软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007318] [CSTR: 32375.14.jos.007318]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于代码控制流图的庞氏骗局合约检测*

黄 静[1,2], 王梦晓[1,2], 韩红桂[1,2]

[1](北京工业大学 信息学部, 北京 100124)

[2](计算智能与智能系统北京市重点实验室 (北京人工智能研究院), 北京 100124)

通信作者: 黄静, E-mail: huangjing@bjut.edu.cn

摘 要: 区块链在加密货币投资领域展现出强劲的生命力, 吸引了大量投资者的参与. 然而, 由于区块链的匿名性, 导致了许多欺诈行为, 其中庞氏骗局智能合约就是一种典型的欺诈性投资活动, 给投资者带来了巨大的经济损失. 因此, 对以太坊上的庞氏骗局合约进行检测变得尤为重要. 但是, 现有研究大都忽略了庞氏骗局合约源代码中的控制流信息. 为提取庞氏骗局合约更丰富的语义信息和结构信息, 提出一种基于代码控制流图的庞氏骗局合约检测模型. 首先, 该模型将获取的合约源代码构建成控制流图的形式. 然后, 使用 Word2Vec 算法提取了包括数据流信息和代码结构信息在内的关键特征. 考虑到每个智能合约的功能不同、代码篇幅差异明显, 导致提取的特征向量维度差异较大, 对不同智能合约生成的特征向量进行对齐操作, 使得所有的特征向量具有相同的维度, 便于之后处理. 其次, 利用基于图卷积和 Transformer 的特征学习模块, 引入多头注意力机制, 来学习节点特征的依赖关系. 最后, 使用多层感知机实现对庞氏骗局合约的识别. 通过在 Xblock 网站提供的数据集上将该模型与传统的图特征学习模型进行对比, 验证该模型引入的多头注意力机制的性能. 实验结果证明, 该模型有效地提升对庞氏骗局合约的检测能力.

关键词: 智能合约; 庞氏骗局; 控制流图; 图 Transformer

中图法分类号: TP311

中文引用格式: 黄静, 王梦晓, 韩红桂. 基于代码控制流图的庞氏骗局合约检测. 软件学报. http://www.jos.org.cn/1000-9825/7318.htm

英文引用格式: Huang J, Wang MX, Han HG. Ponzi Scheme Contract Detection Based on Code Control Flow Graph. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7318.htm

Ponzi Scheme Contract Detection Based on Code Control Flow Graph

HUANG Jing[1,2], WANG Meng-Xiao[1,2], HAN Hong-Gui[1,2]

[1](Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China)

[2](Beijing Key Laboratory of Computational Intelligence and Intelligence System (Beijing Institute of Artificial Intelligence), Beijing 100124, China)

Abstract: Blockchain has shown strong vitality in the field of cryptocurrency investment, attracting the participation of a large number of investors. However, due to the anonymity of blockchain, it induces a lot of fraud, among which the Ponzi scheme smart contract is a typical fraudulent investment activity, causing huge economic losses for investors. Therefore, the detection of Ponzi scheme contracts on Ethereum becomes particularly important. Nevertheless, most existing studies have ignored control flow information in the source code of Ponzi scheme contracts. To extract more semantic and structural information from Ponzi scheme contracts, this study proposes a Ponzi scheme contract detection model based on code control flow graph. First, the model constructs the obtained contract source code in the form of a control flow diagram. Then, key features including data flow information and code structure information are extracted by the Word2Vec algorithm. Considering that the functions of each smart contract are different and the length of the code varies significantly,

* 基金项目: 国家重点研发计划 (2022YFB3305802)
收稿时间: 2024-04-17; 修改时间: 2024-06-26; 采用时间: 2024-10-28; jos 在线出版时间: 2025-03-26

resulting in a large difference in the extracted feature vectors. In this study, feature vectors generated by different smart contracts are aligned so that all feature vectors have the same dimension, which is convenient for subsequent processing. Secondly, the feature learning module based on graph convolution and Transformer is utilized to introduce multi-head attention mechanism to learn the dependency of node features. Finally, the multilayer perceptron is used to identify the Ponzi scheme contract. By comparing the proposed model with the traditional graph feature learning model on the dataset provided by the Xblock website, the performance of the multi-head attention mechanism introduced by the model is verified. Experimental results demonstrate that this model effectively improves the ability to detect Ponzi scheme contracts.

Key words: smart contract; Ponzi scheme; control flow graph; graph Transformer

区块链 (blockchain) 由比特币发展而来, 虽然世界对比特币的态度褒贬不一, 但作为比特币底层技术之一的区块链技术受到了广泛的关注[1]. 区块链本质上是一个共享数据库, 用来存储去中心化的分布式数据, 其主要特点是由一系列数据块按照一定顺序组成的链式结构[2]. 以太坊 (Ethereum) 是一个建立在区块链技术之上的去中心化应用平台, 被称为区块链 2.0. 它于 2015 年创建[3], 旨在推动区块链技术的发展, 使其不仅局限于数字货币交易, 而是能够支持更广泛的应用场景. 与比特币等加密货币不同, 以太坊提供了一个图灵完备的以太坊虚拟机 (EVM), 它可以执行任意算法复杂度的代码, 并允许用户通过几行代码创建区块链应用程序, 这使得以太坊成为一个功能更加强大和灵活的区块链平台.

通过以太坊, 用户可以轻松地使用几行代码创建智能合约. 智能合约是一种在区块链上执行的自动化协议, 无需中间人的干预即可执行. 智能合约的概念最初由 Szabo 在 20 世纪 90 年代提出, 他将其定义为: “通过结合协议与用户界面, 规范和保障计算机网络安全的工具”[4]. 在智能合约中, 预先编写的合约条款会在满足特定条件时自动执行, 这使得交易和协议能够自动化执行, 从而降低了交易成本、提高了安全性, 并且减少了诸如欺诈等问题的可能性[5]. 传统的合约通常需要受信任的第三方进行监督, 这既费时又费力, 并且会消耗额外的成本. 相比之下, 智能合约存储于分布式区块链中, 实现了安全、高效的点对点交易, 无需依赖可信赖的权威机构[6].

智能合约的广泛运用带来了许多重要的改变. 在金融服务领域, 智能合约可以用于自动化交易、资产管理和支付系统, 从而提高了交易的效率和安全性[7]. 在供应链管理方面, 智能合约可以跟踪产品的流向和状态, 确保生产和物流的透明度和可追溯性. 在投资领域, 智能合约可以用于创建各种投资工具和金融衍生品, 从而扩大了投资者的选择空间. 在游戏和身份验证方面, 智能合约可以用于构建数字身份和安全访问系统, 保护用户的隐私和数据安全. 随着技术的不断发展和普及, 智能合约将在更多领域发挥重要作用, 推动着数字化经济和社会的进步.

区块链技术正在改变人们的生活和工作方式, 为人们带来了更加便捷、高效和安全的交易方式. 随着区块链的日益普及, 已经吸引了越来越多的用户参与, 普通投资者对区块链以及数字货币的兴趣越来越大[8]. 然而, 由于这项新兴技术缺乏监管, 许多不法分子利用区块链具有匿名性、不可篡改, 智能合约自动执行、可信度高等优势进行宣传来获取虚假的信任, 发起了一系列诈骗活动[9], 其中最具代表性的就是以太坊庞氏骗局[10].

庞氏骗局是传统金融投资领域的一种典型骗局[11], 其运行特点如图 1 所示, 新加入的用户作为金字塔拓扑结构的下一层级用户, 其投资的钱大部分都作为投资回报支付给最初的投资者[12]. 庞氏骗局的发起者作为金字塔顶层用户获得了巨额的收益, 而底层的用户只会失去他们的投资. 庞氏骗局金字塔型的投资者结构以及欺骗拉拢下线投资者的传销方式使大量的投资者蒙受了巨大的经济损失, 它在许多国家已经被明令禁止. 然而, 与传统的庞氏骗局不同的是, 现在不法分子将庞氏骗局引入区块链, 利用智能合约进行伪装, 这给区块链用户带来了极为昂贵的损失, 也严重危害了区块链生态系统的安全[13].

Vasek 等人[14]对现有的智能合约骗局进行了实证分析, 发现在 192 个骗局中有 13 000 名潜在受害者损失了大约 1 100 万美元. 一些大型骗局甚至会建立复杂的网站, 进行积极的营销活动来吸引投资者. 然而, 对区块链知之甚少的投资者很难分辨那些伪装成高收益投资计划的智能合约的真实面目[15]. 研究统计, 从 2013 年 9 月到 2014 年 9 月, 通过比特币运营的庞氏骗局已经筹集了 700 多万美元[14]. 根据区块链分析公司 Chainanalysis 发布的研究报告, 诈骗交易是加密货币中最大的犯罪类型, 2019 年诈骗者从数百万受害者那里骗取了价值约 43 亿美元的加密货币, 其中绝大部分都与庞氏骗局有关.

与传统金融投资领域的庞氏骗局相比, 区块链中的庞氏骗局智能合约具有更高的隐蔽性和欺骗性. 智能合约自动执行的特点使得投资者更容易受到欺骗, 因为他们相信合约会按照预先设定的规则自动执行, 而不会考虑到可能存在的欺诈行为[16]. 此外, 智能合约的不可篡改性质意味着一旦资金被转移, 就无法撤销, 投资者的损失也就无法挽回. 这种无法终止的合约执行特性使得投资者的风险不断增加. 由于智能合约的自动化执行和不可篡改性质, 不法分子可以通过虚假宣传和高额回报来吸引投资者, 而这些庞氏骗局合约的发起者往往可以隐藏在匿名的区块链背后, 使其更加难以追踪和制止[17].

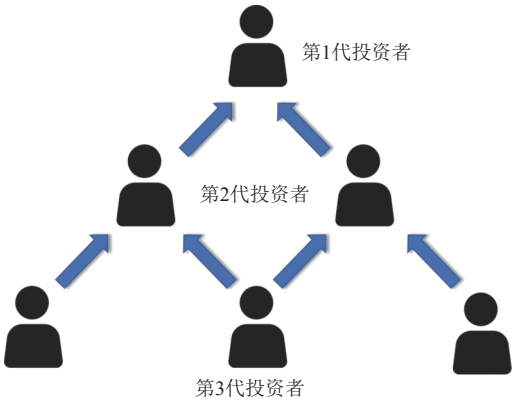

图 1　庞氏骗局的运行特点

庞氏骗局合约的出现不仅给个人投资者带来了经济损失, 也给整个区块链生态系统带来了负面影响. 这些欺诈活动损害了投资者对区块链技术的信心, 阻碍了区块链的发展和应用. 因此, 迫切需要研究出一种有效的方法来检测和防范这种智能合约中的欺诈行为. 庞氏骗局合约的检测尤为重要, 它可以帮助以太坊用户避免不必要的损失, 同时有助于维护智能合约的安全性, 促进区块链技术在各个领域的可持续发展.

现有研究大都以提取合约单个操作码的频率作为代码特征进行庞氏骗局合约的检测. 但是, 智能合约是一个有逻辑的程序, 合约代码反映着交易逻辑特征, 其中存在投资者进行投资和接收回报的交易机制, 因此现有研究忽略了合约代码中反映庞氏骗局交易特点的交易流信息. 为了深入地挖掘智能合约代码中的交易流特征, 本文提出了一种基于代码控制流图的庞氏骗局合约检测模型. 该模型将合约代码转换成图的形式, 以便提取其中的数据流信息和代码结构信息. 然而, 传统的图学习方法在处理复杂性日益增加的合约时可能存在一定的局限性, 因为它们难以充分利用丰富的上下文信息. 鉴于此, 本文受到 Transformer 在自然语言处理领域成功应用的启发, 将 Transformer 与图神经网络结合, 以应用于庞氏骗局合约的检测. 本文的主要贡献如下.

(1) 设计了一种基于源代码控制流图的方法, 它可以将合约源代码转换成控制流图的形式, 便于提取合约代码的控制流特征.

(2) 设计了一种基于多头注意力机制的模型, 将 Transformer 与图卷积网络 (graph convolutional network, GCN) 进行结合对得到合约控制流图的节点和边的特征进行学习, 捕捉庞氏骗局合约的重要特征.

(3) 所提模型在公开数据集进行了实验验证, 对比其他机器学习模型显著地提升了检测性能, 证实了本模型对于精确识别庞氏骗局合约的有效性.

本文第 1 节介绍本文模型所涉及的相关工作, 包括庞氏骗局合约检测的研究现状和智能合约的中间表示方法. 第 2 节详细地阐述了基于控制流图的庞氏骗局合约检测方法的模型架构. 第 3 节通过对比实验验证了所提模型的有效性. 最后总结全文.

1　相关工作

本节对庞氏骗局合约检测的相关技术研究现状进行梳理, 并介绍本文模型所用到的智能合约中间表示方法.

1.1 庞氏骗局合约检测

结合庞氏骗局和智能合约的特点, 本文从以下两个维度对庞氏骗局合约进行剖析: 一是通过对合约的代码进行分析, 挖掘合约的控制逻辑; 二是通过对合约的交易记录进行分析, 挖掘合约的交易特点. 目前, 对庞氏骗局合约检测的相关研究可以分为 3 类: 第 1 类是基于源代码检查的庞氏骗局检测方法, 该方法主要是通过手动检查合约源代码来检查识别庞氏骗局; 第 2 类是基于特征工程的庞氏骗局检测方法, 该方法主要是通过设计一组特征来对智能合约进行表示, 然后将其输入到机器学习模型中, 进而实现对庞氏骗局的检测, 不同的基于特征的方法之间的主要区别在于如何选取有效特征; 第 3 类是基于网络嵌入的庞氏骗局检测方法, 该方法主要是通过节点嵌入技术, 将交易网络的形成过程、智能合约的操作码等信息整合为一个低维连续向量, 然后将庞氏骗局合约的检测问题建模为一个节点分类任务. 下面对这 3 类方法分别进行介绍.

(1) 基于源代码检查的庞氏骗局合约检测方法

基于源代码检查的庞氏骗局合约检测方法主要是通过人工检查合约源代码来分析合约的控制逻辑. Chen 等人[18]通过分析庞氏骗局合约源代码的逻辑, 对 4 种类型的庞氏骗局合约字节码级别模式进行了分析. Bartoletti 等人[19]建立了一套将合约归类为庞氏骗局的标准, 通过手动分析开源代码来检索庞氏骗局合约, 并提出使用归一化的 Levenshtein 距离 (normalized Levenshtein distance, NLD)[20]比较合约相似性, 从而进一步挖掘出隐藏在非开源合约中的庞氏骗局.

这种方法的弊端在于只能找到与已知庞氏骗局合约字节码相似的庞氏骗局合约, 代码检查需要大量人力资源, 效率低下.

(2) 基于特征工程的庞氏骗局合约检测方法

随着机器学习技术和数据挖掘方法在该领域的应用, 提取合约特征用于机器学习模型的特征工程思路成为研究主流. Chen 等人[21]首先从智能合约的交易数据和操作码中提取账户特征和单个操作码的频率特征, 然后构建基于机器学习分类模型, 检测智能合约中潜在的庞氏骗局. Zhang 等人[22]创新性地增加了合约字节码特征, 然后使用改进的 LightGBM 识别庞氏骗局合约. Shen 等人[23]将合约字节码转换成高维矩阵, 并将庞氏骗局合约检测转换为异常检测问题. Zhang 等人[24]提出使用操作码的 2-gram 来更客观的反映相邻操作指令的关联. Wang 等人[25]同时考虑账户特征和代码特征, 提出一种将合成少数过采样技术 (synthetic minority over-sampling technique, SMOTE) 和长短期记忆网络 (long short-term memory, LSTM) 相结合的智能合约庞氏骗局检测方法 PSD-OL, 以解决庞氏骗局智能合约检测中存在的类不平衡问题. Sun 等人[26]通过行为森林来捕获智能合约在其交互过程中的动态特征, 可以实现对刚部署的庞氏骗局合约的检测. Jung 等人[27]应用数据挖掘技术来提取账户特征以及代码特征, 进而挖掘与庞氏骗局相关的比特币地址. Ibba 等人[28]将合约代码解析为抽象语法树 (abstract syntax tree, AST), 通过对 AST 进行解析来提取代码特征, 进而实现对庞氏骗局合约的检测. Huynh 等人[29]考虑到庞氏骗局合约的交易特点, 提出了仅依赖交易的新型检测模型, 并创新了与时间相关的特征提取方法, 提升了检测模型的鲁棒性. Wang 等人[30]也考虑了智能合约交易的时间序列信息, 并使用 LSTM 从庞氏骗局合约中学习时间特征进行模型训练. Zheng 等人[31]提出了多视图级联集成模型, 从多个视图提取庞氏骗局合约的特征, 包括字节码、语义以及开发人员特征, 提升了模型的性能和鲁棒性.

基于特征工程的庞氏骗局检测方法虽然可以较好地实现庞氏骗局合约自动检测, 但是仍存在数据集不平衡、选定的训练特征并未考虑到合约源代码的逻辑关系和模型的可移植性差等问题.

(3) 基于网络嵌入的庞氏骗局合约检测方法

基于交易网络嵌入的庞氏骗局检测方法主要是通过动态节点嵌入技术, 将交易的形成过程, 智能合约的操作码等整合为一个低维连续向量, 然后使用二元分类器来检测是否为庞氏骗局合约. Liang 等人[32]首次提出通过动态图嵌入实现智能合约庞氏骗局检测的工作, 提出了一种基于数据驱动的智能庞氏骗局检测系统 DSPSD, 根据账户的操作码和交易数据直接预测合约账户是否实现了庞氏骗局. Jin 等人[33]提出了一个通用异构特征增强模块来捕获账户行为模式的异构特征, 提高了庞氏骗局检测方法的性能. Cai 等人[34]针对特征工程方法中存在的语义特

征提取不充分的问题, 提出了一种用于智能合约函数表示的切片交易属性图, 以全面准确地捕获智能合约函数中的交易语义信息. Yu 等人[35]将庞氏骗局的识别和检测建模为节点分类任务, 通过人工抽取庞氏骗局合约的固有特征, 并将这些特征与交易网络的拓扑结构相结合, 提出一种基于图卷积网络的检测模型来精确区分庞氏骗局合约.

将智能合约交易建模为交易网络的方式可以很好地捕捉交易的时间特性. 但是, 现有的节点嵌入方法会丢失部分节点信息, 且不能很好地结合交易网络的节点特征和拓扑结构.

1.2 智能合约的中间表示

智能合约是一个有逻辑的程序, 合约代码反映着交易逻辑特征, 其中存在着投资者进行投资和接收回报的交易机制[36]. 因此, 对智能合约代码进行建模和分析, 以提取其中的控制流信息具有极其重要的意义. 在对智能合约源代码进行控制流分析的相关研究中, 常用的方法主要包括将智能合约源代码描述为序列或抽象语法树的形式[37].

序列表示将代码简单地视为一系列连续的标记 (如词法单元或语句), 按照它们在源代码中的顺序排列. 这种表示不考虑代码结构或语义, 只是将代码视为一串字符或标记的线性排列. 当代码被描述为序列时, 可以进一步使用处理序列的深度学习模型来学习智能合约的特征, 例如, Liu 等人[38]和 Hochreiter 等人[39]将合约代码表示为序列, 然后分别使用 *n*-gram 语言模型和 LSTM 网络处理合约序列, 进而实现对智能合约的漏洞检测. 基于序列的代码表示描述了智能合约语句的执行顺序, 但不能表征数据流、指令之间的调用关系以及语句的详细执行结构.

对此, 有研究[28]将智能合约表示为抽象语法树, 将代码表示为树状结构, 其中每个节点是代码中出现的一个结构, 这种表示方式捕捉了代码的语法结构和组织方式. 抽象语法树去除了具体的细节, 如括号、分号等, 并以树的形式展现代码的层次结构和语义信息. 例如, Xu 等人[40]将合约源代码表示为抽象语法树的形式, 然后使用 *k* 最近邻方法 (*k* nearest neighbor, KNN) 对抽象语法树特征进行提取, 实现了对智能合约漏洞的检测. Wang 等人[41]使用一种语言识别工具 ANTLR 将智能合约的功能片段解析为抽象语法树, 以提取智能合约的漏洞特征. 基于树的代码表示可以描述代码的语法信息, 但仍然不能明确地描述语句的执行顺序, 并且缺乏数据流的信息.

现有研究[42]表明, 程序可以转换为符号图表示, 这能够保持程序元素之间的语义关系, 如数据依赖关系和控制依赖关系. 例如, Liu 等人[43]将源代码丰富的控制流和数据流语义转换为合约图, 并设计了一个节点消除过程来规范化合约图并突出图中的重要节点, 实现了对 3 种智能合约漏洞类型的高效识别. Yan 等人[44]将恶意软件程序构建为控制流图 (control flow graph, CFG) 的方形式, 克服了现有恶意软件分类方法中非自适应图匹配技术的缺点, 实现了对恶意软件的高效分类. Cai 等人[45]结合抽象语法树、控制流图、程序依赖图 (program dependency graph, PDG), 构建了具有语法和语义特征的智能合约的图表示, 使得模型对智能合约漏洞检测的准确率高达 89.2%. Wen 等人[46]基于有向图的可视化方法来显示投资和奖励流程以及关键任务的执行路径, 从而轻松识别以太坊上的庞氏骗局合约.

受此启发, 本文将智能合约源代码转换成图的形式来实现对庞氏骗局合约的检测. 具体来说, 本文使用基于控制流图的方法来表征代码之间的数据依赖和控制依赖关系, 将合约代码的执行路径表示为有向图的形式. 控制流图由节点和边组成, 其中节点代表基本块 (一组顺序执行的语句), 边代表控制流转移 (如条件分支、循环等)[47]. 控制流图可以更形象直观地描述代码执行的路径和控制流程, 便于反映庞氏骗局合约的代码特征, 从而更有利于提高对庞氏骗局合约的检测效果.

2 方 法

本文设计了一种基于代码控制流图的庞氏骗局合约检测模型, 该模型包括智能合约特征提取、特征对齐、特征学习和合约分类 4 个模块.

模型的总体架构如图 2 所示. 首先, 将智能合约的源代码转换成控制流图的结构, 然后提取图结构的节点特征和边特征. 其次, 考虑到不同合约生成的控制流图结构并不相同, 导致生成的特征维度有很大差别, 本文加入了特征对齐模块. 然后, 利用 Transformer 与 GCN 对得到的节点和边的特征进行学习, 捕捉庞氏骗局合约的特征. 最后,

利用多层感知机 (multilayer perceptron, MLP) 对合约样本进行分类, 以实现对庞氏骗局合约的检测和识别.

2.1 控制流图构建

智能合约是一个由 Solidity 代码编写的程序, 不同代码块之间可能会存在调用与被调用的关系, 由此带来数据流的转移. 如图 3 所示, 这是一个合约代码的片段, 该合约片段通过调用不同的函数实现了以太币的存入、撤回、回退等一系列功能. 当智能合约执行时, 会自动自上而下地运行, 这意味着每次交易只有一个调用入口. 但是函数功能的调用会发生控制流的改变. 本文考虑到合约代码数据流转移的这一特性, 将合约源代码划分成多个基本块, 然后根据控制流的方向, 在不同基本块之间添加边, 以图形的方式还原合约的运行细节.

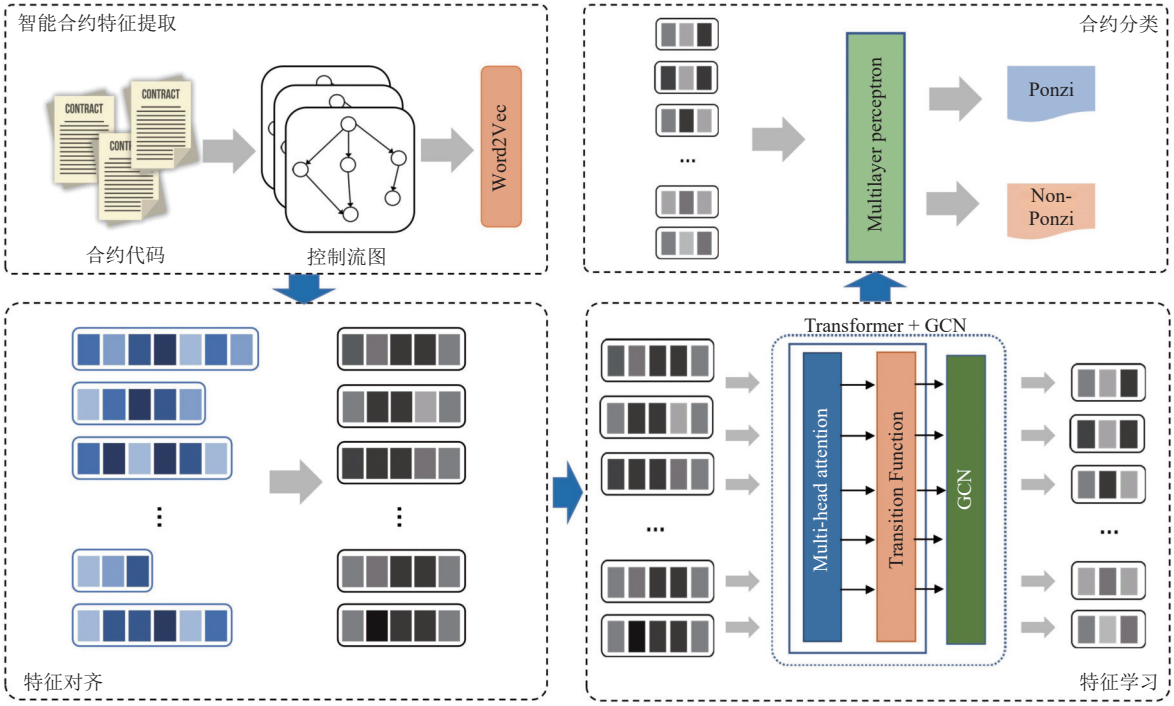

图 2 模型总体结构图

图 3 合约代码示例

为了深入挖掘合约代码的运行逻辑, 本文根据 Chen 等人[21]数据集. 从 Etherscan 网站爬取了每一个合约的源代码, 并将每个合约的源代码都转换成控制流图的形式. 控制流图是一种图形表示, 能够清晰地展示代码中的控制结构和执行路径, 有助于理解代码的逻辑结构和可能的执行流程. 具体实现过程如下.

(1) 代码解析与语法分析

首先需要将智能合约的编程语言 Solidity 转化为计算机能够理解的抽象语法树. 该过程主要是通过语法分析器将代码字符串分解成的词法单元序列转换为树状结构, 可以更好地反映合约源代码的层次和语法结构. AST 是程序的抽象表示, 树状结构的节点代表代码的各个部分, 如函数声明、表达式、语句等, 树状结构的边则表示这些节点之间的关系, 如语法结构的父子关系、节点之间的操作依赖关系、程序执行的控制流转移等.

(2) 构建基本块

在对智能合约进行语法分析的基础上, 将代码划分为基本块, 每一个代码块即一组连续的语句, 在执行的过程中不存在跳转或分支. 每个基本块只能从第 1 条语句进入并从最后一条语句离开, 即只有一个入口点和一个出口点, 这是基本块的特征. 构建基本块的目的是将代码分割成更小、易于处理的模块, 为后续的分析和图形的构建奠定基础.

(3) 建立控制流图

根据基本块之间的控制关系, 构建控制流图. 控制流图由节点和边组成, 其中每个节点对应一个基本块, 每条边对应基本块之间的控制转移路径, 从而形成一个有向图结构. 例如, 条件语句会引入分支, 循环语句会引入循环体, 这些都会在控制流图中体现出来.

控制流图的边不仅可以表示控制转移关系, 还可以表示数据流的传递. 通过对节点之间的数据流进行分析, 可以追踪合约中变量在程序执行过程中的使用和赋值情况, 帮助理解智能合约源代码的交易行为和潜在的安全问题. 图 4 展示了一个由智能合约生成的控制流图的局部结构, 其中每个节点代表一组连续的语句, 而节点之间的连线则表示代码块之间的数据流转移路径. 通过这个图可以直观地观察到各个代码模块之间的控制流转移情况.

庞氏骗局合约交易存在的显著特点是前期投资者的回报往往来自后期加入的投资者的资金, 因此, 控制流图可以很好地捕捉这一数据流特点, 从而有助于识别潜在的庞氏骗局合约.

2.2 特征提取与对齐

(1) 特征提取

Word2Vec 是自然语言处理中一种流行的词嵌入算法, 可以将单词映射到低维的向量表示, 并且还会保留单词之间的语义关系. 本文将基本块视为“单词”, 控制流图视为“文本”, 然后利用 Word2Vec 算法将基本块的结构信息转换为向量表示. 具体实现过程如图 5 所示.

首先, 需要遍历控制流图中的每一个基本块, 收集其中的指令和操作数. 将收集到的基本块内容按照控制流图中的顺序排列, 形成一个文本序列. 然后, 将生成的文本序列作为输入, 使用 Word2Vec 模型进行训练, 得到每个基本块的词向量表示. 通过对图中的节点进行嵌入, 将节点表示为低维度的向量, 以捕捉节点之间的相似性和关联性. 本文使用的是 Word2Vec 模型中的 Skip-gram 模型, 该模型主要是通过对给定的中心词来预测上下文词的概率. 这里, 对于给定的中心词 ω_c, 上下文词 ω_o 的概率可以通过如下公式计算:

$$P(\omega_o|\omega_c)=\frac{\exp(v_{\omega_o}^{\mathrm{T}}v_{\omega_c})}{\sum_{\omega\in V}\exp(v_{\omega}^{\mathrm{T}}v_{\omega_c})} \tag{1}$$

其中, v_{ω_o} 和 v_{ω_c} 分别表示上下文词 ω_o 和中心词 ω_c 的向量表示. Skip-gram 模型是通过最大化预测正确上下文词的概率来学习单词的向量表示.

对于一个给定的控制流图, 可以将节点 (基本块) 看作是单词, 基于节点序列训练 Skip-gram 模型, 学习每个节点的向量表示.

通过上述模型训练, 可以得到每个基本块对应的向量表示, 这些基本块向量捕捉了基本块在控制流图中的语义信息. 但是, 这些原始向量可能过于庞大, 不太适合后续任务的直接应用. 因此, 需要对这些基本块向量进行降维

处理, 将高维的基本块词向量映射到低维空间, 从而获得低维的特征向量表示. 处理后的特征向量具有更强的表征能力, 同时更容易用于后续任务的处理.

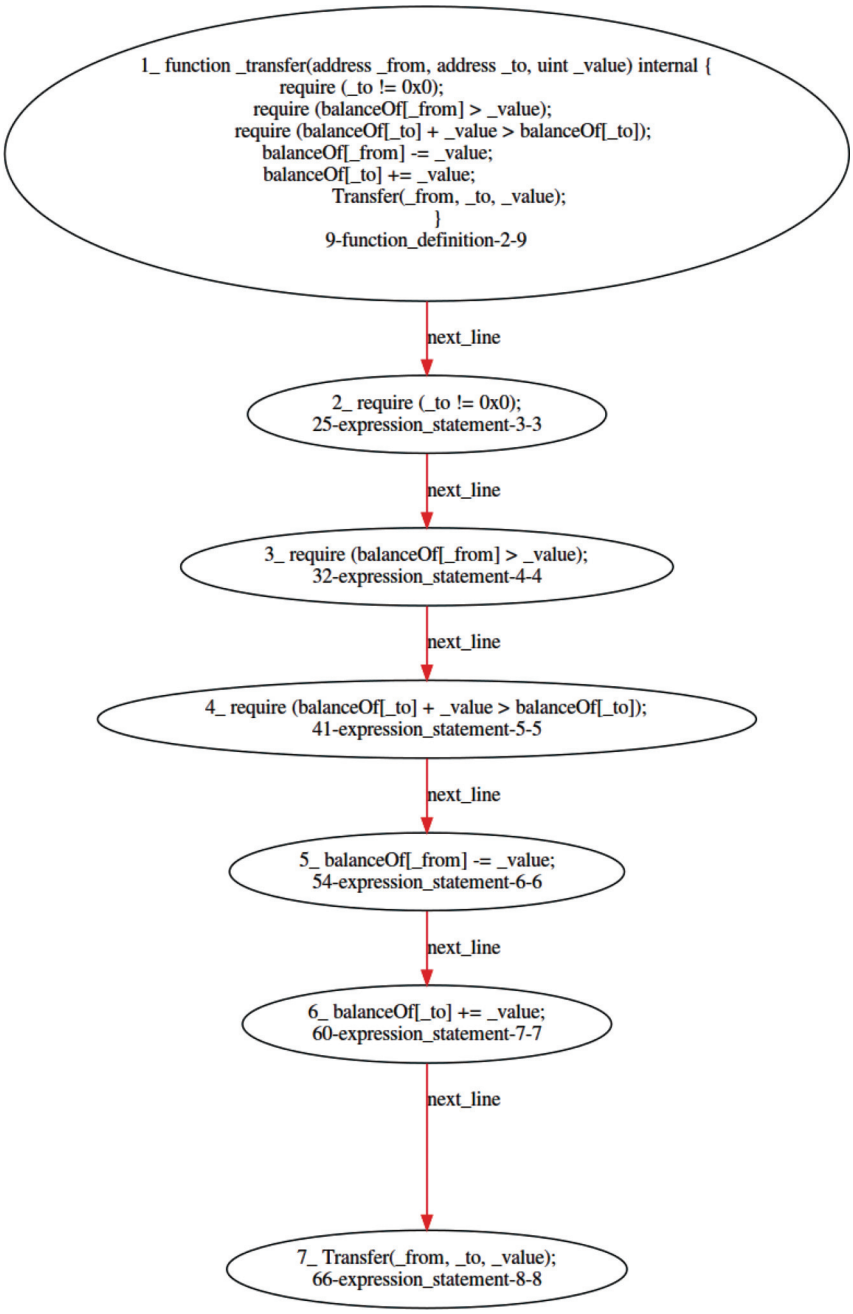

图 4　合约代码控制流图的局部结构示例

图 5　Word2Vec 特征提取过程

概括来说, 本文利用 Word2Vec 算法对控制流图进行了特征提取. 具体地, 本文将每个控制流图节点的特征表示为向量, 并将每个控制流图边的特征描述为两个相邻节点的平均特征. 通过这种方式, Word2Vec 生成的节点表示包含了图的部分拓扑信息, 其中融入了一部分语义信息. Word2Vec 模型的运用实现了控制流图到特征向量的转换, 这些特征向量捕捉了控制流图基本块之间的语义关系和上下文信息, 为后续的图分析任务提供了有力支持.

(2) 特征对齐

因为每个智能合约的功能不同, 所以每个合约源代篇幅长短有很大差别, 自然生成的控制流图也会大小各异, 以及由此带来上述 Word2Vec 模型提取的特征向量维度不同的问题. 特征向量维度不同会给后续的分析和模型训练带来困难, 因此, 本文对不同智能合约生成的特征向量进行了对齐, 使得所有的特征向量具有相同的维度.

具体来说, 本文通过遍历所有的特征向量, 确定对齐后的目标维度. 接下来, 对每个控制流图的特征向量进行对齐操作, 将其调整到统一的维度. 其中, 对于维度较小的特性向量, 使用零向量进行填充; 对于维度较大的特征向量, 根据向量分布进行截断. 最终实现特征向量维度的统一, 方便后续对特征数据的分析以及对模型的训练.

(3) 特征表示

为了简单起见, 本文将构建的控制流图定义为 $G=(V,E)$, 其中, $V=\{v_1,\ldots,v_n\}$ 表示控制流图的节点集, $E=\{e_1,\ldots,e_n\}$ 表示控制流边集.

2.3 基于图卷积和 Transformer 的特征学习模块

在给定合约特征图 G 的基础上, 本节将进行进一步的模型处理, 旨在通过节点和边的特征学习识别庞氏骗局的关键特征. Transformer[48]是一种基于自注意力机制的模型, 适用于处理序列数据, 能够有效地捕捉序列中的全局依赖关系, 而且在训练和推理时可以并行化处理. GCN 适用于处理图结构数据, 能够利用节点之间的连接关系进行特征学习, 通过聚合节点邻居的信息来更新节点的特征表示, 从而实现对图结构的特征学习. 为了捕捉合约特征中更细致的语义和结构关系, 本文将 Transformer 与 GCN 进行结合, 具体架构如图 6 所示. 在本模块引入了 Transformer 中的多头注意力机制[49], 多头注意力机制能捕捉更复杂的节点特征依赖关系, 而不需要遍历整个序列. 同时, 为了丰富这种节点依赖, 还引入了图神经网络. 最终设计出一个基于多头注意力机制的图卷积神经网络模块, 充分利用了图结构数据的局部连接信息和自注意力机制的全局关系学习能力, 用于学习合约代码控制流图的隐藏特征. 具体来说, GCN 用于学习节点特征表示, 并且能够捕捉图结构中的局部特征; 而 Transformer 则能够在全局范围内建模节点之间的依赖关系. 这一创新点使得模型能够充分利用 GCN 和 Transformer 的优势, 更好地学习到图结构数据中的特征表示, 并且能够更准确地捕捉节点之间的复杂关系, 从而提高庞氏骗局合约的检测效果.

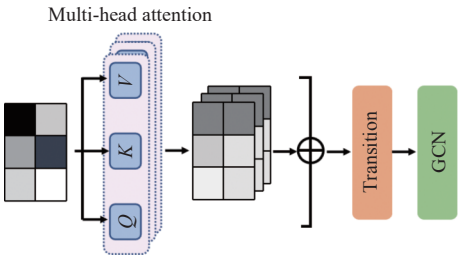

图 6 特征学习模块的结构图

在第 2.2 节的特征对齐与提取模块得到了合约的特征表示. 在这些特征中, 每个节点 v 都带有独特的嵌入特征, 这些特征能够深入表达合约的语义信息. 为了更准确地捕获上下文之间的语义联系, 本文采用自注意力机制来有效聚合节点特征. Transformer 中的多头注意力机制描述如下:

$$Attention(Q,K,V)=Softmax\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V \tag{2}$$

其中, $Q^{(k)}\in R^{d\times d}$, $K^{(k)}\in R^{d\times d}$ 分别是 Query 权重矩阵和 Key 权重矩阵, $V^{(k)}\in R^{d\times d}$ 是值投影权重矩阵, $\sqrt{d_k}$ 是缩放因子. 公式表示通过权重有针对性的捕获特征, 大大简化了神经网络的复杂度.

通过利用 Transformer 构建堆叠在一起的多个层, 这个过程可以用如下公式表示:

$$x_{t,v}^{(k)}=Normal(h_{t-1,v}^{(k)}+ATT(h_{t-1,v}^{(k)})) \tag{3}$$

$$h_{t,v}^{(k)}=Normal(x_{t,v}^{(k)}+Trans(x_{t,v}^{(k)})) \tag{4}$$

其中, $h_{t-1,v}^{(k)}$ 表示第 k 个隐藏层 v 的特征向量, *Normal* 表示层归一化操作. 这里的 *Trans* 和 *ATT* 分别表示多层感知机网络和自注意力层.

权重矩阵在所有位置和时间步长上的自注意力参数和转换函数 Transition 都是共享的. 特别的是, 在第 k 层, 对于给定节点 $v\in V$, 在每个时间步 t 中, 本文引入基于 Transformer 的函数来聚合所有节点的向量表示. 注意力层特征提取的过程如下式:

$$ATT(h_{t-1,v}^{(k)})=\sum\alpha_v^{(k)}(V^{(k)}h_{t-1,v}^{(k)}) \tag{5}$$

其中, $V^{(k)}\in R^{d\times d}$ 是值投影权重矩阵, $\alpha_v^{(k)}$ 是注意力权重, 它的具体计算方法如下:

$$\alpha_v^{(k)}=Softmax\left(\frac{(Q^kh_{t-1,v}^{(k)})^{\mathrm{T}}(K^{(k)}h_{t-1,v}^{(k)})}{\sqrt{d}}\right) \tag{6}$$

完整的注意力层操作可以用如下公式表示:

$$H_t^{(k)}=Attention\left(H_{t-1}^{(k)}Q^{(k)},H_{t-1}^{(k)}K^{(k)},H_{t-1}^{(k)}V^{(k)}\right) \tag{7}$$

为了充分利用智能合约中节点之间的依赖关系, 本文采用图卷积网络 (GCN) 来聚合节点之间传递的消息, 这样可以更有效地学习节点表示, 即通过公式 (8) 来构建相应函数.

$$H^{(k+1)}=GCN\left(A,H^{(k)}\right) \tag{8}$$

其中, H^k 是第 k 层数据的隐藏特征, A 是邻接矩阵.

图卷积网络和 Transformer 的这种结合使得模型能够在合约图中有效地学习到庞氏骗局的关键特征, 并提高了对庞氏骗局合约的识别性能.

2.4 合约分类模块

在实现对庞氏骗局合约分类的过程中, 本文使用多层感知机学习上述图 Transformer 模型提取的高维特征之间的非线性关系, 并将这些特征映射到低维空间, 以实现合约的二分类. MLP 是一种常见的人工神经网络结构, 由一个或多个全连接的隐藏层和一个输出层组成.

具体实现过程如图 7 所示, MLP 的输入是从图 Transformer 模型中提取的高维特征, 这些特征是对智能合约控制流图特征向量的嵌入表示. 这些特征向量被送入 MLP 的输入层. 随着特征向量在隐藏层之间的传递, MLP 学习到特征之间的复杂关系. 每个隐藏层由一组神经元组成, 每个神经元都与上一层的所有神经元相连, 通过权重连接进行信息传递. 在隐藏层中, 每个神经元接收来自前一层的输入, 并对其进行加权和求和后通过激活函数处理, 以产生输出.

在经过隐藏层的一系列非线性变换后, MLP 将特征映射到一个更低维的空间. 最后, MLP 的输出层采用 *Sigmoid* 函数作为激活函数, 如公式 (9) 所示, 将输出值映射到 [0, 1] 的范围内, 实现对合约的二分类. *Sigmoid* 函数能够将输入值映射到概率值, 表示合约属于正例 (庞氏骗局) 的概率. 最终, 基于输出概率的阈值设定, 可以对合约进行分类, 即将其归类为庞氏骗局合约或正常合约.

$$Sigmoid(x)=\frac{1}{1+\mathrm{e}^{-x}} \tag{9}$$

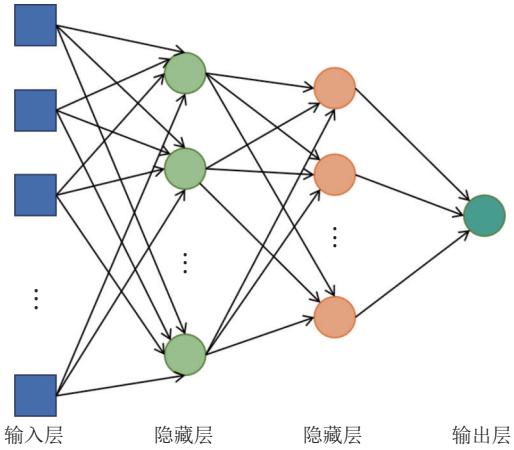

图 7　MLP 分类过程

3　实验分析

3.1　实验数据集

本文实验采用了 Chen 等人使用的数据集[21], 该数据集可以从 XBlock 网站获取, 其中包含 3 580 个非庞氏骗局合约和 200 个庞氏骗局合约的地址. 根据数据集中的合约地址, 获取相应智能合约的源代码, 然后对这些源代码进行控制流图的构建, 从而提取其中的控制流信息. 为了方便后续的模型训练, 本文首先对提取的控制流图特征向量进行对齐, 将所有合约的特征向量调整到相同的维度. 因此, 本文数据集中每一个合约都对应一个形状相同的二维特征矩阵.

本文数据集进行随机划分, 将其中 80% 作为训练集, 10% 作为验证集, 以验证每个回合 (Epoch) 后模型的性能, 其余 10% 作为测试集.

3.2　评价指标

为了准确评估模型的性能, 并便于与其他具有代表性的庞氏骗局检测方法进行比较, 本文选择了精确率、召回率和 $F1$ 分数作为评估指标. 精确率 (*Precision*) 是指在所有被判定为庞氏骗局的合约中, 真正的庞氏骗局合约所占的比例. 召回率 (*Recall*) 是在所有庞氏骗局中检测到的庞氏骗局合约的比例. $F1$ 分数 ($F1$-*score*) 是一种结合了准确率和召回率的混合度量方式. 准确率 (*Accuracy*) 表示正确分类的合约样本数占总合约样本数的比例. 这些指标的具体公式如下所示:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{10}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{11}$$

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{12}$$

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Negative} \tag{13}$$

3.3　特征学习模型的对比实验

为了验证本文提出图 Transformer 训练模型在对图结构特征学习方面的效果, 本文将所提出的模型与 5 种常用的传统方法进行了比较, 这些模型主要包括 RNN、LSTM、GRU、CNN、Transformer.

RNN (recurrent neural network)[50]: 适用于处理序列数据的神经网络结构, 能够捕捉序列中的时间依赖关系.

CNN (convolutional neural network)[51]: 主要用于图像处理的神经网络结构, 通过卷积操作提取局部特征并利

用池化操作降低数据维度.

GRU (gated recurrent unit)[52]: 类似于 LSTM 的一种 RNN 变体, 具有更简单的结构, 但仍然能够有效地处理序列数据.

Transformer[48]: 基于自注意力机制的神经网络结构, 适用于处理序列数据, 在自然语言处理领域中取得了显著的成果.

实验结果如表 1 所示.

表 1 不同特征学习模型的对比实验

模型	*Accuracy*	*Precision*	*Recall*	*F*1-*score*
RNN	0.7273	0.7563	0.6458	0.6967
LSTM	0.7576	0.5941	0.8636	0.7039
GRU	0.7879	0.6061	0.6667	0.6350
CNN	0.8182	0.6417	0.7778	0.7032
Transformer	0.8025	0.8167	0.7418	0.7775
本文模型	**0.8930**	**0.8674**	**0.8667**	**0.8670**

从表 1 可以看出, 在庞氏骗局合约检测任务中, RNN、LSTM 和 GRU 模型表现最差, 其准确率、精确率、召回率和 *F*1 分数均较低. CNN 和 Transformer 在一定程度上提升了性能, 但是效果仍旧不够理想, 召回率和 *F*1 分数这两个指标均未达到 80%. 而本文提出的图 Transformer 模型在庞氏骗局合约检测任务中表现最佳, 具有更高的检测性能, 其中准确率达到了 89.3%. 这说明将 GCN 与 Transformer 相结合能够充分利用图结构数据的局部连接信息和全局关系学习能力, 从而提高庞氏骗局合约的检测效果.

3.4 参数敏感性实验

本节对图 Transformer 模块 GCN 层数对检测结果的影响进行了实验验证, 对应的实验结果如图 8 所示. 如图所示, 随着 GCN 层数的增加, 模型对庞氏骗局合约的检测性能不断提升. 这说明增加 GCN 层数可以帮助模型更好地学习和理解数据之间的复杂关系, 从而提高模型的性能. 然而, 需要注意的是, 随着 GCN 层数的增加, 模型的复杂性和计算量也会增加, 可能会导致过拟合或训练时间增加. 因此, 在选择 GCN 层数时需要权衡性能和计算效率, 本文的模型选择了 3 层 GCN 网络, 这既考虑到了性能的提升, 也考虑到了计算效率的问题.

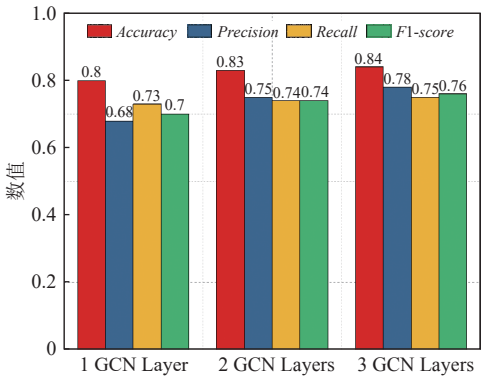

图 8 不同 GCN 层数的性能指标

3.5 注意力机制讨论实验

为了验证本文模型在特征学习方面的性能, 本节对 Transformer 注意力机制进行了实验验证. 图 9 展示了引入不同注意力机制对实验性能的影响. 具体来说, 图中对比了以下 4 种情况: 不引入注意力机制 (None-Att)、引入节点注意力机制 (Node-Att)、引入边注意力机制 (Edge-Att), 以及引入 Transformer 的多头注意力机制 (本模型) 的实

验结果.

其中, 节点注意力机制通过计算当前节点与其邻居节点之间的注意力权重, 并根据这些权重来聚合邻居节点的信息. 边注意力机制不仅考虑节点之间的关系, 还考虑了图中的边, 通过关联边的权重来调节节点之间的信息传递. 而本文所使用的 Transformer 多头注意力机制可以在序列中建立全局依赖关系, 以便更好地捕捉序列中的长距离依赖. 同时, 通过引入多个注意力头, 每个注意力头都会学习不同的注意力权重, 从而允许模型同时关注序列中不同位置的不同特征.

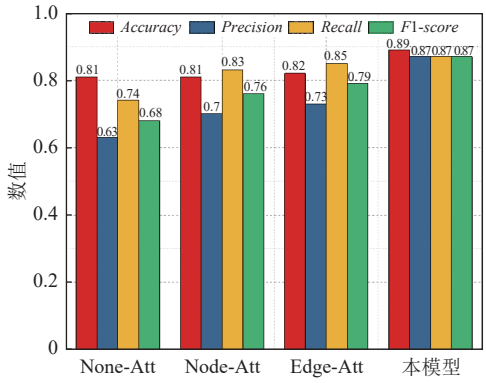

图 9　注意力机制对模型性能的影响

如图 9 所示, 不引入任何注意力机制可能导致模型无法充分捕捉数据之间的关系和重要特征, 从而降低了模型的性能. 引入节点注意力机制有助于模型更加关注节点之间的重要关系和特征, 从而提高了模型对庞氏骗局合约的检测能力. 进一步地, 引入边注意力机制可以进一步提升模型的性能. 边注意力机制使模型能够在节点之间的边上进行加权, 更细致地捕捉到节点之间的关联性和重要性, 从而进一步增强了模型的分类能力. 而本文所采用的 Transformer 多头注意力机制进一步增强了模型的表征学习能力. 通过多头注意力机制, 模型可以并行地学习多组注意力权重, 从不同的角度关注数据的特征和关系, 从而更全面地挖掘数据的信息. 这种机制有助于提高模型的鲁棒性和泛化能力, 使其能够更好地识别和预测庞氏骗局合约.

综上所述, 上述实验结果验证了本文所提出的图 Transformer 模型的有效性, 证明了引入注意力机制对于提升模型性能的重要性.

3.6　不同分类器的对比实验

为了验证本文所使用的分类模型在特征学习方面性能, 本节设置了多层感知机 MLP (multilayer perceptron) 与支持向量机 SVM (support vector machine)、随机森林 RF (random forest) 和自适应提升算法 AdaBoost (adaptive boosting) 分类器的对比实验. 实验中, 本节使用特征学习模块得到的控制流图的特征表示作为输入数据, 然后分别使用上述 4 个分类器对智能合约进行分类. 实验结果如表 2 所示.

表 2　不同分类器对比实验

模型	*Accuracy*	*Precision*	*Recall*	*F*1-*score*
SVM	0.5938	0.6667	0.1429	0.2354
AdaBoost	0.7500	0.8000	0.5714	0.6666
RF	0.8125	0.8333	0.7128	0.7684
MLP	**0.8930**	**0.8674**	**0.8667**	**0.8670**

从上述实验结果可以看出, SVM 对合约特征的处理效果最差, 其中 *F*1 分数仅达到了 23.53%. 另外两个集成学习方法 AdaBoost 和 RF 对合约的处理效果得到了提升, 但仍有提升的空间. 本文使用的 MLP 分类器在对合约的特征表示进行降维处理时的性能最好.

3.7 模型对比实验

为了验证本文所提出的模型在庞氏骗局合约检测方面的性能, 本节将所提出的模型与目前具有代表性的庞氏骗局检测模型进行了比较, 主要包括交易网络嵌入模型、基于账户特征模型[21](以下简称 Account)、基于代码特征模型[21](以下简称 Code) 及混合特征模型[21](以下简称 Combination). 几种方法简单介绍如下.

基于交易网络嵌入的庞氏骗局合约检测模型将获取的合约交易记录建模为时间加权多维有向图, 采用基于时间游走的网络嵌入方法提取交易网络中的隐含特征, 最后采用随机森林分类器完成庞氏骗局合约检测. 基于账户特征的检测模型从用户账户中提取交易特征, 结合不同分类器来检测庞氏骗局合约. 基于代码特征的检测模型以智能合约操作码特征作为其特征表达, 结合不同分类器来检测庞氏骗局合约. 混合特征模型则将账户特征与代码特征融合以完成庞氏骗局合约检测.

对应的实验结果如表 3 所示.

表 3　不同模型性能对比实验

模型	*Precision*	*Recall*	*F*1-*score*
交易网络嵌入模型	0.77	0.85	0.81
Account+IF (isolation forest, 孤立森林)	0.04	0.09	0.06
Account+SVM	0.32	0.06	0.09
Account+DT (decision tree, 决策树)	0.58	0.64	0.60
Account+XGBoost (eXtreme gradient boosting, 极端梯度提升)	0.59	0.22	0.32
Account+RF	0.64	0.20	0.30
Code+IF	0.03	0.06	0.04
Code+SVM	0.95	0.43	0.59
Code+DT	0.64	0.73	0.68
Code+XGBoost	0.91	0.73	0.81
Code+RF	0.94	0.73	0.82
Combination+IF	0.02	0.05	0.04
Combination+SVM	0.91	0.16	0.27
Combination+DT	0.31	0.24	0.27
Combination+XGBoost	0.90	0.67	0.76
Combination+RF	**0.95**	0.69	0.79
本模型	0.87	**0.87**	**0.87**

从实验结果上看, 本方法在检测精度、召回率及 *F*1 分数上全面超过了交易网络嵌入模型及基于账户特征模型, 说明对于庞氏骗局合约检测问题, 精准的代码特征提取是提升检测性能的关键. 对比基于代码特征模型及混合特征模型, 本方法在召回率及 *F*1 分数上超过了所有对比方法. 特别是 *F*1 分数, 本方法比次优方法高出 5 个百分点, 优势明显, 说明本方法兼顾了庞氏骗局合约的查全及查准, 具有优异的检测性能.

4 总　结

本文提出了一种基于代码控制流图的庞氏骗局合约检测模型, 旨在充分挖掘智能合约源代码中的控制流信息, 从而有效检测庞氏骗局合约. 该模型引入了基于多头注意力机制的 Transformer 模型, 并结合了图卷积网络, 以学习智能合约的控制流特征. 经实验验证, 该模型在庞氏骗局合约的检测准确率达到了 0.893, *F*1 分数达到了 0.857, 优于传统的特征学习模型. 这表明模型中引入的多头注意力机制对庞氏骗局合约的检测具有显著的提升效果.

References:

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. 2008. https://pdos.csail.mit.edu/6.824/papers/bitcoin.pdf

[2] Jiao T, Shen DR, Nie TZ, Kou Y, Li XH, Yu G. BlockchainDB: Querable and Immutable database. Ruan Jian Xue Bao/Journal of

Software, 2019, 30(9): 2671–2685 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5776.htm [doi: 10.13328/j.cnki.jos.005776]

[3] Wood G. Ethereum: A secure decentralised generalised transaction ledger. 2014. https://mholende.win.tue.nl/seminar/references/ethereum_yellowpaper.pdf

[4] Szabo N. Smart contracts: Building blocks for digital markets. 1996. https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html

[5] Raskin M. The law and legality of smart contracts. Georgetown Law Technology Review, 2016, 1(2): 305–341.

[6] Zheng ZB, Xie SA, Dai HN, Chen WL, Chen XP, Weng J, Imran M. An overview on smart contracts: Challenges, advances and platforms. Future Generation Computer Systems, 2020, 105: 475–491. [doi: 10.1016/j.future.2019.12.019]

[7] Guo HQ, Yu XJ. A survey on blockchain technology and its security. Blockchain: Research and Applications, 2022, 3(2): 100067. [doi: 10.1016/j.bcra.2022.100067]

[8] Chen WL, Zheng ZB. Blockchain data analysis: A review of status, trends and challenges. Journal of Computer Research and Development, 2018, 55(9): 1853–1870 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2018.20180127]

[9] Hewa TM, Hu YN, Liyanage M, Kanhare SS, Ylianttila M. Survey on blockchain-based smart contracts: Technical aspects and future research. IEEE Access, 2021, 9: 87643–87662. [doi: 10.1109/ACCESS.2021.3068178]

[10] Hu HW, Bai QL, Xu YD. SCSGuard: Deep scam detection for Ethereum smart contracts. In: Proc. of the 2022 IEEE Conf. on Computer Communications Workshops. New York: IEEE, 2022. 1–6. [doi: 10.1109/INFOCOMWKSHPS54753.2022.9798296]

[11] Hidajat T, Primiana I, Rahman S, Febrian E. Why are people trapped in Ponzi and pyramid schemes? Journal of Financial Crime, 2020, 28(1): 187–203. [doi: 10.1108/JFC-05-2020-0093]

[12] Frankel T. The Ponzi Scheme Puzzle: A History and Analysis of Con Artists and Victims. New York: Oxford University Press, 2012. [doi: 10.1093/acprof:osobl/9780199926619.001.0001]

[13] Vasek M, Moore T. Analyzing the bitcoin Ponzi scheme ecosystem. In: Proc. of the 2018 Int'l Workshops on Financial Cryptography and Data Security. Nieuwpoort: Springer, 2019. 101–112. [doi: 10.1007/978-3-662-58820-8_8]

[14] Vasek M, Moore T. There's no free lunch, even using bitcoin: Tracking the popularity and profits of virtual currency scams. In: Proc. of the 19th Int'l Conf. Financial Cryptography and Data Security. San Juan: Springer, 2015. 44–61. [doi: 10.1007/978-3-662-47854-7_4]

[15] Moore T, Han J, Clayton R. The postmodern Ponzi scheme: Empirical analysis of high-yield investment programs. In: Proc. of the 16th Int'l Conf. on Financial Cryptography and Data Security. Kralendijk: Springer, 2012. 41–56. [doi: 10.1007/978-3-642-32946-3_4]

[16] Hock B, Button M. Why do people join pyramid schemes? Journal of Financial Crime, 2023, 30(5): 1130–1139. [doi: 10.1108/JFC-09-2022-0225]

[17] Song LL, Kong XH. A study on characteristics and identification of smart Ponzi schemes. IEEE Access, 2022, 10: 57299–57308. [doi: 10.1109/ACCESS.2022.3178747]

[18] Chen WM, Li XR, Sui YT, He NY, Wang HY, Wu L, Luo XP. Sadponzi: Detecting and characterizing Ponzi schemes in Ethereum smart contracts. Proc. of the ACM on Measurement and Analysis of Computing Systems, 2021, 5(2): 26. [doi: 10.1145/3460093]

[19] Bartoletti M, Carta S, Cimoli T, Saia R. Dissecting Ponzi schemes on Ethereum: Identification, analysis, and impact. Future Generation Computer Systems, 2020, 102: 259–277. [doi: 10.1016/j.future.2019.08.014]

[20] Liu YJ, Liu B. A normalized Levenshtein distance metric. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2007, 29(6): 1091–1095. [doi: 10.1109/TPAMI.2007.1078]

[21] Chen WL, Zheng ZB, Ngai ECH, Zheng PL, Zhou YR. Exploiting blockchain data to detect smart Ponzi schemes on Ethereum. IEEE Access, 2019, 7: 37575–37586. [doi: 10.1109/ACCESS.2019.2905769]

[22] Zhang YM, Yu WQ, Li ZY, Raza S, Cao HH. Detecting Ethereum Ponzi schemes based on improved LightGBM algorithm. IEEE Trans. on Computational Social Systems, 2022, 9(2): 624–637. [doi: 10.1109/TCSS.2021.3088145]

[23] Shen XJ, Jiang SM, Zhang L. Mining bytecode features of smart contracts to detect Ponzi scheme on blockchain. Computer Modeling in Engineering and Sciences, 2021, 127(3): 1069–1085. [doi: 10.32604/cmes.2021.015736]

[24] Zhang YM, Kang SQ, Dai W, Chen SP, Zhu ZM. Code will speak: Early detection of Ponzi smart contracts on Ethereum. In: Proc. of the 2021 IEEE Int'l Conf. on Services Computing. Chicago: IEEE, 2021. 301–308. [doi: 10.1109/SCC53864.2021.00043]

[25] Wang L, Cheng H, Zheng ZB, Yang AJ, Zhu XH. Ponzi scheme detection via oversampling-based long short-term memory for smart contracts. Knowledge-based Systems, 2021, 228: 107312. [doi: 10.1016/j.knosys.2021.107312]

[26] Sun WS, Xu GY, Yang ZJ, Chen ZY. Early detection of smart Ponzi scheme contracts based on behavior forest similarity. In: Proc. of the 20th IEEE Int'l Conf. on Software Quality, Reliability and Security. Macao: IEEE, 2020. 297–309. [doi: 10.1109/QRS51102.2020.00047]

[27] Jung E, Tilly ML, Gehani A, Ge YJ. Data mining-based Ethereum fraud detection. In: Proc. of the 2019 IEEE Int'l Conf. on Blockchain.

Atlanta: IEEE, 2019. 266–273. [doi: 10.1109/Blockchain.2019.00042]

[28] Ibba G, Pierro GA, Di Francesco M. Evaluating machine-learning techniques for detecting smart Ponzi schemes. In: Proc. of the 4th IEEE/ACM Int'l Workshop on Emerging Trends in Software Engineering for Blockchain. Madrid: IEEE, 2021. 34–40. [doi: 10.1109/WETSEB52558.2021.00012]

[29] Huynh PD, Dau SH, Li XD, Luong P, Viterbo E. Improving robustness and accuracy of Ponzi scheme detection on Ethereum using time-dependent features. arXiv:2308.16391, 2023.

[30] Wang L, Cheng H, Zheng ZB, Yang AJ, Xu M. Temporal transaction information-aware Ponzi scheme detection for Ethereum smart contracts. Engineering Applications of Artificial Intelligence, 2023, 126: 107022. [doi: 10.1016/j.engappai.2023.107022]

[31] Zheng ZB, Chen WL, Zhong ZJ, Chen ZG, Lu YT. Securing the Ethereum from smart Ponzi schemes: Identification using static features. ACM Trans. on Software Engineering and Methodology, 2023, 32(5): 130. [doi: 10.1145/3571847]

[32] Liang YZ, Wu WJ, Lei K, Wang FY. Data-driven smart Ponzi scheme detection. arXiv:2108.09305, 2021.

[33] Jin CX, Jin J, Zhou JJ, Wu JJ, Xuan Q. Heterogeneous feature augmentation for Ponzi detection in Ethereum. IEEE Trans. on Circuits and Systems II: Express Briefs, 2022, 69(9): 3919–3923. [doi: 10.1109/TCSII.2022.3177898]

[34] Cai J, Li B, Zhang JL, Sun XB. Ponzi scheme detection in smart contract via transaction semantic representation learning. IEEE Trans. on Reliability, 2024, 73(2): 1117–1131. [doi: 10.1109/TR.2023.3319318]

[35] Yu SQ, Jin J, Xie YY, Shen J, Xuan Q. Ponzi scheme detection in Ethereum transaction network. In: Proc. of the 3rd Int'l Conf. Blockchain and Trustworthy Systems. Guangzhou: Springer, 2021. 175–186. [doi: 10.1007/978-981-16-7993-3_14]

[36] Ben-Nun T, Jakobovits AS, Hoefler T. Neural code comprehension: A learnable representation of code semantics. In: Proc. of the 32nd Int'l Conf. on Neural Information Processing Systems. Montréal: Curran Associates Inc., 2018. 3589–3601.

[37] Chen D, Feng L, Fan YQ, Shang SY, Wei ZC. Smart contract vulnerability detection based on semantic graph and residual graph convolutional networks with edge attention. Journal of Systems and Software, 2023, 202: 111705. [doi: 10.1016/j.jss.2023.111705]

[38] Liu H, Liu C, Zhao WQ, Jiang Y, Sun JG. S-gram: Towards semantic-aware security auditing for Ethereum smart contracts. In: Proc. of the 33rd IEEE/ACM Int'l Conf. on Automated Software Engineering. Montpellier: IEEE, 2018. 814–819. [doi: 10.1145/3238147.3240728]

[39] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780. [doi: 10.1162/neco.1997.9.8.1735]

[40] Xu YJ, Hu GG, You L, Cao CT. A novel machine learning-based analysis model for smart contract vulnerability. Security and Communication Networks, 2021, 2021: 5798033. [doi: 10.1155/2021/5798033]

[41] Wang B, Chu HT, Zhang PC, Dong H. Smart contract vulnerability detection using code representation fusion. In: Proc. of the 28th Asia-Pacific Software Engineering Conf. Taipei: IEEE, 2021. 564–565. [doi: 10.1109/APSEC53868.2021.00069]

[42] Rossi RA, Zhou R, Ahmed NK. Deep inductive graph representation learning. IEEE Trans. on Knowledge and Data Engineering, 2020, 32(3): 438–452. [doi: 10.1109/tkde.2018.2878247]

[43] Liu ZG, Qian P, Wang XY, Zhuang Y, Qiu L, Wang X. Combining graph neural networks with expert knowledge for smart contract vulnerability detection. IEEE Trans. on Knowledge and Data Engineering, 2021, 35(2): 1296–1310. [doi: 10.1109/TKDE.2021.3095196]

[44] Yan JQ, Yan GH, Jin D. Classifying malware represented as control flow graphs using deep graph convolutional neural network. In: Proc. of the 49th Annual IEEE/IFIP Int'l Conf. on Dependable Systems and Networks. Portland: IEEE, 2019. 52–63. [doi: 10.1109/DSN.2019.00020]

[45] Cai J, Li B, Zhang JL, Sun XB, Chen B. Combine sliced joint graph with graph neural networks for smart contract vulnerability detection. Journal of Systems and Software, 2023, 195: 111550. [doi: 10.1016/j.jss.2022.111550]

[46] Wen XL, Yeo KS, Wang Y, Cheng L, Zhu FD, Zhu M. Code will tell: Visual identification of Ponzi schemes on Ethereum. In: Proc. of the Extended Abstracts of the 2023 CHI Conf. on Human Factors in Computing Systems. Hamburg: ACM, 2023. 70. [doi: 10.1145/3544549.3585861]

[47] Liao XC, Meng HY. Code vulnerability detection of programmable logic controller based on control flow slicing. Computer Integrated Manufacturing Systems, 2023: 1–13 (in Chinese with English abstract). https://link.cnki.net/urlid/11.5946.TP.20231212.1734.013 [doi: 10.13196/j.cims.2023.0428]

[48] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6000–6010.

[49] Nguyen DQ, Nguyen TD, Phung D. Universal graph Transformer self-attention networks. In: Proc. of the 2022 Web Conf. Lyon: ACM, 2022. 193–196. [doi: 10.1145/3487553.3524258]

[50] Mao ST, Sejdić E. A review of recurrent neural network-based methods in computational physiology. IEEE Trans. on Neural Networks and Learning Systems, 2023, 34(10): 6983–7003. [doi: 10.1109/TNNLS.2022.3145365]

[51] Li ZW, Liu F, Yang WJ, Peng SH, Zhou J. A survey of convolutional neural networks: Analysis, applications, and prospects. IEEE Trans. on Neural Networks and Learning Systems, 2022, 33(12): 6999–7019. [doi: 10.1109/TNNLS.2021.3084827]

[52] Shu WN, Cai K, Xiong NN. A short-term traffic flow prediction model based on an improved gate recurrent unit neural network. IEEE Trans. on Intelligent Transportation Systems, 2022, 23(9): 16654–16665. [doi: 10.1109/TITS.2021.3094659]

附中文参考文献:

[2] 焦通, 申德荣, 聂铁铮, 寇月, 李晓华, 于戈. 区块链数据库: 一种可查询且防篡改的数据库. 软件学报, 2019, 30(9): 2671–2685. http://www.jos.org.cn/1000-9825/5776.htm [doi: 10.13328/j.cnki.jos.005776]

[8] 陈伟利, 郑子彬. 区块链数据分析: 现状、趋势与挑战. 计算机研究与发展, 2018, 55(9): 1853–1870. [doi: 10.7544/issn1000-1239.2018.20180127]

[47] 廖雪超, 孟航宇. 基于控制流切片的可编程逻辑控制器代码漏洞检测方法. 计算机集成制造系统, 2023: 1–13. https://link.cnki.net/urlid/11.5946.TP.20231212.1734.013 [doi: 10.13196/j.cims.2023.0428]

黄静(1979－), 女, 博士, 副教授, 博士生导师, 主要研究领域为智能合约漏洞检测, 区块链欺诈行为检测.

韩红桂(1983－), 男, 博士, 教授, 博士生导师, 主要研究领域为人工神经网络, 智慧环保技术.

王梦晓(1998－), 女, 硕士生, 主要研究领域为区块链欺诈行为检测.