软件学报 ISSN 1000-9825, CODEN RUXUEW
2026,37(3):1413−1426 [doi: 10.13328/j.cnki.jos.007564] [CSTR: 32375.14.jos.007564]

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

融合 TextCNN 和对抗训练的以太坊庞氏骗局检测模型*

宁小勇[1], 高睿杰[2], 叶楚涵[2], 刘 园[3], 王兴伟[1], 黄 敏[4]

[1](东北大学 计算机科学与工程学院, 辽宁 沈阳 110819)

[2](东北大学 软件学院, 辽宁 沈阳 110819)

[3](广州大学 网络空间安全学院, 广东 广州 510006)

[4](东北大学 信息科学与工程学院, 辽宁 沈阳 110819)

通信作者: 刘园, E-mail: yuanliu@gzhu.edu.cn; 王兴伟, E-mail: wangxw@mail.neu.edu.cn

摘 要: 以太坊上智能合约的广泛部署为区块链生态系统注入了活力, 而智能合约的不可逆性和匿名性却给监管带来了巨大挑战. 不法分子趁机在以太坊部署庞氏骗局, 引发了严重的安全风险和经济损失. 因此, 迅速高效地检测庞氏骗局智能合约至关重要. 目前的庞氏骗局检测方法存在的主要挑战包括智能合约操作码行为特征被忽略, 特征提取不全面, 检测方法在受到对抗干扰时性能不稳定、准确率低等问题. 为克服这些不足, 提出一种融合 TextCNN 和对抗训练的以太坊庞氏骗局检测方法. 该方法通过静态分析智能合约操作码来提取智能合约的行为特征, 同时结合 Word2Vec 模型保留智能合约的语义信息, 确保了操作码特征的完整性. 与此同时, 还采用改进后的动态步长投影梯度下降算法训练 TextCNN 模型, 增强检测模型的鲁棒性, 提高检测准确率. 在 XBlock 数据集上展开实验, 实验结果表明, 所提方法在确保精确率和鲁棒性的同时, 召回率达到 98.36%, *F*1 分数达到 98.31%. 该方法重点关注智能合约操作码而不依赖交易特征, 能在智能合约部署时迅速、高效地检测出庞氏骗局智能合约.

关键词: 智能合约; 庞氏骗局; 行为序列; 对抗训练; TextCNN

中图法分类号: TP393

中文引用格式: 宁小勇, 高睿杰, 叶楚涵, 刘园, 王兴伟, 黄敏. 融合TextCNN和对抗训练的以太坊庞氏骗局检测模型. 软件学报, 2026, 37(3): 1413–1426. http://www.jos.org.cn/1000-9825/7564.htm

英文引用格式: Ning XY, Gao RJ, Ye CH, Liu Y, Wang XW, Huang M. Ethereum Ponzi Scheme Detection Model Combining TextCNN and Adversarial Training. Ruan Jian Xue Bao/Journal of Software, 2026, 37(3): 1413–1426 (in Chinese). http://www.jos.org.cn/1000-9825/7564.htm

Ethereum Ponzi Scheme Detection Model Combining TextCNN and Adversarial Training

NING Xiao-Yong[1], GAO Rui-Jie[2], YE Chu-Han[2], LIU Yuan[3], WANG Xing-Wei[1], HUANG Min[4]

[1](School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China)

[2](Software College, Northeastern University, Shenyang 110819, China)

[3](Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China)

[4](College of Information Science and Engineering, Northeastern University, Shenyang 110819, China)

Abstract: The widespread deployment of smart contracts on Ethereum has injected vitality into the blockchain ecosystem, while the irreversibility and anonymity of smart contracts have posed great challenges to supervision. Criminals take the opportunity to deploy Ponzi schemes on Ethereum, causing serious security risks and economic losses. Therefore, it is essential to detect Ponzi scheme smart contracts quickly and efficiently. The main challenges of the current Ponzi scheme detection method include the neglect of the behavior characteristics of smart contract opcodes, incomplete feature extraction, unstable performance, and low accuracy of the detection method when the method is subjected to anti-interference. To overcome these shortcomings, this study proposes an Ethereum Ponzi scheme

* 基金项目: 国家自然科学基金 (92267206, 62032013, 62432003); 广东省重点领域研发计划 (2020B0101090005)
收稿时间: 2024-02-01; 修改时间: 2024-12-12; 采用时间: 2025-10-11; jos 在线出版时间: 2025-12-17
CNKI 网络首发时间: 2025-12-26

detection method that combines TextCNN and adversarial training. This method extracts the behavioral characteristics of smart contracts by the static analysis of smart contracts, and combines the Word2Vec model to retain the semantic information of smart contracts to ensure the integrity of the opcode features. Meanwhile, the improved dynamic step projection gradient descent algorithm is adopted to train the TextCNN model to enhance the robustness of the detection model and improve the detection accuracy. Experiments carried out on the XBlock dataset show that the proposed method achieves a *Recall* of 98.36% and *F*1-*score* of 98.31% while ensuring precision and robustness. The method focuses on smart contract opcodes without relying on transaction features, and can quickly and efficiently detect Ponzi scheme smart contracts at the time of smart contract deployment.

Key words: smart contract; Ponzi scheme; behavioral sequence; adversarial training; TextCNN

1 引 言

近年来, 区块链凭借去中心化、匿名性、可追溯性和防篡改等特点发展迅速. 以太坊[1]作为区块链 2.0 的代表为开发者提供了一个支持智能合约开发的平台. 以太坊上部署了大规模的智能合约以便构建更广泛的应用. 智能合约[2]本质上是一段实现特定函数的代码, 同时也是一个独立于中心组织的去中心化应用程序. 由于以太坊中的智能合约具有满足一定条件后自动执行、成功部署后不能修改且运行后无法停止的特性[3], 使得投资者误以为智能合约中不存在诈骗行为, 导致智能合约成为庞氏骗局迷惑受害者进行投资的有效工具. Chainalysis 的数据[4]表明, 在 2017–2019 年期间, 以太坊上发生的诈骗事件影响了数百万人, 导致总计 43 亿美元的损失. 其中, 庞氏骗局占据了 92%, 受害者数量达到 240 万, 平均受损金额为 1 676 美元. 据 TRM Labs[5]监测, 加密领域庞氏及金字塔骗局危害显著且持续: 如图 1 所示, 2022 年庞氏及金字塔骗局的资金流入高达 168 亿美元, 给投资者造成巨额损失; 2024 年资金流入降至 43 亿美元, 但仍占全年诈骗资金的 40.19%. 庞氏骗局[6]伪装成“高收益”的投资项目, 实际上是以新投资者的投资作为旧投资者的短期回报, 制造虚假盈利假象来吸引更多投资者的诈骗行为. 更为严重的是, 智能合约的匿名性和溯源困难使得投资者被骗的资金难以追回, 投资者将面临巨大的损失. 因此, 庞氏骗局智能合约检测显得尤为重要.

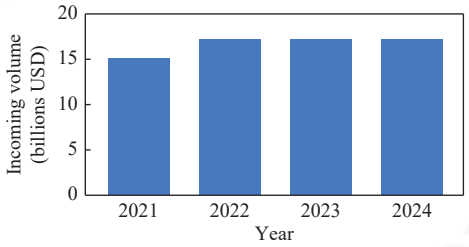


图 1 2021–2024 年加密货币市场中非法诈骗资金流入情况

目前的庞氏骗局检测机制主要包括基于统计模拟、基于机器学习以及基于符号执行等方法. 其中, 统计模拟方法[7]通过随机化算法估计智能合约的相似性, 并按提前设定的阈值来判断智能合约是否为庞氏骗局; 机器学习方法则根据人为制定的规则提取智能合约的交易特征和操作码特征, 通过数据集训练模型实现庞氏骗局的检测. 符号执行方法结合以太坊客户端生成智能合约可行路径的语义信息, 识别投资者相关的转移和分配策略, 从而实现庞氏骗局的检测.

现有的检测机制中, 在分析庞氏骗局交易历史数据集并提取交易特征时, 由于庞氏骗局智能合约数量较少, 交易特征对模型性能的影响有限. 在针对智能合约操作码进行分析时, 重点关注操作码的频率特征和语义特征, 忽略了操作码本身的行为信息和结果信息. 符号执行技术虽然可以模拟智能合约的行为, 但耗时且耗费资源, 易受规避技术的影响. 此外, 随着研究不断推进, 犯罪分子对现有检测机制进行对抗干扰可能性增加, 使得检测方法失效. 为得到一个高准确性和强抗干扰能力的庞氏骗局检测方法, 本文提出了一种基于改进对抗训练的庞氏骗局检测方法. 该方法以智能合约操作码行为序列为基础, 使用动态步长投影梯度下降算法 (dynamic step projection gradient descent algorithm, DSPGD) 对 TextCNN 进行对抗训练, 旨在确保庞氏骗局检测的准确性的同时提高模型鲁棒性.

本文的主要贡献包括以下内容.

1) 提出了融合 TextCNN 和对抗训练的庞氏骗局检测框架. 该框架可以在智能合约部署时对操作码进行深度分析, 以尽早、准确地识别潜在的庞氏骗局, 为智能合约的安全提供有效的支持.

2) 设计了智能合约操作码行为序列提取算法, 该算法通过对智能合约操作码进行静态分析, 构建操作码连通图, 最终生成保留操作码结构与行为信息的行为序列, 有效解决了特征提取不全面的问题.

3) 提出动态步长投影梯度下降算法 (DSPGD), 该方法通过添加步长变化控制因子和优化趋势判断, 解决了投影梯度下降算法 PGD 所存在的问题. 将该算法与 TextCNN 相结合 (称为 TextCNN_DSPGD), 提高了检测模型的鲁棒性和准确性.

4) 通过大量实验对提出的融合模型进行了验证, 并与已有经典方法进行性能对比. 结果表明, 在保证模型精确度的前提下, 该融合模型可有效增强模型的鲁棒性.

本文第 2 节综述庞氏骗局检测和对抗训练的相关工作. 第 3 节介绍本文提出方法, 其中包括行为序列提取过程、改进后的 DSPGD 算法以及 TextCNN 模型构建等. 第 4 节详细说明实验过程, 并对结果进行详细分析. 最后, 第 5 节总结全文, 并提出未来的研究方向.

2 相关工作

在本节中, 我们将综述庞氏骗局智能合约检测的现有方法以及对抗训练的研究进展.

2.1 庞氏骗局检测

Bartoletti 等人[7]最早开展庞氏骗局合约检测研究, 他们从描述性信息、源代码和交易记录这 3 个方面分析了庞氏骗局智能合约的共性, 并在其报告中首次对区块链上庞氏骗局的识别进行了实证研究, 通过人工识别对样本进行了区分, 并对相关合约在不同研究层面进行了详细分析. Chen 等人[8]根据 Bartoletti 等人[7]的分析结果, 从以太坊智能合约操作码中提取特征, 然后通过 XGBoost 模型将 Ethereum 上的智能合约与普通合约区分开来. 事实证明, 该方法比人工对比更加准确. 后来, Chen 等人[9]用随机森林 (random forest, RF) 的方法来检测庞氏骗局的智能合约, 准确率高于其他机器学习方法. 然而, 前者存在潜在的目标泄漏, 而后者通过不同值的类别特征产生的属性权重并不那么可靠. Jung 等人[10]在 Chen 等人[9]的基础上增加了账户特征, 但简化数据集更容易导致模型过拟合. He 等人[11]提出 CTRF 方法, 通过提取智能合约代码的操作码特征、序列特征和交易特征形成数据集, 同时采用过采样技术处理正负样本不平衡的问题. 现有的方法在处理类别特征和计算梯度估计时, 存在潜在的目标泄漏和预测转移问题. 于是 Fan 等人[12]介绍了一种检测区块链中智能庞氏骗局的新方法, 提出了一个基于有序提升思想的反泄漏智能庞氏骗局检测模型, 减轻梯度估计偏差, 提高了模型的泛化能力.

Wang 等人[13]提出一种基于超采样的智能合约长短期记忆 (long short term memory, LSTM) 的庞氏骗局检测方法, 整合了 SMOTE (synthetic minority over-sampling technique) 算法和 LSTM 来构建检测模型, 其中数据集由智能合约的账户特征和操作码的频率特征组成. Lou 等人[14]将智能合约字节码转化成不同大小的单通道图像, 并引入空间金字塔池化 (spatial pyramid pooling) 对卷积神经网络进行改进, 使得模型可以训练不同大小的图像. Bian 等人[15]提出了一种基于以太坊的注意力胶囊网络的基于图像的骗局检测方法. 根据合约地址下载得到其字节码文件和应用程序二进制接口, 从中提取特征转换成灰度图像, 再将它们合并生成 RGB 图像作为模型输入, 改善单一特征不能全面表征庞氏骗局合约特征的问题. Chen 等人[16]提出了一种将操作码转换成序列来处理数据的 Ponzi-Conv1D-Z3 方法, 以避免丢失重要信息. 他们使用一维卷积神经网络结合形式验证, 并利用形式验证工具 Z3 求解器对最终模型进行形式安全验证, 确保其安全性.

Sun 等人[17]在以太坊测试网络上的行为树上构建行为森林, 全面描述智能合约的动态行为, 采用自适应的全路径树编辑距离 (all path tree edit distance, AP-TED) 算法计算加权行为树之间的相似性. Chen 等人[18]基于动态符号执行技术提出 SADPonzi, 是一种用于识别以太坊智能合约中庞氏骗局的语义感知检测方法. 实际上, 它是一种启发式引导的符号执行技术, 模拟源程序的运行过程, 在根据庞氏骗局的再分配策略和用于存储用户信息的数据

结构来进行庞氏骗局的检测. Liang 等人[19]提出了 PonziGuard 方法, 建立了全面的合约运行时行为图以准确描述庞氏合约的行为. 然后将检测过程表述为图分类任务, 从而提高了整体效果. 然而, 这种动态分析操作码序列的方式, 耗费了大量的资源和时间.

目前的检测机制对智能合约的分析主要在易获取的操作码序列上, 但庞氏骗局的行为是一个连续的过程, 不应侧重于单个操作码的分析, 导致模型学习不充分, 而需要注重庞氏骗局操作码的结构特征与行为特征. 本文提出的行为序列提取算法静态分析智能合约操作码, 构建操作码连通图, 最终深度遍历连通图获得保留结构信息和行为信息的行为序列, 同时采用 Word2Vec 模型保留智能合约的语义信息.

2.2 对抗训练

近年来, 深度神经网络的对抗攻击问题引起广泛关注. 为增强深度学习模型对抗性鲁棒性, 人们提出了各种防御措施, 其中对抗性训练被认为是最有效的实践方法之一[20]. 对抗训练的思想是在每个训练迭代中用对抗性例子来增加训练数据. 因此, 对抗性训练的模型在面对对抗性例子时比标准训练的模型表现得更正常, 相较于标准训练的模型更具鲁棒性. Szegedy 等人[21]首次提出对抗样本的概念, 即通过向原始数据上添加扰动来构造对抗样本. 在图像领域, 对抗训练通常能提高其鲁棒性, 但往往会导致泛化性下降. 而在语言领域, 对抗训练不仅提高了鲁棒性还有助于提高泛化性能. 从数学上看, 对抗性训练被表述为一个最小-最大问题[22], 即寻找最坏情况下的最佳解决方案. 因此, 对抗训练的主要挑战可看作如何解决内部最大化问题[23].

Miyato 等人在文献 [24] 中提出快速梯度符号法 (fast gradient sign method, FGSM) 和快速梯度法 (fast gradient method, FGM), 可以让扰动的方向沿着梯度提升的方向, 也就意味着让损失增大到最大. FGSM 和 FGM 的区别在于采用了不同的归一化方法, FGSM 是通过 *sign* 函数对梯度采取最大归一化, FGM 则采用的是 L2 归一化. 而这两种方法都有个假设, 即损失函数是线性的或者是局部线性的. 如果不是 (局部) 线性的, 那梯度提升的方向就不一定是最优方向. 为了解决这一线性假设问题, Madry 等人[25]提出了使用投影梯度下降法 (projection gradient descent, PGD) 方法来求解内部的最大值问题. PGD 采用迭代攻击的方式, 相比于 FGSM 和 FGM 的单次迭代, PGD 进行多次迭代, 每次走一小步, 并且每次迭代都会将扰动投影到预先设定的范围内. 但在一次前向后向计算过程中, 不能同时利用计算出来的参数的梯度和输入的梯度. Shafahi 等人[26]在 PGD 的基础上提出 FreeAT, 对于每个 min-batch 的样本会求 K 次梯度, 每次求得的梯度, 既用来更新扰动, 也用来更新参数, 从而优化了对抗训练的效率.

Zhang 等人[27]提出了 YOPO (you only propagate once) 算法, 简化了生成对抗扰动的计算方法. Jiang 等人[28]提出了 SMART 算法, 使用平滑诱导对抗正则化方法来替代传统的极小-极大对抗训练算法. Li 等人[29]提出了一种令牌感知虚拟对抗训练方法, 引入令牌级的归一化, 更细粒度地约束扰动. 但其可能会受到少数梯度模长较大的令牌影响, 不利于神经网络的训练. Cheng 等人[30]采用分布归一化 (distribution normalization, DN) 和边际平衡 (margin balance, MB) 两种策略对不同类别的分布进行正则化的方法, 以提高对抗鲁棒性. 其中 DN 归一化类别特征, 消除易受攻击的类内方向. 而 MB 平衡不同类别之间的边缘, 使攻击更加困难. Zhao 等人[31]通过采用结合 Lyapunov 稳定性和保守 Hamiltonian 神经流来构建图卷积神经网络, 显著提高了其抵御对抗性扰动的能力.

在智能合约领域, Han 等人[32]在智能合约异常检测中引入对抗样本的概念, 生成了新型蜜罐样本. 受此启发, 欲利用对抗训练算法提高庞氏骗局检测模型的鲁棒性, 并不需要在对抗训练时生成文本层面上的真实样本. 本文将庞氏骗局智能合约的行为序列作为 TextCNN 的输入, 采用主流的对抗训练算法 PGD 进行对抗训练. PGD 的对抗过程实际上分为内外两部分, 内部是在约束范围内找到使得损失函数最大的最佳扰动, 外部则是在此扰动的作用下, 让模型拟合对抗样本, 提高模型的鲁棒性. 然而, PGD 总是按照固定步长寻找最佳扰动, 并且不能正确感知优化趋势, 存在局部最优问题. 本文在 PGD 的基础上提出了动态步长投影梯度下降算法, 该算法步长不再固定, 且结合两个方向来感知优化趋势, 更利于寻找最佳的扰动.

3 融合 TextCNN 和对抗训练的庞氏骗局检测方法

本文提出了一种基于动态步长投影梯度下降算法的庞氏骗局检测方法, 其整体架构如图 2 所示. 首先从智能

合约的操作码中提取行为序列作为数据集, 以保留操作码的行为信息和语义信息, 并将其划分为训练集、验证集和测试集. 采用 Word2Vec 模型对行为序列进行训练, 得到包含词语关联信息的词向量矩阵. 然后在词向量中加入扰动生成对抗样本, 再将对抗样本与原始样本结合用于联合训练 TextCNN 模型, 以模拟对模型的对抗攻击, 从而增强模型的鲁棒性. 最后, 采用测试集来验证模型的有效性, 主要通过测量其性能指标 (如 *Recall* 和 *F*1 分数) 来评估. 综上, 该方法使用包含丰富操作码结构与语义信息的行为序列作为数据集, 再引入对抗训练防御对抗攻击, 在一定程度上提高了模型在检测庞氏骗局时的有效性和鲁棒性.

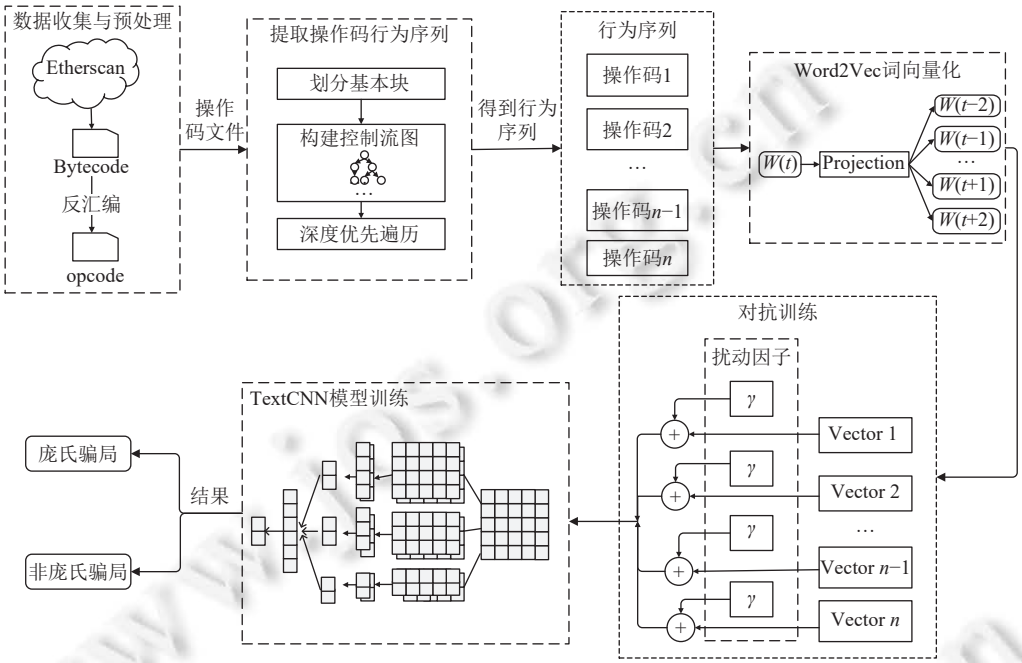


图 2 融合 TextCNN 和对抗训练的庞氏骗局检测模型框架

3.1 智能合约行为序列提取算法

智能合约是一个在以太坊虚拟机 (Ethereum virtual machine, EVM) 上运行的特殊程序, 包含了丰富的语义和结构信息. 在文献 [33] 的启发下, 我们采用操作码来捕获智能合约的行为特征. 操作码是字节码的存储形式, 操作码所对应的操作类型是固定的. 目前的庞氏骗局检测机制将智能合约操作码当作文本序列进行研究, 包括计算操作码的频率特征和语序特征等, 但却忽略了操作码本身的行为特性. 智能合约的特性让庞氏骗局源代码能够直接反映出其回报逻辑, 但源代码有可能通过其他手段掩盖其真实的行为. 相比之下, 操作码作为底层的执行指令, 直接决定了智能合约的运行过程, 不受其他中间环节的干扰, 确保合约行为的真实性. 因此, 可以通过分析操作码直观的了解合约的执行逻辑、数据处理方式以及条件判断的过程.

为了充分保留智能合约的结构和行为信息, 基于操作码的特性提出了行为序列提取算法如算法 1 所示, 通过分析智能合约操作码以构建智能合约操作码的控制流图, 并在控制流图的基础上提取行为序列, 从而保留丰富的智能合约结构和行为信息.

算法 1 主要分为 3 个阶段: 基本块划分阶段、控制流图构建和提取操作码行为序列时的遍历阶段. 在第 1 阶段, 根据控制流相关的操作码, 将原始操作码序列分成多个基本块. JUMP 和 JUMPI 是改变控制流的条件跳转操作码, 而 RETURN、STOP、INVALID、SELFDESTRUCT 和 REVERT 是停止执行操作码, 它们都作为基本块的最后一条指令. 此外, JUMPDEST 作为基本块中的第 1 条指令, 标志着跳跃目标的有效地址. 在第 2 阶段, 当处理基本块时, 如果 JUMP 和 JUMPI 指令的目标地址被指向以 JUMPDEST 开始的基本块, 在两个基本块之间添加跳

转边, 以构建控制流图. 然而, 仍然可能存在孤立的子图. 为保证行为序列的完整性, 如图 3 所示在孤立的子图上添加连接边得到一个连通图. 第 3 阶段, 通过使用深度优先遍历算法来遍历构建的连通图, 得到智能合约操作码的行为序列. 通过建立控制流关系和操作码之间的顺序依赖关系, 算法 1 确保了提取的行为序列的完整性.

算法 1. 智能合约行为序列算法.

输入: 操作码序列 opcodes;
输出: 行为序列 behavior_seq.

1. BEGIN
2. 控制流相关的操作码集合 Control, 基本块集合 basicblocks={}, 图 G<Block, E>
3. FOR opcode in opcodes DO //遍历操作码序列
4. IF opcode 是 Control 中的一种 THEN //比较得到最大的损失
5. 以 opcode 为基本块的最后一条指令, 得到一个基本块 block
6. 将 block 保存到基本块集合 basicblocks 中
7. END IF
8. END FOR
9. FOR block in basicblocks DO //遍历基本块集合
10. IF 当前基本块的最后一条指令是 JUMP 或 JUMPI THEN
11. 根据该基本块的跳转地址, 找到其跳转的基本块 block′
12. 添加 block 与 block′之间的跳转边
13. END IF
14. END FOR
15. 此时得到初始图 G′
16. IF G′中存在孤立基本块或孤立子图 THEN //与当前点的损失进行比较
17. 寻找以当前孤立基本块或孤立子图的起始指令–1 为结尾地址的上一基本块
18. 添加当前基本块与上一基本块之间的连接边
19. 得到完整的连通图 G
20. END IF
21. 深度优先遍历图 G, 得到行为序列 behavior_seq
22. END

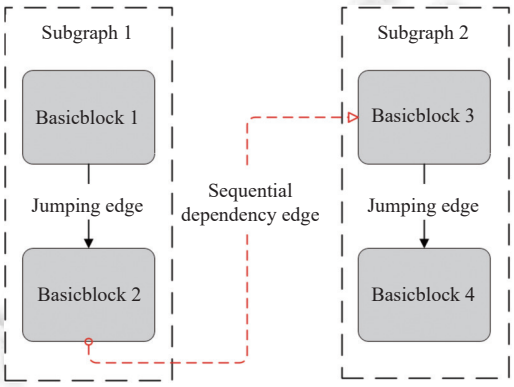


图 3 连通图示意

在得到行为序列之后使用 Word2Vec[34]、GloVe[35]等模型进行向量初始化. 预先训练的词向量嵌入可以利用多种语料库从而获得更多的先验知识, 而当前网络训练的词向量可以更好地捕获与当前任务相关的特征. 在本文中, 将智能合约行为序列作为输入, 操作码数量种类固定, 且重点关注操作码的分布情况, 因此本文采用 Word2Vec 作为词向量嵌入算法, 将行为序列输入到 Word2Vec 生成词向量, 向量维度 k 设置为 300. $V^i = R^k$ 是行为序列中第 i 个单词对应的 k 维单词向量. 长度为 n 的序列可以表示为矩阵 $V_{1:n} = [V^1, V^2, \ldots, V^n] \in R^{n \times k}$.

3.2 动态步长投影梯度下降算法

对抗性训练是深度学习模型防御对抗攻击的最有效方法之一. 与其他防御策略不同的是, 对抗训练旨在从本质上增强模型的稳健性. 投影梯度下降算法 (PGD) 是当前最流行的对抗训练算法, 但由于步长固定, PGD 无法保证在面对非凸问题时达到最优. 此外, 在使用神经网络时, 无法确保收敛, 并且无法确定扰动方向是否有利, 因此无法感知正确的优化趋势.

为了解决这些问题, 本文提出了一个动态步长投影梯度下降算法 (DSPGD), 引入动态步长控制因子来控制步长的大小, 并根据损失的更新情况来判定优化的方向. 如算法 2 所示, 该算法在 PGD 的基础上, 根据 $\alpha \pm \beta$ 确定最优点的方向, 如果在这两个方向上找到比上一次迭代更大的损失, 则表示该方向有效, 反之该方向无效, 减少步长为原来的 1/2, 继续进行迭代, 直到找到使得目标函数最大的扰动因子. 改进后的 PGD 步长可变, 且能够感知正确的优化方向, 从而更能找到使得目标函数最大的步长, 生成对抗样本 $x_{\max}$, 将 $x_{\max}$ 加入 TextCNN 模型进行训练, 让模型拟合该样本. 考虑到词嵌入含有丰富的语义信息, 因此将在词嵌入层进行扰动, 得到扰动后的向量矩阵 $V_{1:n}^{\text{adv}}$.

算法 2. 动态步长投影梯度下降算法.

输入: 行为序列 x, 初始步长 S, 变化步长 α, 控制因子 β, 约束阈值 ε, 迭代次数 K, 目标函数 $f(x)$;
输出: 对抗样本 x_{adv}, $\max_k f(x)$.

1. BEGIN
2. x^0 初始样本、$S = \alpha$ 初始步长
3. FOR k=1 TO K–1 DO
4. 计算当前样本的梯度 $\nabla f(x^k)$
5. $s_1 = \alpha + \beta$, $s_1 = \alpha - \beta$ //计算两个方向的步长, 步长可变
6. 计算由 s_1、s_2 生成的 $x_{s_1}^{k+1}$、$x_{s_2}^{k+1}$ 的梯度 $\nabla f\left(x_{s_1}^{k+1}\right)$、$\nabla f\left(x_{s_2}^{k+1}\right)$
7. IF $\nabla f\left(x_{s_1}^{k+1}\right) > \nabla f\left(x_{s_2}^{k+1}\right)$ THEN //比较得到最大的损失下的步长
8. $S = s_1$
9. ELSE IF $\nabla f\left(x_{s_1}^{k+1}\right) < \nabla f\left(x_{s_2}^{k+1}\right)$ THEN
10. $S = s_2$
11. END IF
12. $\nabla f(x_S^{k+1}) = \max\left(\nabla f\left(x_{S_1}^{k+1}\right), \nabla f\left(x_{S_2}^{k+1}\right)\right)$ //找到正确的优化方向
13. IF $\nabla f(x_S^{k+1}) > \nabla f(x^k)$ THEN //与当前点的损失进行比较
14. $S = S \div 2$
15. END IF
16. ELSE IF $\nabla f(x_S^{k+1}) < \nabla f(x^k)$ THEN
17. $x_{\text{adv}}^{k+1} = x^k + s \cdot sign(\nabla f(x_S^k))$ //生成对抗样本
18. $\max_k f(x) = f\left(x_{\text{adv}}^{k+1}\right)$
19. END IF

20. END FOR
21. END

3.3 文本卷积神经网络

TextCNN[36]是一种基于卷积神经网络的文本分类模型, 其架构如图 4 所示, 在这个架构中, 输入的文本首先被表示为一个词向量矩阵, 其中每一行代表一个词向量. 卷积层中将该词向量矩阵作为输入, 并使用不同大小的卷积核对文本进行卷积操作, 并利用 ReLU 激活函数进行非线性变换, 得到多个特征图. 卷积操作可以捕获不同长度的局部特征, 从而在不同的层次上提取文本的语义信息. 池化层对每个特征图进行最大池化操作, 从每个特征图中提取出最重要的特征向量. 将池化层输出的特征向量拼接起来形成一个全面的特征向量, 并使用全连接层对其进行归一化处理, 最终输出预测结果.

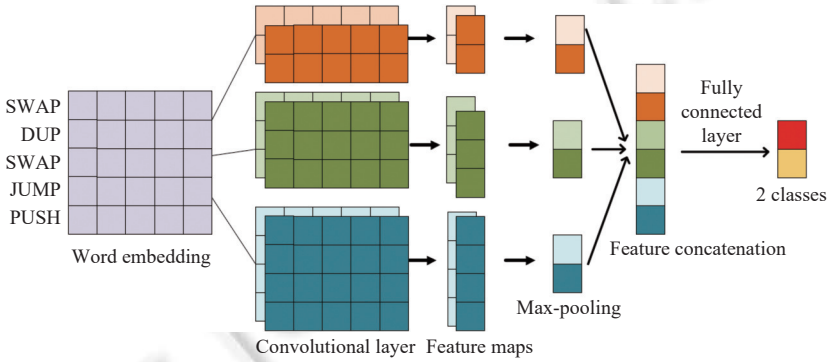


图 4 TextCNN 模型架构

将第 3.2 节中得到的词向量矩阵 $V_{1:n}^{\text{adv}}$ 作为 TextCNN 卷积层的输入. TextCNN 使用的卷积是一维卷积, 卷积核的宽度与词嵌入的维度 k (其中, $k=300$) 一致; 高度 h 为每个窗口中词的数量, 受文献 [12] 与实验结果的启发, 最终将卷积核的高度设置为 $h=\{2,3,4\}$, 所以卷积核 $\omega \in R^{k\times h}$ 具体可表示为:

$$\omega \in \left\{R^{300\times 2}, R^{300\times 3}, R^{300\times 4}\right\} \tag{1}$$

$V_{i:i+h-1}^{\text{adv}}$ 是输入矩阵的第 i 行到第 $i+h-1$ 行所组成的一个大小为 $k\times h$ 的窗口, 每个滑动窗口的卷积结果 C_i^m 如下:

$$C_i^m = f\left(\omega \cdot V_{i:i+h-1}^{\text{adv}} + b\right) \tag{2}$$

其中, $b\in R$ 是偏置项, f 是非线性函数. 对 $n-h+1$ 个滑动窗口同时进行卷积操作, 最终生成一个特征映射 $C_m=[C_1^m, C_2^m, \ldots, C_{n-h+1}^m]$. 池化操作是为了提取特征映射中的最大值. 于是将 C_m 传入池化层, 并使用最大池化法提取重要特征 $C_{\max}$, 如公式 (3) 所示:

$$C_{\max} = \max(C_m) \tag{3}$$

卷积核的数量决定输出的特征映射矩阵的维度. 假设卷积核的数量设置为 j 个, 再分别计算每个卷积核的最大值, 最后将其叠加成一个完整的特征映射矩阵 $M\in R^{m\times j}$ 作为输出向量. 此外, 可在全连接层之前添加 Dropout 操作来防止模型过拟合. 最后, 在全连接层中使用 Softmax 函数对池化层的输出向量进行归一化处理, 将其转换为我们想要的预测结果.

4 实验评估

4.1 数据集与评价指标

本文从 XBlock 中获得了已经标注的数据集, 其中包括 3 607 个非庞氏骗局智能合约和 296 个庞氏骗局, 两者的比例大概为 10:1. 此数据集中提供了智能合约的字节码文件, 因此我们通过搭建 Go-Ethereum 客户端将字节码

文件反编译为操作码文件, 并使用算法 1 对其进行处理, 进而得到智能合约的行为序列. 本文将智能合约的行为序列作为数据集, 并按照 7:1:2 的比例划分成训练集、验证集和测试集. 其中训练集用于模型训练, 验证集用于评估模型训练效果, 测试集用于对训练好的模型进行分类测试.

为了准确评估该模型的性能, 并方便与其他检测庞氏骗局智能合约的方法进行性能比较, 我们提出使用 3 个指标, 即精确度 (*Precision*)、召回率 (*Recall*) 和 *F*1 分数 (*F*1-score), 这 3 个指标均是取值越高, 模型的性能越好. 这 3 个指标的具体定义如下所示:

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

$$F1\text{-}score = 2\times\frac{Precision\times Recall}{Precision+Recall} \tag{6}$$

其中, *TP* 表示被正确识别为正样本的数量, *TN* 表示被正确识别为负样本的数量, *FP* 表示负样本被识别为正样本的数量, *FN* 表示正样本被识别为负样本的数量.

ROC 曲线用于可视化分类模型性能, 它显示了在不同分类阈值下, 真正阳率 (*TPR*, 也称召回率) 与假正阳率 (*FPR*) 之间的权衡关系. ROC 曲线的横轴是 *FPR*, 纵轴是 *TPR*. *TPR* 和 *FPR* 的具体定义如公式 (7) 和公式 (8) 所示. ROC 曲线下的面积 (AUC), 用于量化分类模型性能. AUC 的取值范围通常在 0–1 之间, 其中 1 表示完美分类器, 0.5 及其以下表示模型性能等同于随机猜测. AUC 值越大, 说明模型在不同阈值下的分类性能越好.

$$TPR = \frac{TP}{TP+FN} \tag{7}$$

$$FPR = \frac{FP}{FP+TN} \tag{8}$$

4.2 基准模型

(1) 对抗训练算法: 本文使用 TextCNN 作为分类器, 但在模型的训练过程中发现存在梯度消失的问题, 同时容易受到对抗攻击. 因此为了提高模型的鲁棒性, 将引入对抗训练算法进行模型训练. 将当前流行的 3 种对抗训练算法 FGM、FreeAT 和 PGD 与改进之后的动态步长投影梯度算法 (DSPGD) 分别用于模型训练.

(2) 庞氏骗局检测方法: 在以往的研究中, 机器学习方法广泛应用于庞氏骗局检测, 其中最常用的方法包括 XGBoost[37]、LightGBM[38]、RF[39]等. 然而前两者是经典的梯度提升算法, 存在梯度偏差引起的过拟合问题, 会影响模型的泛化性能. 深度学习方法也被广泛用于异常检测, 如卷积神经网络 (CNN)[40]、文本卷积神经网络 (TextCNN) 和 LSTM[13]等. 此外, 动态符号执行方法 SADPonzi 进行庞氏骗局智能合约检测, 能够输出具有庞氏骗局类别的检测报告.

在选择庞氏骗局检测基准模型时综合考虑了多种因素, 选择了一系列具有代表性的模型和方法. 其中, 机器学习领域的 XGBoost、LightGBM、RF 以及神经网络范畴的 CNN、TextCNN、LSTM 等被纳入基准模型, 归因于它们在数据处理、特征提取及序列建模等方面具有广泛的实践应用和代表性优势, 能够从多个维度全面、精准地评估我们所提新方法的性能表现. 而 SADPonzi 作为基准模型之一, 因其基于动态符号执行技术, 能依据庞氏骗局关键特征进行语义感知检测, 为我们提供独特对比视角. 而相关工作中的其他技术, 如 PonziGuard 资源耗费大且通用性受限, Ponzi-Conv1D-Z3 依赖特定工具且针对性过强, 部分基于特定特征工程或数据集处理的方法稳定性和通用性不足, 故而未被选作基准模型. 通过严谨的基准模型选择与分析过程, 便于更加精准评估我们所提出的检测方法的性能与价值.

4.3 实验结果与分析

4.3.1 鲁棒性验证

TextCNN 常需要将输入的行为序列的长度进行标准化处理, 确保序列的长度相同. 于是对行为序列的长度进

行分析, 绘制了概率分布直方图如图 5 所示, 智能合约的长度大多集中在 0–2 048 之间, 其中庞氏骗局中约有 58.7% 在此区间, 非庞氏骗局中约 56.05%. 为了评估 DSPGD 算法的有效性, 将 4 种对抗训练算法 (FGM、FreeAT、PGD、DSPGD) 分别加入 TextCNN 模型中进行对抗训练, 并考虑不同嵌入长度 (1 024、1 536、2 048) 进行实验, 以保留操作码行为序列的行为特征和语义特征.

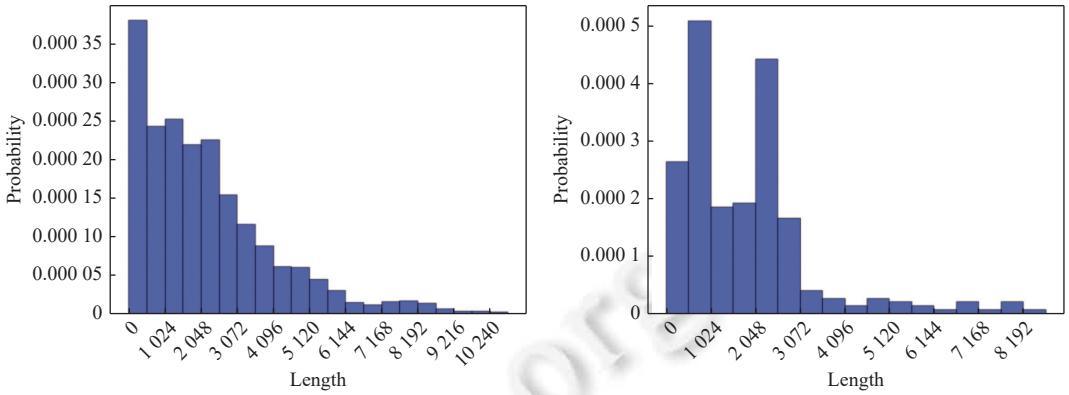


图 5 行为序列长度的概率分布直方图

图 6 展示了 4 种对抗训练算法在不同嵌入长度下的精确度、召回率和 $F1$ 分数的表现. 与 FGM、FreeAT 和 PGD 相比, DSPGD 的精确度、召回率和 $F1$ 分数获得最佳的效果, 且当嵌入长度为 1 536 时检测模型的性能最佳. 由此说明, DSPGD 与 TextCNN 结合之后能有效提升庞氏骗局的检测性能.

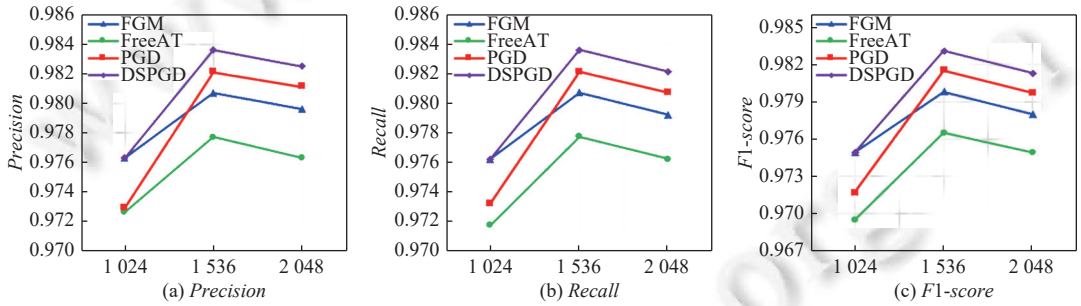
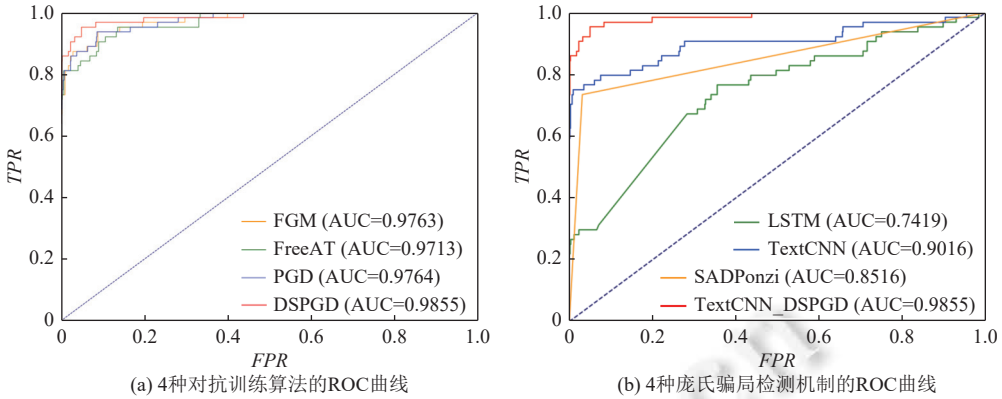


图 6 4 种对抗训练算法的性能对比图

为了进一步验证模型抗干扰能力, 后文图 7(a) 中绘制了 4 种对抗训练算法的 ROC 曲线, 其中 DSPGD 的 ROC 曲线 (红色) 更加接近左上角, AUC 为 0.985 5, 更接近 1, 由此说明加入 DSPGD 算法后的模型分类性能更好. 另一方面, 后文图 8 中展示了 4 种对抗训练算法在模型训练中的损失情况, 可以发现在 DSPGD 算法的训练下, 模型的损失最快收敛且达到最小值. 损失与鲁棒性之间往往是负相关的, 而 DSPGD 不仅获得更小的损失且有更好的检测效果, 因此可以证明 DSPGD 算法能有效增强模型的鲁棒性.

4.3.2 检测方法的性能比较

针对第 4.2 节中提到的前 4 种庞氏骗局检测方法 RF、XGBoost、LightGBM 以及 CNN_RF 均使用 79 个操作码频率特征和经过 Doc2vec 提取的 20 个操作码序列特征进行实验. 而 TextCNN、LSTM 使用操作码序列, SAD-Ponzi 使用字节码文件进行实验. 另外, TextCNN_DSPGD 使用第 3.1 节提取的操作码行为序列作为输入进行实验. 实验结果见表 1.



(a) 4种对抗训练算法的ROC曲线 (b) 4种庞氏骗局检测机制的ROC曲线

图 7 ROC 曲线

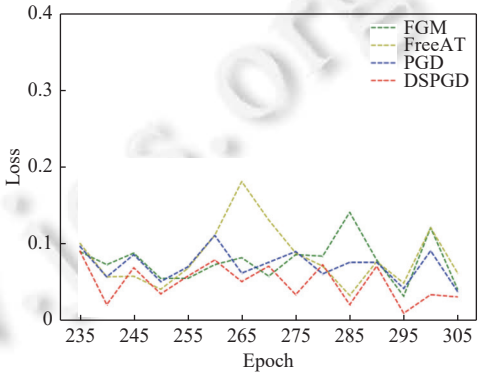


图 8 4 种对抗训练算法的训练时的损失情况

表 1 TextCNN_DSPGD 与庞氏骗局检测方法的性能对比

模型	*Precision*	*Recall*	*F*1-*score*
RF	0.9538	0.7047	0.8095
XGBoost	0.9257	0.7602	0.8327
LightGBM	0.9342	0.7740	0.8453
CNN_RF	0.9268	0.7308	0.8172
TextCNN	0.9627	0.9673	0.9649
LSTM	0.9129	0.9226	0.9050
SADPonzi	0.9472	0.9464	0.9468
TextCNN_DSPGD	**0.9836**	**0.9836**	**0.9831**

从结果上来看, TextCNN_DSPGD 着实表现出优秀的性能. 与常见的机器学习模型中表现最好的 LightGBM 方法相比, TextCNN_DSPGD 在 *Recall* 和 *F*1-*score* 上分别提高了 27.04% 和 16.25%. 与神经网络中表现最好的 TextCNN 相比, 其在 *Recall* 和 *F*1-*score* 上分别提高了 1.69% 和 1.89%. 最后, 与 SADPonzi[18]相比, TextCNN_DSPGD 在 *Recall* 和 *F*1-*score* 上分别提高了 3.92% 和 3.84%. 总体而言, 本文所提出的方法更优一些.

为了深入探讨 TextCNN_DSPGD 的检测性能, 图 7(b) 展示了表 1 中表现最好的 4 个模型 TextCNN、LSTM、SADPonzi 和 TextCNN_DSPGD 的 ROC 曲线. 其中 SADPonzi 的 ROC 曲线 (橙色) 只有一个折点, SADPonzi 方法的输出是标签, 而不是预测概率, 因此只有一个分类阈值, 不会出现多个折点. 而 TextCNN_DSPGD 的 ROC 曲线 (红色) 更靠近右上角, 当 *TPR* 达到 1 时 *FPR* 约为 0.4 左右, *TPR* 远大于 *FPR*, 且具有最高的 AUC 值, 足以说明本

文模型检测性能的显著优势. 此外, 4 种检测模型的混淆矩阵如图 9 所示, TextCNN_DSPGD 展现出最佳的性能表现, 正确识别了 54 个庞氏骗局和 607 个非庞氏骗局, 仅有 10 个 *FP* 和 1 个 *FN*. 相比之下, 其他方法的 *FN* 和 *FP* 都较高.

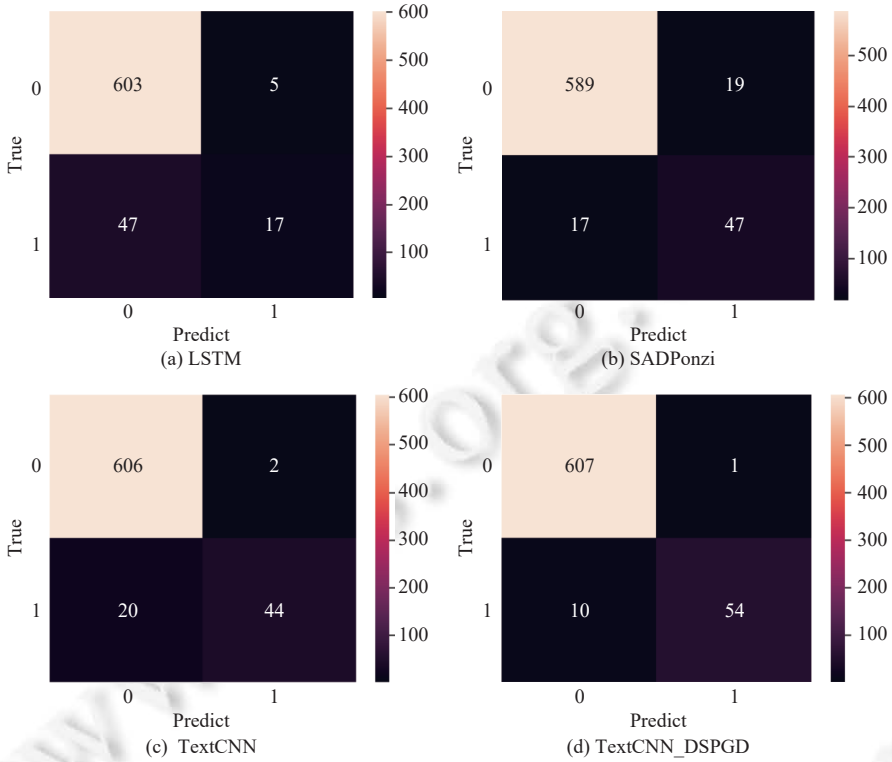


图 9 4 种检测机制的混淆矩阵图

综上, TextCNN_DSPGD 是一种基于对抗训练的庞氏骗局检测方法, 其不仅较全面地提取了操作码的语义和行为信息, 并且通过对抗训练增强模型的鲁棒性. 同时该方法不依赖交易历史特征, 能在智能合约部署后尽快准确地检测出庞氏骗局, 更加具有实践意义.

5 总 结

本文提出了一个基于动态步长投影梯度下降算法的 TextCNN 模型 (TextCNN_DSPGD) 用于检测以太坊上的庞氏骗局智能合约. 首先, 我们从 XBlock 上获取智能合约字节码文件, 将其反汇编成操作码文件. 接着, 我们设计了基于深度优先的智能合约行为序列提取算法, 对操作码文件进行特性提取, 获得包含丰富语义和行为信息的智能合约行为序列. 最后, 在 TextCNN 模型中引入 DSPGD 进行对抗训练, 并通过实验与其他算法进行比较, 验证了本文所提出方法在检测庞氏骗局智能合约方面的有效性. 实验结果显示, TextCNN_DSPGD 在 *Recall* 和 *F*1-*score* 上分别取得了 98.36% 和 98.31% 的性能, AUC 高达 0.985 5, 均超过对比方法. 在未来的工作中, 我们计划收集更多的庞氏骗局智能合约, 进一步验证 TextCNN_DSPGD 的通用性. 此外, 我们准备将我们的方法扩展到区块链领域的其他恶意行为检测中, 以确保以太坊生态系统的安全.

References

[1] Buterin V. Ethereum white paper: A next generation smart contract & decentralized application platform. 2025. https://github.com/ethereum/wiki/wiki/White-Paper

[2] Wood G. Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, 2014, 151: 1–32.
[3] Taherdoost H. Smart contracts in blockchain technology: A critical review. Information, 2023, 14(2): 117. [doi: 10.3390/info14020117]
[4] Chainalysis. The Chainalysis 2021 crypto crime report. 2020. https://go.chainalysis.com/2020-Crypto-Crime-Report.html
[5] TRM. 2025 crypto crime report. 2025. https://www.trmlabs.com/reports-and-whitepapers/2025-crypto-crime-report
[6] Vasek M, Moore T. Analyzing the bitcoin Ponzi scheme ecosystem. In: Proc. of the 22nd Int'l Conf. on Financial Cryptography and Data Security. Nieuwpoort: Springer, 2019. 101–112. [doi: 10.1007/978-3-662-58820-8_8]
[7] 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]
[8] Chen W, Zheng Z, Cui J, Ngai E, Zheng P, Zhou Y. Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology. In: Proc. of the 2018 World Wide Web Conf. Lyon: Int'l World Wide Web Conf. Steering Committee, 2018. 1409–1418. [doi: 10.1145/3178876.3186046]
[9] 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]
[10] Jung E, Le Tilly M, Gehani A, Ge YJ. Data mining-based Ethereum fraud detection. In: Proc. of the 2019 IEEE Int'l Conf. on Blockchain (Blockchain). Atlanta: IEEE, 2019. 266–273. [doi: 10.1109/Blockchain.2019.00042]
[11] He XZ, Yang T, Chen LP. CTRF: Ethereum-based Ponzi contract identification. Security and Communication Networks, 2022, 2022: 1554752. [doi: 10.1155/2022/1554752]
[12] Fan SH, Fu SJ, Xu HR, Cheng XC. Al-SPSD: Anti-leakage smart Ponzi schemes detection in blockchain. Information Processing & Management, 2021, 58(4): 102587. [doi: 10.1016/j.ipm.2021.102587]
[13] 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]
[14] Lou YC, Zhang YM, Chen SP. Ponzi contracts detection based on improved convolutional neural network. In: Proc. of the 2020 IEEE Int'l Conf. on Services Computing (SCC). Beijing: IEEE, 2020. 353–360. [doi: 10.1109/SCC49832.2020.00053]
[15] Bian LY, Zhang LL, Zhao K, Wang H, Gong SJ. Image-based scam detection method using an attention capsule network. IEEE Access, 2021, 9: 33654–33665. [doi: 10.1109/ACCESS.2021.3059806]
[16] Chen SB, Li F. Ponzi scheme detection in smart contracts using the integration of deep learning and formal verification. IET Blockchain, 2024, 4(2): 185–196. [doi: 10.1049/blc2.12056]
[17] 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 (QRS). Macao: IEEE, 2020. 297–309. [doi: 10.1109/QRS51102.2020.00047]
[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] Liang RC, Chen J, He K, Wu YM, Deng GL, Du RY, Wu C. PonziGuard: Detecting Ponzi schemes on Ethereum with contract runtime behavior graph (CRBG). In: Proc. of the 46th IEEE/ACM Int'l Conf. on Software Engineering. Lisbon: ACM, 2024. 64. [doi: 10.1145/3597503.3623318]
[20] Goodfellow IJ, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. arXiv:1412.6572, 2014.
[21] Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R. Intriguing properties of neural networks. arXiv: 1312.6199, 2013.
[22] Athalye A, Carlini N. On the robustness of the CVPR 2018 white-box adversarial example defenses. arXiv:1804.03286, 2018.
[23] Huang RT, Xu B, Schuurmans D, Szepesvári C. Learning with a strong adversary. arXiv:1511.03034, 2015.
[24] Miyato T, Dai AM, Goodfellow I. Adversarial training methods for semi-supervised text classification. arXiv:1605.07725, 2016.
[25] Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A. arXiv:1706.06083, 2017.
[26] Shafahi A, Najibi M, Ghiasi MA, Xu Z, Dickerson J, Studer C, Davis LS, Taylor G, Goldstein T. Adversarial training for free! In: Proc. of the 33rd Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2019. 302.
[27] Zhang DH, Zhang TY, Lu YP, Zhu ZX, Dong B. You only propagate once: Accelerating adversarial training via maximal principle. In: Proc. of the 33rd Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2019. 21.
[28] Jiang HM, He PC, Chen WZ, Liu XD, Gao JF, Zhao T. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. ACL, 2020. 2177–2190. [doi: 10.18653/v1/2020.acl-main.197]
[29] Li LY, Qiu XP. Token-aware virtual adversarial training in natural language understanding. In: Proc. of the 35th AAAI Conf. on Artificial

Intelligence. AAAI, 2021. 8410–8418. [doi: 10.1609/aaai.v35i9.17022]
[30] Cheng Z, Zhu F, Zhang XY, Liu CL. Adversarial training with distribution normalization and margin balance. Pattern Recognition, 2023, 136: 109182. [doi: 10.1016/j.patcog.2022.109182]
[31] Zhao K, Kang QY, Song Y, She R, Wang SJ, Tay WP. Adversarial robustness in graph neural networks: A Hamiltonian approach. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2024. 148.
[32] Han Y, Ji TT, Wang ZR, Liu H, Jiang H, Wang WD, Cui X. An adversarial smart contract honeypot in Ethereum. Computer Modeling in Engineering & Sciences, 2021, 128(1): 247–267. [doi: 10.32604/cmes.2021.015809]
[33] Ding YX, Wei D, Zhang YB, Xue CL. Malicious code detection using opcode running tree representation. In: Proc. of the 9th Int'l Conf. on P2P, Parallel, Grid, Cloud and Internet Computing. Guangzhou: IEEE, 2014. 616–621. [doi: 10.1109/3PGCIC.2014.140]
[34] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv:1301.3781, 2013.
[35] Pennington J, Socher R, Manning CD. GloVe: Global vectors for word representation. In: Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP). Doha: ACL, 2014. 1532–1543. [doi: 10.3115/v1/D14-1162]
[36] Chen YH. Convolutional neural network for sentence classification [MS. Thesis]. Waterloo: University of Waterloo, 2015.
[37] Chen TQ, Guestrin C. XGBoost: A scalable tree boosting system. In: Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. San Francisco: ACM, 2016. 785–794. [doi: 10.1145/2939672.2939785]
[38] Ke GL, Meng Q, Finley T, Wang TF, Chen W, Ma WD, Ye QW, Liu TY. LightGBM: A highly efficient gradient boosting decision tree. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 3149–3157.
[39] Breiman L. Random forests. Machine Learning, 2001, 45(1): 5–32. [doi: 10.1023/A:1010933404324]
[40] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc. of the IEEE, 1998, 86(11): 2278–2324. [doi: 10.1109/5.726791]

作者简介

宁小勇, 硕士生, 主要研究领域为机器学习, 以太坊庞氏骗局智能合约检测, 网络安全.
高睿杰, 硕士生, 主要研究领域为网络安全, 以太坊钓鱼行为研究.
叶楚涵, 硕士生, 主要研究领域为人工智能, 以太坊勒索行为检测.
刘园, 博士, 教授, 博士生导师, 主要研究领域为网络安全威胁情报共享, 数据安全, 区块链安全.
王兴伟, 博士, 教授, 博士生导师, CCF 会士, 主要研究领域为互联网, 云计算, 网络安全.
黄敏, 博士, 教授, 博士生导师, 主要研究领域为数据解析与机器学习, 计算机智能软件工程.