图即 CDFG,最后将 CDFG 作为输入对 GraphCodeBert[56]模型进行预训练,结合下游分类模型进行漏洞检测.

文献[57]开发了第一个系统的、模块化的基于深度学习的智能合约漏洞检测框架 DeeSCVHunter.该框架包含了 8 个主流的深度学习模型和 3 个嵌入方案.该框架提出了漏洞候选切片(VCS)的概念,首先在智能合约源代码中匹配与漏洞相关的候选语句(CS),如 call.value()、block.timestamp 等,然后根据控制流和数据流匹配与候选语句相关的语句构成漏洞候选切片,使用嵌入方案将其转换为向量,作为深度学习模型的输入进行训练.

EtherGIS[58]是一种利用图神经网络和专家知识从智能合约控制流图 CFG 中提取图特征的漏洞检测框架.EtherGIS 结合专家知识,通过分析多种漏洞触发机制,构建敏感 EVM 指令语料库.根据语料库确定 CFG 中节点的特征,同时根据控制流的触发类型确定边的特征,最终生成相应的属性图.使用 GNN 聚合整个图的属性和结构特征,作为分类模型的输入进行漏洞检测.此外,采用 AutoML 来自动化整个深度学习的优化过程.实验结果表明,EtherGIS 可以有效地检测智能合约中的漏洞.

上述智能合约漏洞检测方法将静态分析与深度学习相结合,使用静态分析方法提取智能合约代码的结构信息,包括控制流图 CFG、抽象语法树 AST 和数据流图 DFG 等,通过深度学习方法特别是图神经网络将结构特征映射到向量空间,进一步提高了智能合约漏洞的检测效果.

4.3　基于图像处理的漏洞检测

智能合约的一些漏洞只出现在一个函数中.基于这一发现,文献[59]提出了切片矩阵的概念.将“RETURN”操作码作为函数的分割点,对合约操作码进行分割.然后提取每个片段的操作码特征,并将其组合成一个切片矩阵,每个矩阵的大小为 $82 * 75$.为了进行比较分析,创建了三个模型,分别是基于操作码特征的神经网络(Neural Network Based on Opcode Feature,NNBOOF)、基于切片矩阵的卷积神经网络(Convolution Neural Network Based on Slice Matrix,CNNBOSM)和基于操作码特征的随机森林模型(Random Forest Based on Opcode Feature,RFBOOF).实验表明,在 RFBOOF 的表现最好,但 CNNBOSM 的表现好于 NNDOOF.

文献[60]提出了一种基于卷积神经网络的轻量级智能合约漏洞检测模型 SC-VDM.EVM 中的操作码是用两位的十六进制数表示的,所以将两位字节码分成一组来分割字节码序列.然后用字节码组成一个矩阵,并将每一组字节码转换为十进制,使每一组数据都在 0～255 的范围内,从而将矩阵转换成一个灰度图.得到灰度图后,将其输入到 CNN 模型中进行训练.由于不同合约的大小不同,因此得到的灰度图大小也是不一样的,导致 CNN 模型中的全连接层会得到不同大小的特征图.SC-VDM 在全连接层前加入了一个 SPP 层将不同特征图转换为统一的大小.SC-VDM 在检测重入漏洞上表现最好,且检测时间大大缩短,每个智能合

约仅需 0.021 s.

CodeNet[61]是一种新的 CNN 体系架构，在保持智能合约语义和上下文的同时对智能合约进行漏洞检测. 为了生成 CNN 体系架构的输入，预处理阶段将固定的字节码序列转换为 RGB 图像，字节码中每三个字节被映射成 RGB 像素. 为了保留字节码的语义和上下文信息，CodeNet 将二维的 RGB 图像修改成一维的形式. 由于带有步幅的卷积操作会丢失像素级的特征，即破坏智能合约的语义和上下文信息，CodeNet 使用非步幅卷积操作，同时采用深度可分离卷积缓解非步幅卷积带来的计算量和参数增加的问题.

上述智能合约漏洞检测方法从图像处理的角度将智能合约进行向量化，通过将智能合约转化为二维图像，使用图像处理方法学习智能合约的图像模式. 然而进行图像转换的方式一般都比较简单直接，缺乏可解释性，还需要做进一步的研究.

5　挑战与未来研究方向

5.1　数据集

神经网络模型的训练主要依赖于数据集. 目前提出的基于深度神经网络的智能合约漏洞检测模型大多数都是在自己构建的数据集上进行评估，导致不同模型之间的指标无法准确地进行比较. 因此迫切需要一个统一的数据集作为智能合约漏洞检测模型的评估基准.

现阶段已经有一些公开的智能合约数据集. Ghaleb 等[62]开源了包含 7 种不同漏洞的 9369 个智能合约构建的数据集. 但是这些智能合约是通过对正常智能合约进行漏洞注入而得到的，可能会与区块链上真实的漏洞智能合约有所不同. Durieux 等[43]构建了一个名为 SmartBugs 的数据集. 该数据集包含两部分，一部分包含 69 个智能合约和 115 个标记的漏洞，另一部分包含以太坊上 47 398 份唯一的智能合约，并且使用 9 个静态分析工具进行了漏洞标记. 然而由于静态分析工具存在误报，所以无法保证智能合约的标签真实性. 此外，类不平衡的问题也会影响模型的训练和评估.

5.2　基于动态特征

现阶段的智能合约漏洞检测模型大多基于源代码、字节码序列、CFG 和 AST 等静态特征进行学习和检测. 然而静态分析获得的特征是有限的，智能合约漏洞大多是在运行时发生的. 如何将智能合约的运行时动态特征结合起来扩展模型的特征表示空间，应该成为未来的一个研究方向. SODA[63]是一个在线智能合约漏洞检测框架，通过对 EVM 虚拟机进行代码插桩，收集智能合约运行时的相关信息，结合预定义的规则分析智能合约是否包含漏洞. 可以将此类工具收集到的智能合约运行时信息作为动态特征帮助模型更好地学习智能合约漏洞的特征表示.

此外，对区块链上产生的交易信息进行分析，可以深入挖掘智能合约与用户间的相互关系[64]，并用于智能合约漏洞检测和溯源[65]. Chen 等[66]设计并实现了以太坊数据收集系统 DataEther，成功获取区块、交易、合约等信息用于进一步分析. 可以利用此类工具构建智能合约相关交易信息数据集，使用深度学习方法结合智能合约交易信息提取漏洞相关行为模式进行漏洞检测.

5.3　模型可解释性

神经网络模型是典型的黑盒模型，模型进行检测的依据和原因对人们来说是未知的. 现有的智能合约漏洞检测研究，并没有将精力放在解释模型的检测行为上. 无法理解模型如何检测智能合约漏洞的原因会导致模型的实用性受到质疑，无法保证模型检测结果的可靠性.

注意力机制可以作为解释模型的一种手段. 借助注意力机制，模型可以选择与检测目标相关性较高的部分. 如源代码中与漏洞相关的关键代码片段、CFG 中与漏洞相关的节点和子图等. 结合注意力机制为可解释的基于深度学习的智能合约漏洞检测提供了研究方向.

6　结　论

随着人工智能技术的兴起以及在各个领域的研究应用，大量基于深度学习的智能合约漏洞检测模型涌现. 本文对近年来提出的智能合约漏洞检测模型进行了详细地介绍，总结了现有基于深度学习的智能合约漏洞检测的研究进展，并提出了该领域面临的挑战和未来可能的研究方向.

参考文献：

[1] Huang K, Mu Y, Rezaeibagha F, *et al*. Design and analysis of cryptographic algorithms in blockchain [M]. Boca Raton: CRC Press, 2021.

[2] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system [EB/OL]. [2022-10-11]. https://www.debr. io/article/21260-bitcoin-a-peer-to-peer-electron-

ic-cash-system, 2008.

[3] Buterin V. A next-generation smart contract and decentralized application platform[EB/OL]. [2022-10-11]. https://finpedia.vn/wp-content/uploads/2022/02/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf, 2014.

[4] Chen J, Xia X, Lo D, *et al*. Defining smart contract defects on ethereum[J]. IEEE T Software Eng, 2020, 48: 327.

[5] Li X, Jiang P, Chen T, *et al*. A survey on the security of blockchain systems[J]. Future Gener Comp Sy, 2020, 107: 841.

[6] Chen T, Li X, Wang Y, *et al*. An adaptive gas cost mechanism for ethereum to defend against underpriced dos attacks[C]//Proceedings of the International Conference on Information Security Practice and Experience. Cham, Switzerland: Springer, Cham, 2017.

[7] 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. Vienna Austria: SIGSAC, 2016: 254.

[8] Kalra S, Goel S, Dhawan M, *et al*. Zeus: analyzing safety of smart contracts[C]//Ndss. San Diego: [s. n.],2018: 1.

[9] Tsankov P, Dan A, Drachsler-Cohen D, *et al*. Securify: Practical security analysis of smart contracts[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. Toronto Canada: SIGSAC, 2018: 67.

[10] Chen T, Li Z, Zhang Y, *et al*. A large-scale empirical study on control flow identification of smart contracts[C]//Proceedings of the 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). Porto de Galinhas, Brazil: IEEE, 2019: 1.

[11] Bhargavan K, Delignat-Lavaud A, Fournet C, *et al*. Formal verification of smart contracts: Short paper[C]//Proceedings of the 2016 ACM workshop on programming languages and analysis for security. Vienna Austria: SIGSAC, 2016: 91.

[12] Yang Z, Lei H. Fether: An extensible definitional interpreter for smart-contract verifications in coq[J]. IEEE Access, 2019, 7: 37770.

[13] Jiang B, Liu Y, Chan W K. Contractfuzzer: fuzzing smart contracts for vulnerability detection[C]// Proceedings of the 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). Montpellier, France: IEEE, 2018: 259.

[14] Chen T, Zhang Y, Li Z, *et al*. Tokenscope: Automatically detecting inconsistent behaviors of cryptocurrency tokens in ethereum[C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. London United Kingdom: SIGSAC, 2019: 1503.

[15] Zeng P, Lin G, Pan L, *et al*. Software vulnerability analysis and discovery using deep learning techniques: a survey[J]. IEEE Access, 2020, 8: 197158.

[16] Szabo N. Formalizing and securing relationships on public networks[J]. First Monday, 1997, 2: 1.

[17] Dannen C. Introducing Ethereum and solidity[M]. Berkeley: Apress, 2017.

[18] 张小松. 区块链安全技术与应用[M]. 北京: 科学出版社, 2021.

[19] Dune. Ethereum smart contracts creation[DB/OL]. [2022-11-02]. https://dune.com/queries/1712220.

[20] Top DeFi Tokens by market capitalization[DB/OL]. [2022-11-10]. https://coinmarketcap.com/view/defi/.

[21] Shae Z, Tsai J J P. On the design of a blockchain platform for clinical trial and precision medicine[C]//Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). Atlanta: IEEE, 2017: 1972.

[22] Zhang P, White J, Schmidt D C, *et al*. FHIRChain: applying blockchain to securely and scalably share clinical data[J]. Comput Struct Biotec, 2018, 16: 267.

[23] Kim H M, Laskowski M. Toward an ontology-driven blockchain design for supply-chain provenance[J]. Intell Syst Account, 2018, 25: 18.

[24] Figorilli S, Antonucci F, Costa C, *et al*. A blockchain implementation prototype for the electronic open source traceability of wood along the whole supply chain[J]. Sensors, 2018, 18: 3133.

[25] Guan Z, Si G, Zhang X, *et al*. Privacy-preserving and efficient aggregation based on blockchain for power grid communications in smart communities[J]. IEEE Commun Mag, 2018, 56: 82.

[26] Huang K, Zhang X, Mu Y, *et al*. Building redactable consortium blockchain for industrial Internet-of-Things[J]. IEEE T Ind Inform, 2019,

15：3670.

[27] DAO. The DAO (organization)[EB/OL]. [2022-10-24]. https://en.w-ikipedia.org/wiki/The_DAO_(organization).

[28] Breidenbach L, Daian P, Juels A, *et al*. An In-Depth look at the parity Multisig Bug[EB/OL]. [2022-10-24]. https://hackingdistributed.com/2017/07/22/deep-dive-parity-bug.

[29] p0n1. A disastrous vulnerability found in smart contracts of BeautyChain(BEC)[EB/OL]. [2022-10-24]. https://medium.com/secbit-media/a-disastrous-vulnerability-found-in-smart-contracts-of-beautychain-bec-dbf24ddbc30e.

[30] Allen i. 币安遭黑客洗劫 7000 比特币，价值约 4100 万美金[EB/OL]. [2022-10-24]. https://www.freebuf.com/news/203024.html.

[31] 观网财经. 币安旗下区块链项目被盗[EB/OL]. [2022-10-24]. https://www.36kr.com/p/1950114960196228.

[32] 谭粤飞. 2021 年区块链安全生态报告：80%DAPP 安全事故缘于智能合约漏洞[EB/OL]. [2022-10-24]. https://new.qq.com/rain/a/20220209A0620600.

[33] Tann W J W, Han X J, Gupta S S, *et al*. Towards safer smart contracts: A sequence learning approach to detecting security threats[EB/OL]. [2022-09-05]. https://arxiv.org/abs/1811.06632.

[34] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Comput, 1997, 9: 1735.

[35] Gogineni A K, Swayamjyoti S, Sahoo D, *et al*. Multi-class classification of vulnerabilities in smart contracts using AWD-LSTM, with pre-trained encoder inspired from natural language processing [J]. IOP SciNotes, 2020, 1: 035002.

[36] Merity S, Keskar N S, Socher R. Regularizing and optimizing LSTM language models [C]//Proceedings of International Conference on Learning Representations. Vancouver: [S. n.], 2018.

[37] Lutz O, Chen H, Fereidooni H, *et al*. ESCORT: ethereum smart contracts vulnerability detection using deep neural network and transfer learning [EB/OL]. [2022-09-15]. https://arxiv.org/abs/2103.12607.

[38] Chung J, Gulcehre C, Cho K H, *et al*. Empirical evaluation of gated recurrent neural networks on sequence modeling [EB/OL]. [2022-09-15]. https://arxiv.org/abs/1412.3555.

[39] Huang J, Zhou K, Xiong A, *et al*. Smart contract vulnerability detection model based on multi-task learning [J]. Sensors, 2022, 22: 1829.

[40] Zhang L, Chen W, Wang W, *et al*. CBGRU: a detection method of smart contract vulnerability based on a hybrid model [J]. Sensors, 2022, 22: 3577.

[41] Mikolov T, Chen K, Corrado G, *et al*. Efficient estimation of word representations in vector space [EB/OL]. [2022-09-16]. https://arxiv.org/abs/1301.3781.

[42] Bojanowski P, Grave E, Joulin A, *et al*. Enriching word vectors with subword information [J]. Trans Assoc Comput Linguist, 2017, 5: 135.

[43] Durieux T, Ferreira J F, Abreu R, *et al*. Empirical review of automated analysis tools on 47, 587 Ethereum smart contracts[C]//Proceedings of the ACM/IEEE 42nd International conference on software engineering. Seoul South Korea: SIGSOFT, 2020: 530.

[44] Ashizawa N, Yanai N, Cruz J P, *et al*. Eth2Vec: learning contract-wide code representations for vulnerability detection on ethereum smart contracts [C]//Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure. Hong Kong, China: SIGSAC, 2021: 47.

[45] Le Q, Mikolov T. Distributed representations of sentences and documents [C]//Proceedings of the International Conference on Machine Learning. PMLR, 2014: 1188.

[46] Zhang L, Wang J, Wang W, *et al*. A novel smart contract vulnerability detection method based on information graph and ensemble learning [J]. Sensors, 2022, 22: 3581.

[47] Vaswani A, Shazeer N, Parmar N, *et al*. Attention is all you need[C]//Proceedings of Advances in Neural Information Processing Systems. [S. l.]: Curran Associates, Inc, 2017.

[48] Mi F, Wang Z, Zhao C, *et al*. VSCL: automating vulnerability detection in smart contracts with deep learning [C]//Proceedings of the 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). Sydney: IEEE, 2021: 1.

[49] Zhuang Y, Liu Z, Qian P, *et al*. Smart contract vulnerability detection using graph neural network [C]// Proceedings of the International Joint Conference on Artificial Intelligence. Yokohama: [s. n.], 2020: 3283.

[50] Liu Z, Qian P, Wang X, *et al*. Smart contract vulnerability detection: from pure neural network to interpretable graph feature and expert pattern fusion

[EB/OL]. [2022-09-16]. https://arxiv.org/abs/2106.09282.

[51] Liu Z, Qian P, Wang X, *et al*. Combining graph neural networks with expert knowledge for smart contract vulnerability detection [J]. IEEE T Knowl Data En, 2021, 2021: 1.

[52] Fan Y, Shang S, Ding X. Smart contract vulnerability detection based on dual attention graph convolutional network [C]//Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing. Cham, Switzerland: Springer, Cham, 2021: 335.

[53] Huang J, Han S, You W, *et al*. Hunting vulnerable smart contracts via graph embedding based bytecode matching [J]. IEEE T Inf Foren Sec, 2021, 16: 2144.

[54] Narayanan A, Chandramohan M, Venkatesan R, *et al*. Graph2vec: learning distributed representations of graphs [C]// Proceedings of the 13th International Workshop on Mining and Learning with Graphs (MLG). Nova Scotia: [s. n.], 2017: 21.

[55] Wu H, Zhang Z, Wang S, *et al*. Peculiar: Smart contract vulnerability detection based on crucial data flow graph and pre-training techniques [C]//Proceedings of the 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE). Wuhan, China: IEEE, 2021: 378.

[56] Guo D, Ren S, Lu S, *et al*. Graphcodebert: Pre-training code representations with data flow [C]// Proceedings of International Conference on Learning Representations. Vienna: [s. n.], 2020.

[57] Yu X, Zhao H, Hou B, *et al*. DeeSCVHunter: a deep learning-based framework for smart contract vulnerability detection [C]//Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN). Shenzhen: IEEE, 2021: 1.

[58] Zeng Q, He J, Zhao G, *et al*. EtherGIS: a vulnerability detection framework for ethereum smart contracts based on graph learning features [C]//Proceedings of the 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC). Los Alamitos: IEEE, 2022: 1742.

[59] Xing C, Chen Z, Chen L, *et al*. A new scheme of vulnerability analysis in smart contract with machine learning [J]. Wirel Netw, 2020, 2020: 1.

[60] Zhou K, Cheng J, Li H, *et al*. SC-VDM: a lightweight smart contract vulnerability detection model [C]//Proceedings of the International Conference on Data Mining and Big Data. Guangzhou: Springer, Singapore, 2021: 138.

[61] Hwang S J, Choi S H, Shin J, *et al*. CodeNet: Code-targeted convolutional neural network architecture for smart contract vulnerability detection [J]. IEEE Access, 2022, 10: 32595.

[62] Ghaleb A, Pattabiraman K. How effective are smart contract analysis tools? evaluating smart contract static analysis tools using bug injection [C]// Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis. Virtual Event USA: SIGSOFT, 2020: 415.

[63] Chen T, Cao R, Li T, *et al*. SODA: A generic online detection framework for smart contracts [C]// NDSS. San Diego: [s. n.],2020.

[64] Chen T, Li Z, Zhu Y, *et al*. Understanding ethereum via graph analysis [J]. ACM T Internet Techn, 2020, 20: 1.

[65] Zhu H, Niu W, Liao X, *et al*. Attacker traceability on ethereum through graph analysis [J]. Secur Commun N, 2022, 2022: 3448950.

[66] Chen T, Li Z, Zhang Y, *et al*. Dataether: data exploration framework for ethereum [C]//Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS). Dallas: IEEE, 2019: 1369.

引用本文格式：

中　文：张小松，牛伟纳，黄世平，等. 基于深度学习的智能合约漏洞检测方法综述[J]. 四川大学学报：自然科学版，2023，60：020001.

英　文：Zhang X S, Niu W N, Huang S P, *et al*. A survey of smart contract vulnerability detection methods based on deep learning [J]. J Sichuan Univ: Nat Sci Ed, 2023, 60: 020001.