Hindawi
Security and Communication Networks
Volume 2022, Article ID 2316310, 15 pages
https://doi.org/10.1155/2022/2316310

Research Article

Ethereum Ponzi Scheme Detection Based on PD-SECR

Shuhui Zhang, Tian Lan, Lianhai Wang, Shujiang Xu, and Wei Shao

Qilu University of Technology (Shandong Academy of Sciences),
Shandong Computer Science Center (National Supercomputer Center in Jinan),
Shandong Provincial Key Laboratory of Computer Networks, Jinan 250014, China

Correspondence should be addressed to Shuhui Zhang; zhangshh@sdas.org

Received 29 April 2022; Revised 16 August 2022; Accepted 26 August 2022; Published 21 September 2022

Academic Editor: Jie Cui

Ethereum, a typical application of blockchain technology, has attracted extensive attention from all walks of life since its release. Owing to imperfections in existing supervision technology, illegal and criminal activities on blockchain platforms are becoming increasingly frequent. The most typical Ethereum fraud is the Ponzi scheme, which causes blockchain investors to lose millions of assets and severely impacts social development. Currently, Ponzi scheme detection primarily focuses on machine learning and data mining. However, existing detection methods still have two problems in data imbalance processing and feature extraction: (1) data enhancement using an oversampling algorithm produces noise and (2) feature redundancy existing in extracted feature data. The SMOTEENN algorithm is introduced to solve data imbalance. The PD-SECR method, the Convolutional Neural Network (CNN) feature extraction, and random forest (RF) classification models are used for detection, but the two models are independently trained. The results show that the detection method proposed in this study is more suitable for the Ethereum Ponzi scheme.

1. Introduction

Blockchain is a list of connected blocks in chronological order. In 2008, Satoshi Nakamoto proposed a new type of distributed ledger [1]. Blockchain, the underlying technology of Bitcoin, is decentralized, immutable, and traceable. Since its rise, blockchain technology has attracted the attention of all walks of life due to its unique characteristics [2]. A smart contract is a communication protocol that allows participants who do not trust each other to interact [3]. However, the lack of a secure and enforceable environment has hindered the development of smart contracts. The blockchain platform provides a trusted execution environment for smart contracts. Because of the programmability of intelligent contracts, various business functions can be realized by writing smart contracts.

Ethereum is a typical blockchain platform with smart contracts. Once the contract is successfully deployed on Ethereum, it is executed automatically without human intervention [4–6]. Although it avoids tampering with the contract code to a certain extent, it can also be used by criminals. Every new technology has various security issues, and blockchain technology is no exception. Studies have found that transaction fraud (e.g., Ponzi schemes and phishing accounts) is a typical security problem on blockchain platforms. Researchers investigated fraud from 2013 to 2014 and found that the financial loss caused by fraud was as high as $7 million in a year [7]. With the continuous improvement of blockchain technology, the complexity of related technologies is also increasing, resulting in high technical barriers between blockchain and investors.

By exaggerating the advantages of blockchain, criminals take advantage of the high technical barriers between blockchain and investors to induce investors to invest [8, 9]. The Ponzi scheme is an old form of investment fraud. Figure 1 shows the return mechanism of a Ponzi scheme masquerading as a high-yield cryptocurrency. Ponzi schemes are now reappearing in society as blockchain-based schemes [7, 10–15]. Ethereum, the preferred platform for blockchain fraud, still lacks an adequate regulatory mechanism. Therefore, Ponzi scheme detection in Ethereum has become a hot topic in current research.

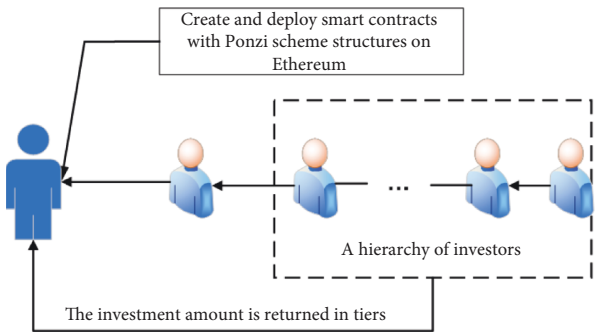


Figure 1: Cryptocurrency Ponzi scheme return mechanism.

Early studies focused on the Bitcoin platform [10–12], and Ponzi scheme detection on Ethereum is still lacking. Since 2018, the Ponzi scheme detection research on Ethereum has gradually increased. Existing Ethereum Ponzi scheme detection methods mostly rely on machine learning and data mining technology [14, 15, 15–18]. Although reducing the burden of manual analysis and detection, existing detection methods still have the following problems and challenges: (1) data enhancement processing using an oversampling algorithm produces noise, (2) the complexity of feature extraction methods and redundancy of feature data, and (3) the detection performance of the detection model can still be improved.

Given these challenges, this study proposes a PD-SECR detection method that introduces the SMOTEENN-mixed sampling algorithm to improve the combination model of CNNs and RFs. To test the model's credibility, we select several indicators, precision, recall, and F1-score, commonly used in anomaly detection for evaluation. The main contributions of this study are as follows:

(i) The SMOTEENN algorithm is introduced for data enhancement to avoid data repetition after data enhancement.

(ii) The automatic extraction of key features using CNN to avoid feature redundancy.

(iii) CNN and RF are fused for classification detection, which improves the detection accuracy of the model.

The remainder of this paper is organized as follows. In Section 2, we discuss related work. In Section 3, we introduce the preliminaries. The fourth section describes the proposed detection method in detail. Section 5 presents specific experimental steps and performance evaluation comparisons. We summarize this study and our next research plan in the last quarter.

2. Related Work

With the rapid development of blockchain, the technical difficulty has also risen. Technical barriers between investment users and blockchain make it harder for investors to spot frauds like Ponzi schemes. In 2012, Moore et al. [19] provided a macro-definition of the high-yield investment program (HYIP), an online Ponzi scheme, and elaborated on the method of fraud in the Ponzi scheme. The advent of blockchain has had a significant impact on various fields. Meanwhile, Blockchain platforms provide a possible avenue for the diffusion of trading fraud proliferation. The trading fraud, based on blockchain, instantly swept the entire Internet. Especially, a Ponzi scheme fraud detection boom was set off. Ponzi schemes focus on Bitcoin and Ethereum, the two most widely used trading platforms. Therefore, in this subsection, we describe our related research from the following three aspects: Bitcoin and Ethereum Ponzi scheme detection and CNN_RF feasibility analysis.

2.1. Bitcoin Ponzi Scheme Detection. In 2015, Vasek and Moore [7], for the first time, conducted an empirical analysis of fraud based on Bitcoin: Operations with fraudulent intent established. By amalgamating reports gathered by voluntary vigilantes and tracked in online forums, 192 scams were identified and grouped into four categories: Ponzi schemes, mining scams, scam wallets, and fraudulent exchanges. The significant finding of this work showed that bitcoin trading scams were diverse. Besides, this study provided a labeled dataset of bitcoin fraud for later researchers to study fraud detection methods. At the same time, intended to analyze the factors behind the success of Ponzi schemes in Bitcoin transactions, Vasek and Moore [10] identified 1,780 scams by searching 11,424 threads on https://bitcointalk.org. Through survival analysis, they identified factors that influence the persistence of fraud. Due to the low timeliness of manual examination and detection, researchers gradually introduced machine learning technology into bitcoin transaction fraud detection. In 2016, Monaco et al. [11] investigated using trimmed k-means, capable of simultaneous clustering fraud detection objects in multi-variable settings, to detect fraudulent activities in bitcoin transactions. Unsupervised learning, while reducing the need for labeled data, also reduces detection accuracy. In 2018, Bartoletti et al. [12] used data mining technology to mine features from real-world Ponzi scheme data and construct feature datasets, using classical machine learning algorithms for anomaly detection. In 2021, Nerurkar et al. [13] proposed a decision tree integration algorithm to solve the problem of limited categories in identifying and detecting illegal users in Bitcoin transactions.

2.2. Ethereum Ponzi Scheme Detection. In 2018, Chen et al. [14] used the XGBoost algorithm to realize the automatic detection of an intelligent Ponzi scheme. However, they ignored the problem of unbalanced sample data, resulting in the low generalization ability of the trained model. In 2021, Bartoletti et al. [15] comprehensively investigated Ponzi schemes on Ethereum, analyzing their behavior and impact from various viewpoints. It can be divided into four categories (i.e., tree-shape schemes, chain-shape schemes, waterfall schemes, and handover schemes) according to different payment methods. They provided a comprehensive information reference for follow-up research. In 2021, Zhang et al. [16] considered the imbalance of positive and negative sample proportions and introduced the

SMOTE + Tomek algorithm to improve the LightGBM model's detection accuracy. However, these two detection methods [14, 16] have problems with target leakage and prediction deviation. In the same year, Fan et al. [17] proposed a detection method, AI-SPSD, similar to the CaBoost algorithm, to solve the problems of target leakage and prediction offset ignored in previous studies. For data imbalance processing, they introduced the Borderline_SMOTE2 algorithm [18] for processing. Although the influence of boundary nodes on the detection results is considered, the problem of processing data duplication is still not considered. Although Zhang et al. considered the issue of data duplication, the data cleaning operation of the SMOTE + Tomke algorithm only finds the cleaning of multiple samples, which quickly leads to a significant deviation from the actual data and is unconvincing. Due to the unique effect of deep learning in security detection, researchers gradually introduced deep learning technology to detect Ethereum Ponzi schemes. In 2021, Luo et al. [20] converted the contract bytecode into a grayscale image. Due to the varying length of the bytecode, they introduced a spatial pooling algorithm to improve the convolutional neural network to handle grayscale images with inconsistent sizes better. Although the detection methods are novel, they ignore the problem of model overfitting caused by data imbalance. Wang et al. [21] introduced the SMOTE data enhancement algorithm to improve the LSTM model, eliminate the restriction of the machine learning model, and enter the detection stage of deep learning. Although they have paid attention to the data imbalance problem, SMOTE data augmentation is prone to data duplication, which is not conducive to detecting actual data by the detection model.

2.3. CNN_RF Fusion Feasibility Analysis. The effective fusion of convolutional neural networks and random forest models is also a significant challenge for this study. As we all know, CNN is the most widely used image classification, and more and more researchers migrate it to text classification because of its robust feature extraction function. In 2018, Wang et al. [22] proposed a densely connected CNN with multi-scale feature attention for text classification. This research solves the problem of combining larger-scale features with smaller-scale features, demonstrating the power of CNN's feature extraction capabilities. In 2019, Guo et al. [23] proposed a novel term weighting scheme combined with word embeddings to improve the classification performance of CNNs. Today, convolutional neural networks are widely used in image processing and text classification. Therefore, researchers started to consider the feasibility of using neural networks in conjunction with machine learning algorithms. So far, many methods of using neural networks in combination with machine learning algorithms have appeared. In the paper, we focus on the research and discussion of the advantage of combining convolutional neural networks and random forests. In 2020, Yang et al. [24] proposed a novel crop classification method based on optimal feature selection (OFSM) and a hybrid convolutional neural network random forest (CNN-RF) combined. To solve the problem of extracting useful information from massive data to balance classification accuracy and processing time. In 2021, Kwak et al. [25] proposed a CNN-RF joint detection method that combines the automatic feature extraction ability of CNN with the excellent discrimination ability of the RF classifier, aiming at the problem of limited input data for crop classification.

Inspired by the research discussion above, this paper proposes the PD-SECR approach, a Ponzi scheme detection method based on mixed sampling-based CNN-RF. Account features and opcode features are extracted from the contract's internal and external transaction information and the decompiled opcode of the running bytecode after the contract is deployed. Combining account features and opcode features is used to detect Ethereum Ponzi schemes. At the same time, we consider the data duplication after the training data is processed by data augmentation and solve it through mixed sampling. The processed dataset is used for crucial feature extraction using a CNN model, and then a random forest classifier is used for classification detection.

3. Preliminaries

This section introduces some of the relevant knowledge covered by the research, such as Ethereum, smart contracts, and Ponzi schemes.

3.1. Ethereum and Contracts. *Ethereum* is an innovation that applies the underlying technology of Bitcoin to computing. Like Bitcoin, it uses blockchain technology and peer-to-peer (P2P) networks to maintain a shared computing platform. A smart contract, first proposed by [26], is defined as a set of computer programs that implement the terms of the contract. Smart contracts allow trusted and traceable transactions without a trusted third party. However, the development of smart contracts has been severely hampered by a lack of a reliable enforcement environment. In 2013, Buterin [27] published a white paper on Ethereum. The Ethereum platform provides a reliable execution environment for smart contracts and promotes the further development of smart contracts.

Since the introduction of smart contracts into the blockchain, the application of blockchain technology has extended from the financial field to other fields [28–31]. Blockchain offers decentralized solutions for all domains. Since then, society has stepped into the blockchain 3.0 era of Smart IoT. In addition, yellow paper of Ethereum gives a complete list of opcodes corresponding to the bytecode. When a smart contract is deployed to the Ethereum platform and compiled into the corresponding bytecode, an available decompiler tool can decompile the corresponding opcodes from bytecodes. It significantly facilitates our subsequent research.

3.2. Ponzi Scheme. The Ponzi scheme is a typical investment fraud in the financial field, known as "robbing Peter to pay Paul" in China. In short, a Ponzi scheme uses money from new investors to pay interest or provide short-term returns to earlier investors. It is fraudulent means, and scammers make money from it.

Due to the sizeable technical barrier between blockchain technology and investors, many investors believe that a smart contract project with continuous operation and income does not have fraud risk. However, any emerging technology is vulnerable to fraud, and the anonymity of its sponsors makes smart contracts particularly difficult to financially regulate.

Today, malicious speculators exaggerate blockchain's unique features to attract investors to invest, and embed Ponzi scheme structures in the contract code to amass money. In this regard, we summarize several characteristics of the Ponzi scheme.

(i) There are bombastic descriptions of blockchain's features on the project website (if any). Exploit high-reward risk-free false advertising to attract investors without providing important information, such as the project operator.

(ii) Investors' returns are mainly supported by the capital invested in by new investors, without real technical support for the project.

(iii) Ethereum contract code contains a hierarchy, a return mechanism in which new investors pay fees to early investors.

4. PD-SECR Detection Method

In this section, we introduce the proposed PD-SECR detection method. It includes the overall process, feature extraction, data preparation based on the SMOTEENN algorithm, and model training.

4.1. Overview of the Entire Work Flow. As illustrated in Figure 2, we give the overall framework of the PD-SECR approach and get transaction information and contract compiled bytecode from Ethereum.io. Corresponding account features are extracted from transaction information through related calculations. Using a decompiler disassembles bytecode into operation code and builds code features by calculating opcode call frequency. Our original dataset consists of account features and code features. Owing to the severe imbalance of the sample data, in order to solve, we introduce the SMTOEENN data-imbalance processing algorithm. The SMOTEENN not only deals with data imbalance problems but also can avoid data duplication. The processed data were then divided into a training set and test set at a ratio of 4 : 1. First, we constructed a CNN feature extraction model consisting of three convolution layers and two fully connected layers to extract key data features from the datasets. The CNN extracts data features input into the RF model for training the classification model. Test sets evaluate the model's performance when training is complete. To facilitate comparisons with other detection method models, three performance indicators, precision, recall rate, and F1-score, are selected to evaluate the model.

4.2. Feature Selection. This study selects 16 features, including nine opcode features and seven account features, which benefit contract identification in Ponzi schemes. The following describes the opcode and account features in detail.

4.2.1. Opcode Feature. An Ethereum intelligent contract can be forced to execute if the default execution conditions are met. Therefore, a fraudulent mechanism is often included in the code structure of a Ponzi scheme contract. The opcodes also characterize the underlying problems of the contracts. To effectively distinguish Ponzi scheme contracts and normal contracts in the real world, this paper analyzes the types and frequencies of contract opcodes and extracts features from contract codes. This paper counts the frequency of the appearance of different opcodes in the smart contract as the opcode features. As shown in Figure 3, we selected the normal contract with ID 0xd07ce4329-b27eb8896c51458468d98a0e4c0394c and Ponzi scheme contract with ID 0xa9fa83d31ff1cfd14b7f9d17-f02e48dcfd9cb0cb and extracted the relevant opcode features from the source code of the contract for visual analysis.

As shown in Figure 3, there is a significant difference between the Ponzi scheme smart contract and the normal contract opcodes without considering opcodes that occur most frequently, such as PUSH, DUP, and SWAP. The most crucial difference is that the Ponzi scheme contract contains more threatening function codes (e.g., CALLER, EXP, etc.) than the normal contract. The analysis above shows that opcode features may be viable for detecting Ponzi scheme contracts.

4.2.2. Account Features. Ponzi scheme contracts on Ethereum have some differences in account trading characteristics compared to normal contracts. In particular, there are apparent differences in the circulation process of the Ether in contract transactions. Through manual verification, the account characteristics can be summarized as follows:

Only a few early contract participants received a high percentage of returns, with almost all the returns concentrated in the first two contract participants. Specifically, the creators of smart contracts receive the highest returns.

Keeping low balance in many Ponzi scheme intelligent contracts compared to the normal smart contract, generally taking the operation of the rapid distribution of the investment obtained.

Ethereum accounts are divided into external and internal accounts (addresses); for example, when a user creates an address, it is called an external address because it is used to access the blockchain from outside. When we deploy a smart contract to Ethereum, we generate an internal address that acts as a pointer to the running blockchain program (deployed smart contract). We can locate it externally as a

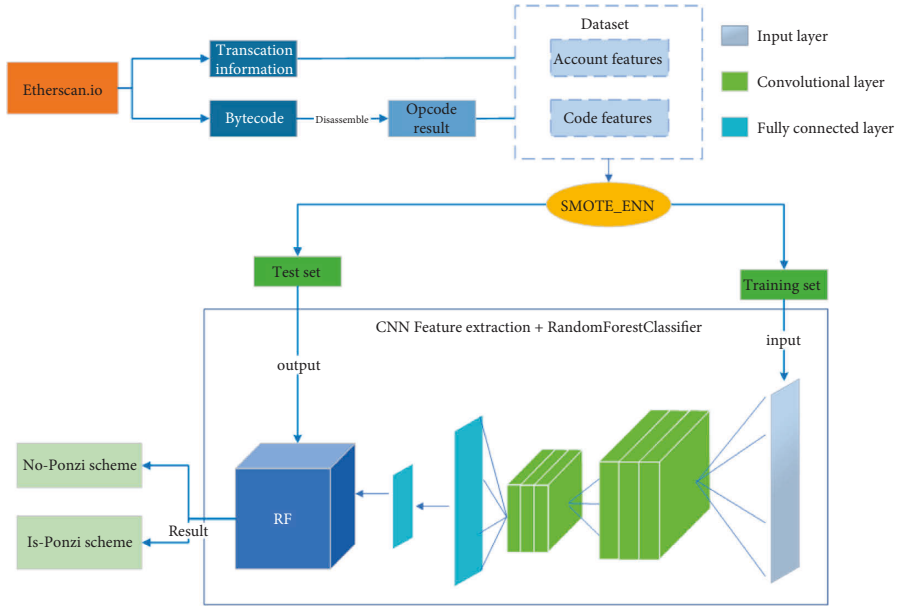


Figure 2: Detection method architecture diagram.

function to be invoked, or internally such that another deployed contract can invoke the function on the deployed contract. Therefore, we extract seven representative characteristics from external and internal account transactions as another manner of identifying a Ponzi scheme contract [7].

Investments_num: The number of investments is received per contract. Assuming that each contract has n transactions, each trade has a transaction account address that initiates the transaction. As shown in Figure 4(a), "from" is the sending account address of the transaction, and then the calculation formula is as follows:

$$\text{Investments\_num} = \sum_{i}^{n} 1^{i}, 1 \le i \le n. \tag{1}$$

Payments_num: The number of transaction payouts per contract. Assuming each contract has p spending, each spending transaction has an account address that receives transaction funds. As shown in Figure 4(b), "to" is the account address that accepts transactions. Then, the calculation formula is as follows:

$$\text{Payments\_num} = \sum_{j}^{p} 1^{j}, 1 \le j \le p. \tag{2}$$

Maxpay: It is the maximum number of transactions that a contract account can pay to the same recipient account. Assuming that there are T expenditures in each contract, and the funds of which I expenditures flow to account S, the Maxpay calculation formula is as follows:

$$\text{Maxpay} = \begin{cases} I, & if\ I \ge \text{other}_{\max}, \\ \text{other}_{\max}, & if\ I < \text{other}_{\max}, \\ 0, & if\ T = 0, \end{cases} \tag{3}$$

where $\text{other}_{\max}$ means that in addition to spending I transaction to account S, the maximum number of transactions paid to the same account.

Rr: Percentage of recipients who have invested before payment. For example, the account address of "to" in Figure 4(b) also exists in "from" in Figure 4(a) and to. timestamp>from. timestamp, and then add 1 to the counter count. Therefore, Rr is the ratio of count to contract expenditure, and the formula is as follows:

$$\text{Rr} = \begin{cases} \dfrac{\text{count}}{\text{Payments\_num}}, & \text{if Payments\_num} \ne 0, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where Payments_num means the number of transaction payouts per contract.

Pr: Percentage of investors who received at least one payment. The calculation method is the same as Rr, but the accumulation conditions of the counter are different. When the "from" address in Figure 4(a) appears once or more in Figure 4(b), the counter will increase by 1. Therefore, the calculation formula of Pr is also formula (4).

A_bal: The balance of the contract account. As shown in Figure 5, the account balance after the contract is traded is obtained from the "Balance" of the contract account information.

D_ind: It is the quantitative difference between payments and investments made by all participants in the contract. Assuming that the smart contract has q participants, V is the vector representation of the length of q, and m_i, n_i represent the investment and payment transaction times of the ith participant, respectively. To calculate the deviation index, we first calculated $V_i = (n_i - m_i)$ to obtain the difference between the number

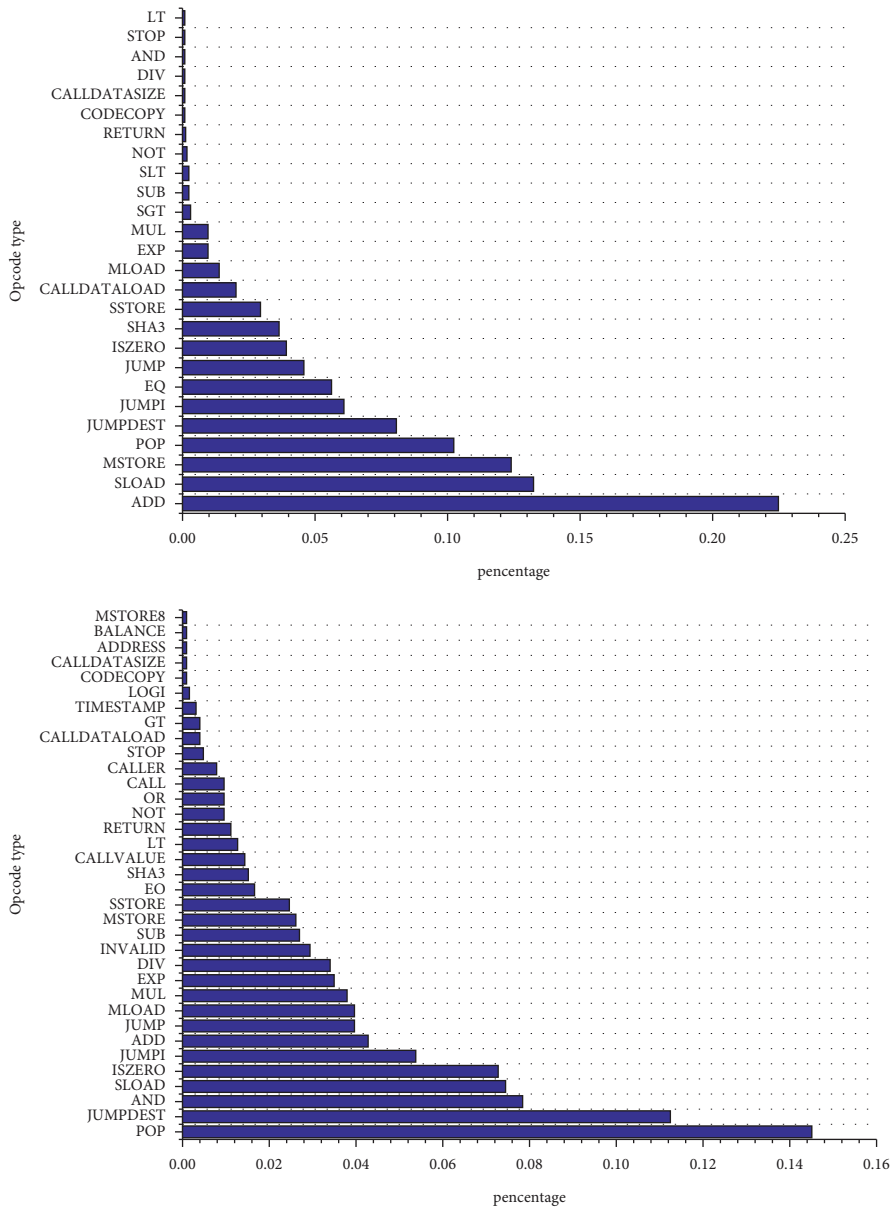


Figure 3: Opcode type ratio of normal contract (a) and Ponzi scheme contract (b).

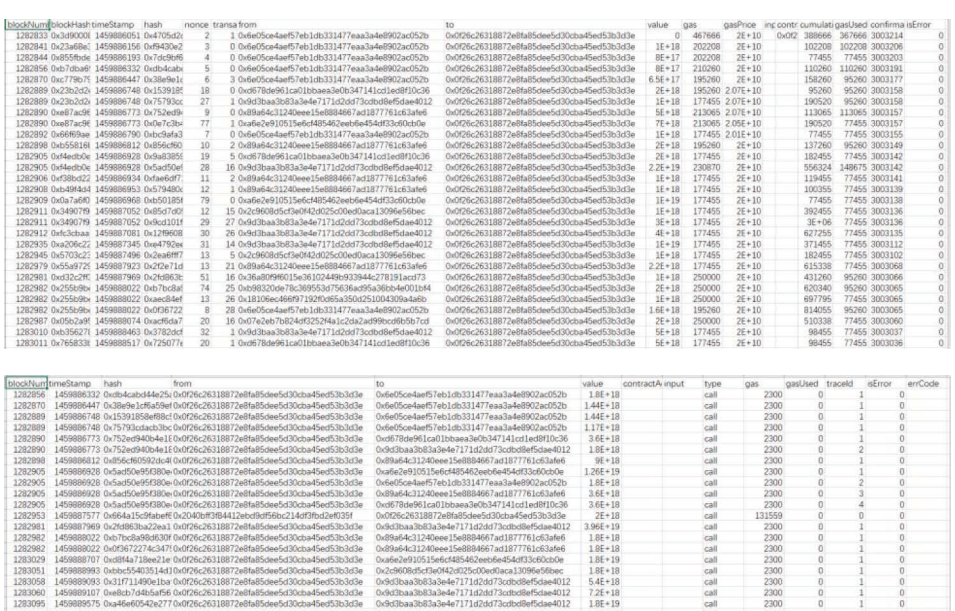
of investment and payment transactions of the participant. Then,

$$D\_ind = f(x) = \begin{cases} 0, & if\ V_i = 0\ or\ q \le 2, \\ s, & q > 2, \end{cases} \tag{5}$$

where s is the skewness of the vector V_i. Typically, the D_ind of Ponzi scheme contracts is negative, and most participants' investments exceed their returns.

Table 1 shows the mean, median, and standard deviation of the account trading characteristics extracted from external and internal trading data. We compare the difference in account characteristics between the Ponzi scheme and normal contracts according to the following table.

As can be seen from Table 1, there are apparent differences between the Ponzi scheme and normal contracts in these seven account characteristics. For example, the balance A_bal of a Ponzi scheme contract is significantly different from that of a normal contract. Because part of the contract balance of the Ponzi scheme is returned to the participants as a reward, the contract balance of the Ponzi scheme is always low. From Investment_num, it can be seen that the mean value of Ponzi scheme contracts is much lower than that of normal contracts. This is because Ponzi scheme contracts rely on the investment of new investors to pay the return fees of previous investors. Thus, the number of subsequent and former investors in Ponzi scheme contracts gradually decreases. In conclusion, these seven characteristics clearly reflect the



blockNum	blockHash	timeStamp	hash	nonce	transa	from	to	value	gas	gasPrice	inp	contr	cumulati	gasUsed	confirma	isError
1282833	0x3d9000(	1459886051	0x4705d2(	2	1	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0	467666	2E+10	0x0f2		388666	367666	3003214	0
1282841	0x23a68e(	1459886156	0xf9430e2	3	0	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	202208	2E+10			202208	202208	3003206	0
1282844	0x855fbde	1459886193	0x7dc9bf6	4	0	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	8E+17	202208	2E+10			77455	77455	3003203	0
1282856	0xb7dba8(	1459886332	0xdb4cabc	5	0	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	8E+17	210260	2E+10			110260	110260	3003191	0
1282870	0xc779b75	1459886447	0x38e9e1c	6	3	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	6.5E+17	195260	2E+10			158260	95260	3003177	0
1282889	0x23b2d2(	1459886748	0x153918f	18	0	0xd678de961ca01bbaaea3e0b347141cd1ed8f10c36	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2E+18	195260	2.07E+10			95260	95260	3003158	0
1282889	0x23b2d2(	1459886748	0x75793cc	27	1	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2.07E+10			190520	95260	3003158	0
1282890	0xe87ac9f	1459886773	0x752ed9(	9	0	0x89a64c31240eee15e8884667ad1877761c63afe6	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	5E+18	213065	2.07E+10			113065	113065	3003157	0
1282890	0xe87ac9f	1459886773	0x0e7c3bc	77	1	0xa6e2e910515e6cf485462eeb6e454df33c60cb0e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	7E+18	213065	2.05E+10			190520	77455	3003157	0
1282892	0x66f69ae	1459886790	0xbc9afa3	7	0	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2.01E+10			77455	77455	3003155	0
1282898	0xb55816(	1459886812	0x856cf60	10	2	0x89a64c31240eee15e8884667ad1877761c63afe6	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2E+18	195260	2E+10			137260	95260	3003149	0
1282905	0xf4edb0e	1459886928	0x9a8385(	19	5	0xd678de961ca01bbaaea3e0b347141cd1ed8f10c36	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2E+18	177455	2E+10			182455	77455	3003142	0
1282905	0xf4edb0e	1459886928	0x5ac50ef	28	16	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2.2E+19	230870	2E+10			556324	148675	3003142	0
1282906	0xf38bd22	1459886934	0xfae6bf7	11	2	0x89a64c31240eee15e8884667ad1877761c63afe6	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2E+10			119455	77455	3003141	0
1282908	0xb49f4a4	1459886953	0x5794800	12	1	0x89a64c31240eee15e8884667ad1877761c63afe6	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2E+10			100355	77455	3003139	0
1282909	0x0a7a6f0	1459886968	0xb50185f	79	0	0xa6e2e910515e6cf485462eeb6e454df33c60cb0e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+19	177455	2E+10			77455	77455	3003138	0
1282911	0x3490769	1459887052	0x85d7d0f	12	15	0x2c9608d5cf5e0f42d025c00ed0aca13096e56bec	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2E+10			392455	77455	3003136	0
1282911	0x3490769	1459887052	0x9cd101f	29	27	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	3E+18	177455	2E+10			3E+06	77455	3003136	0
1282912	0xfc3cbaa	1459887081	0x12ff60b	30	26	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	4E+18	177455	2E+10			627255	77455	3003135	0
1282935	0xa206c22	1459887345	0xe4792ea	31	14	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+19	177455	2E+10			371455	77455	3003112	0
1282945	0x5703c22	1459887496	0x2ea6fff7	13	5	0x2c9608d5cf5e0f42d025c00ed0aca13096e56bec	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	177455	2E+10			182455	77455	3003102	0
1282979	0x55a972(	1459887923	0x2f2e71d	13	21	0x89a64c31240eee15e8884667ad1877761c63afe6	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2.2E+18	177455	2E+10			615338	77455	3003068	0
1282981	0xd32c2ff(	1459887969	0x2fd863c	51	16	0x36a80f9f6015e36102449b933944c278191acd73	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	250000	2E+10			431260	95260	3003066	0
1282982	0x255dc9b	1459888022	0xb7bc8af	74	25	0xb98320de78c369553d75636ad95a36bb4e001bf4	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2E+18	250000	2E+10			620340	95260	3003065	0
1282982	0x255b9bc	1459888022	0xaec84ef	13	26	0x18106ec466f97192f0d65a350d251004309a4a6b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	250000	2E+10			697795	77455	3003065	0
1282982	0x255b9bc	1459888022	0x0f36722	8	28	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1.66E+18	195260	2E+10			814055	95260	3003065	0
1282987	0x05b2a9f	1459888074	0xacf6da7	20	16	0x07e2eb7b824d9252f4a1c2da2ad99bcd6b5b7cd	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	1E+18	250000	2E+10			510338	77455	3003060	0
1283010	0xb356271	1459888463	0x3782dcf	32	1	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	5E+18	177455	2E+10			98455	77455	3003037	0
1283011	0x7658338	1459888517	0x725077e	20	1	0xd678de961ca01bbaaea3e0b347141cd1ed8f10c36	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	5E+18	177455	2E+10			98455	77455	3003036	0

blockNum	timeStamp	hash	from	to	value	contractA	input	type	gas	gasUsed	traceId	isError	errCode
1282856	1459886332	0xdb4cabd44e25a	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	1.8E+18			call	2300	0	1	0	
1282870	1459886447	0x38e9e1cf6a59ef	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	1.44E+18			call	2300	0	1	0	
1282889	1459886748	0x15391858e68cf	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	1.44E+18			call	2300	0	1	0	
1282889	1459886748	0x75793cdacb3bc	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	1.17E+18			call	2300	0	1	0	
1282890	1459886773	0x752ed940b4e1f	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0xd678de961ca01bbaaea3e0b347141cd1ed8f10c36	3.6E+18			call	2300	0	1	0	
1282890	1459886773	0x752ed940b4e1f	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	1.8E+18			call	2300	0	2	0	
1282898	1459886812	0x856cf60592dc4f	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x89a64c31240eee15e8884667ad1877761c63afe6	9E+18			call	2300	0	1	0	
1282905	1459886928	0x5ad50e95f380e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0xa6e2e910515e6cf485462eeb6e454df33c60cb0e	1.26E+19			call	2300	0	1	0	
1282905	1459886928	0x5ad50e95f380e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x6e05ce4aef57eb1db331477eaa3a4e8902ac052b	1.8E+18			call	2300	0	2	0	
1282905	1459886928	0x5ad50e95f380e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x89a64c31240eee15e8884667ad1877761c63afe6	3.6E+18			call	2300	0	3	0	
1282905	1459886928	0x5ad50e95f380e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0xd678de961ca01bbaaea3e0b347141cd1ed8f10c36	3.6E+18			call	2300	0	4	0	
1282953	1459887577	0x664a15c9fabef	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	2E+18			call	131559	0	0	0	
1282981	1459887969	0x2fd863ba22ea1	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	3.96E+19			call	2300	0	1	0	
1282982	1459888022	0xb7bc8a98d630f	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x89a64c31240eee15e8884667ad1877761c63afe6	1.8E+18			call	2300	0	1	0	
1282982	1459888022	0x0f3672274c347f	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x89a64c31240eee15e8884667ad1877761c63afe6	1.8E+18			call	2300	0	1	0	
1283029	1459888707	0xd8f4a718ee21e	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0xa6e2e910515e6cf485462eeb6e454df33c60cb0e	1.8E+19			call	2300	0	1	0	
1283051	1459888993	0xbbc55403514d1	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x2c9608d5cf5e0f42d025c00ed0aca13096e56bec	1.8E+18			call	2300	0	1	0	
1283058	1459889093	0x31f7114f90e1b	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	5.4E+18			call	2300	0	1	0	
1283060	1459889107	0xe8cb7d4b5af56	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	7.2E+18			call	2300	0	1	0	
1283095	1459889575	0xa46e60542e277	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x9d3baa3b83a3e4e7171d2dd73cdbd8ef5dae4012	1.8E+19			call	2300	0	1	0	
1283100	1459889685	0x6c0e0d78e77bf	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x2c9608d5cf5e0f42d025c00ed0aca13096e56bec	1.8E+18			call	2300	0	1	0	
1283100	1459889685	0x6c0e0d78e77bf	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x20406bff3f84412ebd9df56bc214df3ffbd2ef635f	3.6E+18			call	2300	0	2	0	
1283100	1459889685	0x6c0e0d78e77bf	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x89a64c31240eee15e8884667ad1877761c63afe6	3.96E+18			call	2300	0	3	0	
1283191	1459890956	0xfc92f55c21b8a	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0x36a80f9f6015e36102449b933944c278191acd73	1.8E+18			call	2300	0	1	0	
1283309	1459892521	0xefa1a1c2459381	0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	0xb98320de78c369553d75636ad95a36bb4e001bf4	3.6E+18			call	2300	0	1	0	

Figure 4: External (a) and internal (b) transaction information for 0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e.

Contract	ContractName	Compiler	Balance	TxCount_2017	DateVerifi	Ponzi
0xe1388626C8dE5Fd7E5683A83d8da6a34153e9B18	NiceGuyPonzi	v0.3.0	0 Ether	1	4/6/2016	1
0x37b53b46fa74ac3f9b4340dc5a39aabb0f2afa33	fivetimes	v0.3.0	0.009 Ether	8	4/6/2016	1
0x4028b9672bfd71ba2fcd97a6c82e06f72eaa14ba	theultimatepyramid	v0.3.0	2.0031 Ether	33	4/6/2016	1
0x428Da5ff72D8Be0EfAa85336B6c6A9fC9E0F73fe	NiceGuyPonzi	v0.3.0	0.0225 Ether	13	4/5/2016	1
0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e	**theultimatepyramid**	**v0.3.0**	**0.7416 Ether**	**46**	**4/5/2016**	**1**
0xf835b307bc5348194ae01ed729170c84217ba688	newponzi	v0.3.0	0.58 Ether	34	4/5/2016	1
0xf243688304fa4f6c6efadc22b9c1dd009aa76650	SendlGet2	v0.3.0	0.1 Ether	23	4/5/2016	1
0x89c2352Cb600df56fe4BFB5882caAdEF3E96213f	TwoAndAHalfPonzi	v0.3.0	0.503 Ether	3	4/5/2016	1

Figure 5: Account information of contract 0x0f26c26318872e8fa85dee5d30cba45ed53b3d3e.

Table 1: Statistics of account characteristics of Ponzi scheme contracts (left) and normal contracts (right).

	Ponzi scheme contracts			Normal contracts		
	Mean	Medium	Std	Mean	Medium	Std
Rr	0.21	0.60	0.51	0.10	0.00	0.38
A_bal	3.06	0.01	1.6*e*	55.86	0.00	1.9*e*
Investments_num	44.85	7.00	1.1*e*	589.94	5.00	2.1*e*
Payments_num	263.61	1.00	1.4*e*	136.98	16.00	4*e*.00
Pr	0.32	0.80	0.51	0.11	0.00	0.41
Maxpay	68.25	6.00	1.3*e*	149.71	1.00	8.9*e*
D_ind	0.14	0.00	0.76	-0.04	0.00	0.67

difference between the Ponzi scheme and a normal contract, which can be used as an effective basis for detection.

4.3. Data Preparation Based on SMOTEENN. The dataset Q used in this study consists of account features and opcode features. Assuming that the dataset contains account and code characteristic data of M contracts, then denote the dataset as $Q = \{(x_i, y_i)|i = 1, 2, 3 \cdots, M\}$, where x_i represents the combined feature vector of the account and opcode feature of the ith contract and y_i represents the class label of the ith contract. When $y_i = 0$, the contract category of the combined feature is a normal contract; on the contrary, when $y_i = 1$, the contract category of the combined feature is a Ponzi scheme contract. Here, our normal dataset is represented as Q_n, and the abnormal dataset is represented as Q_f. The "data-description" in Section 5.1 shows that the samples of the two categories differ significantly, which is $Q_n \gg Q_f$.

The SMOTEENN-mixed sampling algorithm is introduced to solve the above data imbalance problem. Previously, oversampling algorithms are used to handle data imbalance, but the oversampling algorithms suffer from data overlap problems. Therefore, we use a combination of oversampling and data cleaning for data processing, which can solve data imbalance overlap after oversampling.

Specifically, first, we use the SMOTE algorithm to generate a new minority class Q_f sample to obtain the expanded dataset W. The formula for the sample generation is as follows:

$$x_{new} = x_i + (\hat{x}_l - x_i) \times \varepsilon, \tag{6}$$

where x_i means a sample point in the minority class, $\hat{x}_l$ means a sample point randomly selected from the K-nearest neighbors, and $\varepsilon \in [0, 1]$ is a random number.

Second, we utilize the Edited Nearest Neighbor (ENN) algorithm to clean the dataset W to get a new dataset. The core idea of ENN is to employ the K-nearest neighbor algorithm to calculate the categories of features. The sample is eliminated if the predicted result is inconsistent with the actual category label. For example, $W = \{(s_j, t_j)|j = 1, 2, \cdots, N\}$ serves as the input dataset for ENN, where $s_j \in S \subseteq R^n$ is an eigenvector of instances and $t_j \in T = \{0, 1\}$ is the category label. According to the distance measure, the k points nearest to s are denoted by $N_k(s)$. According to the classification decision rule in $N_k(s)$, the category t that determines s is computed by the following formula:

$$t = \text{argmax}_{c_g} \sum_{s_j \in N_k(s)} I\left(t_j = c_g\right), j = 1, 2, \cdots, N, \tag{7}$$

where I is the indicator function, that is, when $t_j = c_g$, $I = 1$; and c_g means the category label 0 or 1. If t is equal to the actual class label t_j of s_j, the data will be retained; otherwise, the data is excluded from dataset W.

Finally, the detection model is trained with the cleaned dataset W as input. Data preparation is based on SMOTEENN, as Algorithm 1 shows.

4.4. CNN-RF. The effective fusion of convolutional neural networks and random forest models is also a significant challenge for this study. As we all know, CNN is the most widely used image classification, and more and more researchers migrate it to text classification because of its robust feature extraction function. In this study, we are also inspired by this and try to fuse CNN with a classic machine learning model—random forest. When studying model fusion, we found that neural network and machine learning algorithm models cannot be embedded and fused in the model structure like other deep learning models. Therefore, in this study, we train CNN and RF separately and embed the trained CNN feature extraction model into the training and testing of the RF model. The embedding of the CNN feature extraction model improves the training and detection speed of the RF model.

In this study, the CNN-RF is used to model the identification of Ponzi scheme contracts, as shown in Algorithm 2. In essence, identifying and detecting Ponzi scheme contracts is a dichotomy problem. The optimal scheme is selected based on previous experience and existing datasets. Finally, CNN is chosen as the feature extractor and RF as the classifier to train the joint model CNN-RF. As a feature extraction model, the CNN can effectively avoid redundancy and automatically select the most critical and decisive features from the 16 selected feature species. In addition, because the sample size of the dataset we used is small, the CNN classification algorithm can easily lead to model overfitting and reduce the generalization ability.

Therefore, we only use CNN to extract critical features, and the features extracted by CNN are used as input data to train the RF classification model. The RF model has the best effect on binary classification detection. Therefore, the proposed joint CNN and RF detection method improves the model's generalization ability and makes it more suitable for the identification and detection of the Ethereum Ponzi scheme. Figure 6 illustrates the model construction.

Figure 6 illustrates the principle of the model in detail:

(1) CNN feature extraction: The feature extraction process is shown in Figure 7. The central role of CNN in this study is feature extraction, so our CNN model structure consists only of convolutional and fully connected layers.

First, input the pre-processed feature sequence matrix with $n \times 1$ as En:1 and then pad the feature sequence to perform the convolution operation better. Second, the input feature sequence of this study can be regarded as an $n \times 1$ single-channel feature map. In the single-channel convolution calculation, each filter has a $k \times 1$ convolution kernel. The filter at this time is the convolution kernel, and they have the same dimension size. In the convolution layer, to extract vital local features, J filters of the same size are convolved on matrix En:1. The width of each filter window is the same as En:1; only the height is different. This way, the filter can obtain the relationship of other elements in the same feature sequence. The convolutional neural networks learn parameters in the convolutional kernel, and each filter has its focus so that multiple filters can learn various pieces of information. The convolution calculation formula for feature extraction is as follows:

$$C_{out} = f\left(\sum_{k=0}^{k=n} E_{n-k:1} \times W + b\right), \tag{8}$$

where $W \in R^{k \times 1}$ denotes the weight of the filter in the convolution operation, C_{out} is the new feature resulting from the convolution operation, $b \in R$ is a bias, and f is a non-linear function. Finally, the convolutionally extracted key features are stitched into feature map outputs using fully connected layers.

(2) CNN model training: In the training process of the CNN feature extraction model, the training model with the best detection performance is saved as best.pt, which uses key features to train the classification models more conveniently.

To further optimize the feature extraction model, we selected the loss function (cross_entropy loss function) with the fastest weight updating speed to update the weight of the model parameters. Cross-entropy was used to evaluate the difference between the probability distribution currently trained and real distribution. The smaller the loss value, the

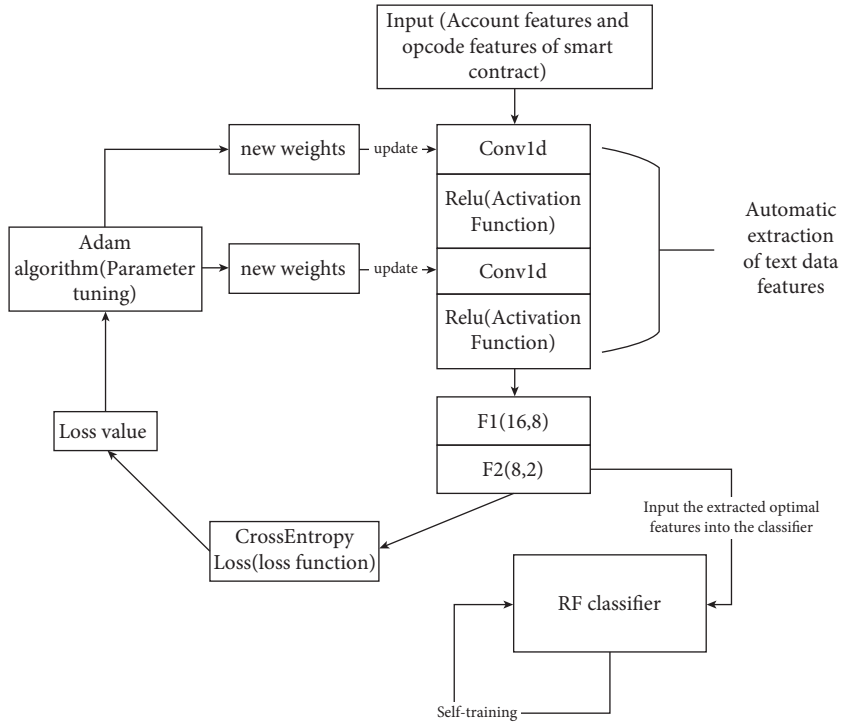


Figure 6: CNN-RF recognition model construction.

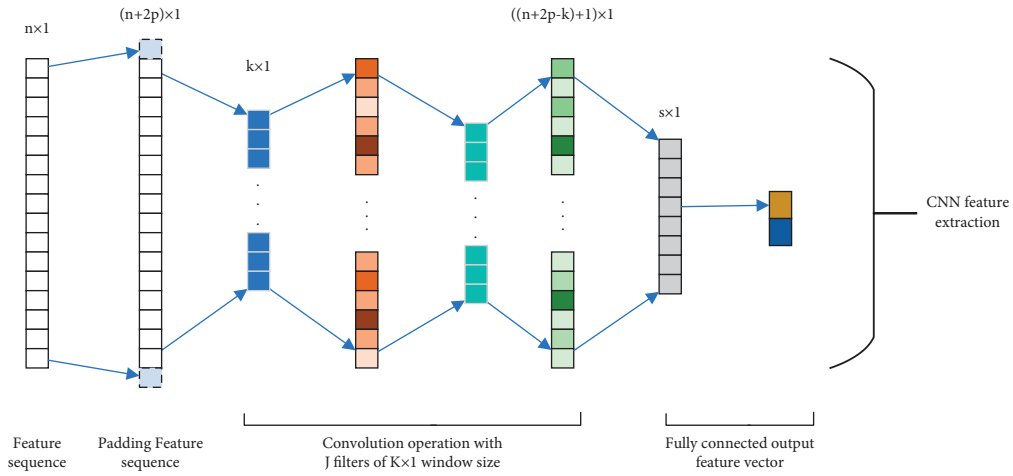


Figure 7: Feature extraction process of CNN.

better the performance of the training model. The cross-entropy loss function formula adopted in this study is as follows [32]:

$$L = \begin{cases} -\log\hat{y}, & if\ y = 1, \\ -[y\ \log\hat{y} + (1-y)\log(1-\hat{y})], & if\ y \neq 1\ and\ y \neq 0, \\ -\log(1-\hat{y}), & if\ y = 0, \end{cases} \tag{9}$$

where y is the actual true value and $\hat{y}$ is an estimate. For example, when $y = 1$, the closer $\hat{y}$ is to 1, the smaller the loss value, and the better the feature extraction model performance. Otherwise, the performance of the feature extraction model deteriorates.

In addition, we used the Adam optimizer. The network parameters are input into the optimizer before model training, and the Adam algorithm are used to calculate the gradient of the backpropagation function. The parameters are updated once in each training batch and updated dynamically during model training. Simultaneously, to improve the learning rate of the feature extraction model, a scheduler is also set to attenuate the learning rate.

(3) RF classification detection model training: The best.pt saved in the CNN model training and training

dataset after partitioning was loaded into the RF model, and the RF model was trained independently. When RF classification model training completed, the loaded test dataset is used to evaluate the performance of the classification model.

5. Experiment

In this section, we evaluate the performance of the PD-SECR method using a series of experiments. First, we established evaluation indicators and verified the feasibility of the proposed model using test datasets. The results show that the proposed method is superior to previous methods.

5.1. Data Description. This paper selects a validated sample dataset from previous studies [10], which provides some contract addresses and their categories. It crawls data from Ethereum.io according to the contract address. Calculate and process the acquired data, and filter the account and opcode features we need. Note that the cleaned data has a severe data sample imbalance problem, as shown in Figure 8. The positive and negative sample ratio is approximately 12 : 1. If a dataset with unbalanced samples trains the model, it will lead to a significant detection bias. Therefore, data enhancement processing is performed on the dataset to improve detection accuracy. The processing process is described in detail in Section 4.3.

5.2. Parameter Settings. In the summary of feature selection, in Section 4.2, it can be concluded that our input feature sequence has a length of 16 (that is, the 16 kinds of combined features selected) and a width of 1 (that is, a feature sequence), so our input feature sequence is a 16×1 matrix. Then, from the description of the feature extraction process in Section 4.4, our input feature sequence is a single-channel sequence. That is, the filter is the convolution kernel. Due to the limitation of the feature sequence dimension, the most suitable convolution kernel size is selected as 3×1 in this experiment. Therefore, in the following parameter adjustment experiments, the structure of our feature extraction model and the output dimensions and trainable parameters of each layer of features can be observed in Figure 9.

5.2.1. Impact of Parameter Epochs. In this round of experiments, we pre-set the parameters as follows: combination number = 3, test_size = 0.2, and batch_size = 25. We continuously changed the number of training rounds, starting from epochs = 25 and increasing it by 40 each time. Six experiments were performed, and each experiment of the same dataset was repeated 10 times. The mean value was obtained after removing the lowest and highest values. Figure 10 shows the comparison results.

From Figure 10, the precision of the model fluctuated with an increase in training rounds. When the epochs are 50 and 140, the accuracy of the model reaches the peak of this experiment round. However, because too few rounds lead to inaccurate model training, we set the epochs of the model to 140 to avoid contingency in the experiment.

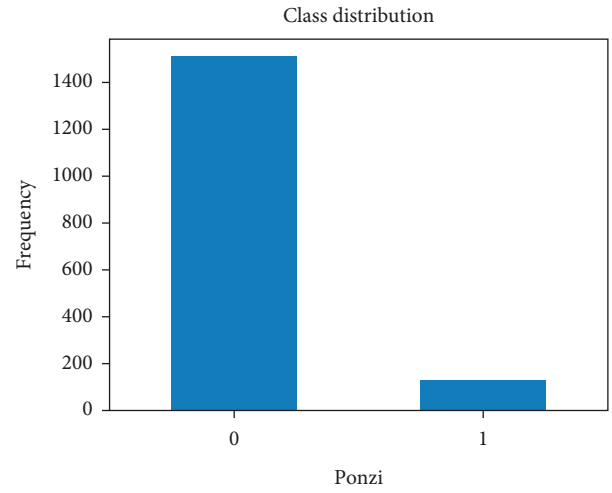


Figure 8: Sample category distribution.

Layer (type)	Output Shape	Param #
Conv1d-1	[−1, 1, 16]	4
ReLU-2	[−1, 1, 16]	0
Conv1d-3	[−1, 1, 16]	4
ReLU-4	[−1, 1, 16]	0
Linear-5	[−1, 8]	136
Linear-6	[−1, 2]	18

Figure 9: Output feature dimension and trainable parameters at each layer.

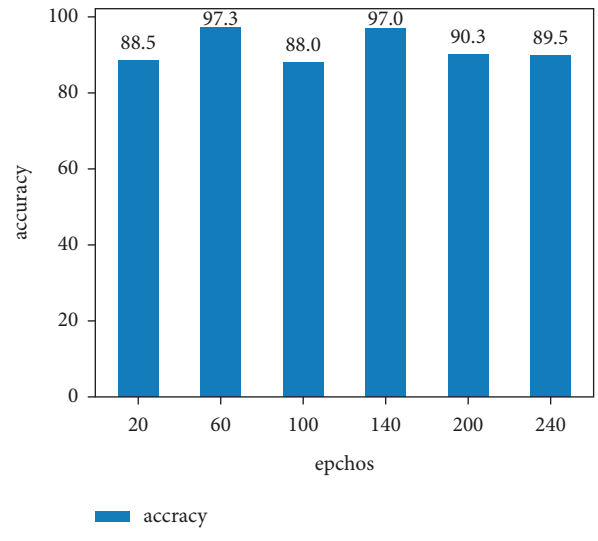


Figure 10: Impact of epochs on accuracy.

5.2.2. Impact of Parameter Batch_size. Batch_size is essentially a gradient descent algorithm. Batch_size determines the time required to complete each epoch and degree

of gradient smoothing between each iteration during deep learning training. In this module experiment, we changed the default parameters to epochs = 140, combination number = 3, and test_size = 0.2. The batch size was constantly changed for each training round. Starting with a batch_size of 10, the batch size was increased by 5 in each round of experiments. Six experiments were conducted, and each experiment of the same data parameter group was repeated 10 times. One maximum and one minimum value were removed, and their average values were obtained. Figure 11 shows the comparison results.

From Figure 11, the precision of the model increases with an increase in batch_size before batch_size is less than 20. However, when batch_size is greater than 20, the precision gradually decreased with an increase in batch_size; in particular, when batch_size was greater than 30, the precision dropped sharply. When batch_size is 20, the model has the highest accuracy. Therefore, the single-round batch size of the model was set to 20.

5.2.3. Impact of Test Set Partition Ratio. During the training process, the division of the test sets affects the performance index of the model. Therefore, we tested the increase in the accuracy by constantly changing the division ratio of the test set. First, we fixed the epochs, batch_size, and combination number to 140, 20, and 3, respectively. Second, we repeatedly changed the division ratio of the test set. In this experiment, the division ratio range of the test set was {0.1, 0.2, 0.3, 0.4, 0.5}. The experiment of the same parameter group was repeated 10 times each time, and the mean value was taken after removing the lowest and highest values. Figure 12 shows the comparison results.

As shown in Figure 12, the accuracy of the training model fluctuated when different proportions of the test sets were divided. When the proportion of the test sets exceeded 0.2, the accuracy of the model showed a decreasing trend. The accuracy was the lowest when the test set ratio was 0.3. Evidently, the accuracy rate decreases continuously with an increase in the proportion of test_size. Therefore, the model is the most accurate when test_size is 0.2. Therefore, we set the scale of the test set of the model to 0.2 here.

5.2.4. Impact of Convolution Layer and Combination Number of ReLu Function in Sequential Module. During model construction, the performance of the model can be debugged by changing its building blocks. Therefore, we searched for appropriate model construction parameters by changing the convolution layer and combination number of ReLu functions in the sequential module. In this experiment round, we fixed the default parameters to epochs = 140, batch_size = 20, and test_size = 0.2. We set the value range of the combination number of the convolution layer and ReLu function in the sequential module to be between 1 and 6 and conducted six experiments. The experiment was repeated 10 times for each experiment with the same parameters; one minimum and one maximum value were removed, and their mean values were obtained. Figure 13 presents the comparison results.

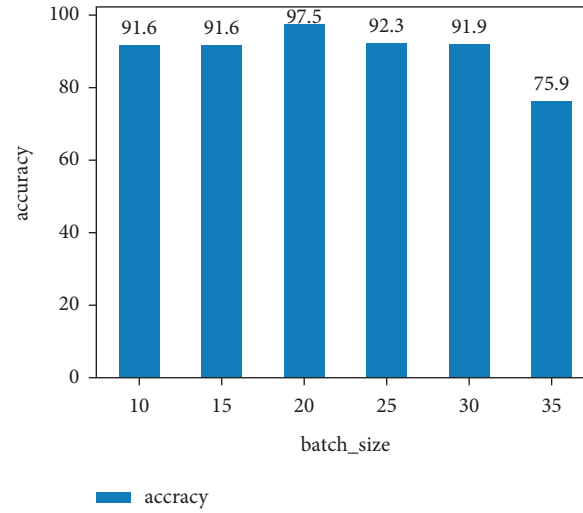


Figure 11: Impact of batch_size on accuracy.

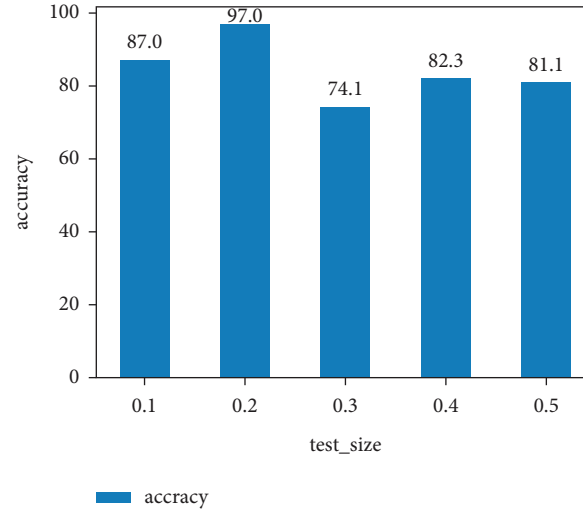


Figure 12: Impact of test_size on accuracy.

Figure 13 shows that the convolution layer and combination number of ReLu functions in the sequential module also significantly influence the training index of the model. When the number of combinations exceeds 2, the accuracy of the model decreases entirely. Because the training dataset is small, too many model structures lead to over-fitting. Therefore, the combination number of the convolution layer and ReLu function in the sequential module is set to 2, and the detection of prediction classification is performed under the condition that the training model is over-fitting.

In summary, after several adjustments to the experimental data, the final training parameters of our model were set as epochs = 140, batch_size = 20, test_set = 0.2, and combination number = 2.

5.3. Performance Metric. To facilitate the performance comparison with the Ponzi scheme contract detection and recognition model adopted by other methods, in this experiment, in addition to more intuitive training accuracy,

input: $Q = \{(x_i, y_i)\}_{i=1,\ \delta}^{m}$ is a random, k points nearest to s are denoted as $N_k(s)$, epochs is the number of training wheels, y is the actual true value and $\hat{y}$ is an estimate value
output: The new dataset W after SMOTEENN processing
(1) Generate new sample x_{new} and perform k_nearer neighbor calculation on x_i;
(2) Generate dataset W from a minority class samples of the input dataset Q;
(3) **for**$i \leftarrow 1$ to range(m) **do**
(4) Find the k minority class samples closer to x_i;
(5) $W = []$;
(6) $x_{new} = x_i + (\hat{x}_l - x_i) \times \delta$;
(7) W.append (x_{new});
(8) return W;
(9) Clean the newly generated dataset $W = \{(s_i, t_i)\}_{j=1}^{n}$;
(10) Calculate the category t of s, according to the classification decision rules in N_k(s);
(11) **for**$j \leftarrow 1$ to range(n) **do**
(12) $t = \text{argmax}_{c_g} \sum_{s_j \in N_k(s)} I(t_j = c_g), j = 1, 2, \ldots, n$;
(13) **if**$t \neq 1$ or $t \neq 0$**then** delete it;
(14) **else** do nothing;
(15) return W processed by us;

Algorithm 1: Data preparation based on SMOTEENN.

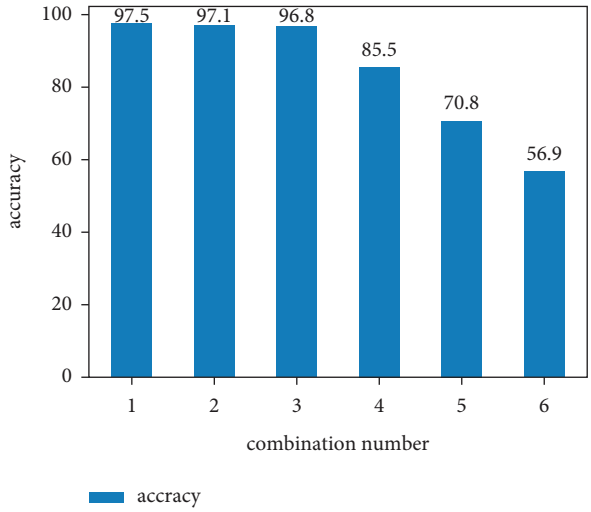


Figure 13: Impact of combination number on accuracy.

verification accuracy, and test accuracy output, 3 indexes, precision, recall, and F1-score, were used to measure the performance of the model. The definitions of these three indicators are as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{10}$$

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

whereTP represents the number of positive samples detected, and the actual samples are positive samples; FP refers to the number of samples that are actually negative but are detected as positive; and FN represents the number of samples that are actually positive but detected as negative.

5.4. Approaches Performance Comparison

5.4.1. Compare with Other approaches. This study uses the proposed PD-SECS method to detect Ponzi scheme contracts and replicates traditional machine learning classification algorithms. For example, extreme gradient enhancement (XGBoost), random forest (RF), lightweight gradient enhancement decision tree (LightGBM), linear support vector machine (LinearSVC), and decision tree (DT) were used to compare and measure the applicability of our proposed PD-SECS method to Ponzi scheme detection. Precision, recall, and F1-score were used to measure the accuracy of the above method in identifying the Ponzi scheme contract, and Table 2 lists the results.

As shown in Table 2, compared with several traditional machine learning models, the proposed PD-SECR method has an obvious improvement in the three performance evaluation indicators. In particular, recall and F1-score both reached over 96%, which indicates that our model has excellent performance in Ethereum intelligent Ponzi scheme detection. Second, the improvement in recall indicates that the quotient of the number of correct samples detected by our model method divided by the number of all correct samples is high, that is, the number of correct samples detected by us is higher. In summary, our proposed method is more suitable for detecting Ponzi scheme contracts on Ethereum than the previous typical machine learning models.

5.4.2. Comparison of Different Data Augmentation Algorithms. The imbalanced dataset is pre-processed in this experiment using the SMOTEENN-mixed sampling algorithm. Therefore, we need to observe whether the algorithm can significantly improve the detection effect of the model. Then, in Figure 14, we draw the confusion matrix diagrams of the models using the SMOTEENN and SMOTE algorithms, respectively, and compare the performance

input:
epochs number of training rounds feature, label features and labels of the input data set.
W processed data set by SMOTEENN
output:
The detection result, output the category 0 or 1 of the predicted feature
(1) W is divided into Train set W_{train} and Test set W_{test} and save as PKl file;
(2) **if** is_balance = true **then** load_balance.pkl;
(3) **else** load imbalance.pkl;
(4) setup $loss\_func = -y\ \log_2 \hat{y} + (1-y)\log_2(1-\hat{y})$;
(5) **for** epoch in range(epochs) **do**
(6) **for** feature, label in W_{train}**do**
(7) Training feature extraction model CNN;
(8) $loss = loss\_func(feature, label)$;
(9) setup *optimizer*;
(10) Evaluate the feature extraction model CNN;
(11) save best feature extraction model as best.pt and return it;
(12) setup *scheduler*
(13) Training classification detection model RF;
(14) **if** is_balance = true **then** load_balance.pkl;
(15) **else** load imbalance.pkl;
(16) load(best.pt);
(17) $clf = fit(classifier) \leftarrow W_{train} - feature, W_{train} - lable$;
(18) Classification model for classification detection;
(19) $pred = clf.predict(W_{test})$;
(20) **if**$pred > 0.6$**then** return 1;
(21) **else** otherwise return 0;

Algorithm 2: CNN-RF detection method.

Table 2: Comparison of the detection performance of various methods.

Method	Precision (%)	Recall (%)	F1-score (%)
XGBoost	83	69	75
RF	89	62	73
LightGBM	83	70	73
LinearSVC	64	58	60
DT	64	67	65
PD-SECR	98	99	98

Table 3: Comparison of different data augmentation algorithms.

Method	Precision (%)	Recall (%)	F1-score (%)	Test time
SMOTEENN + CNN_RF	98	98	98	0.56
SMOTE + CNN_RF	96.7	97	98	0.63

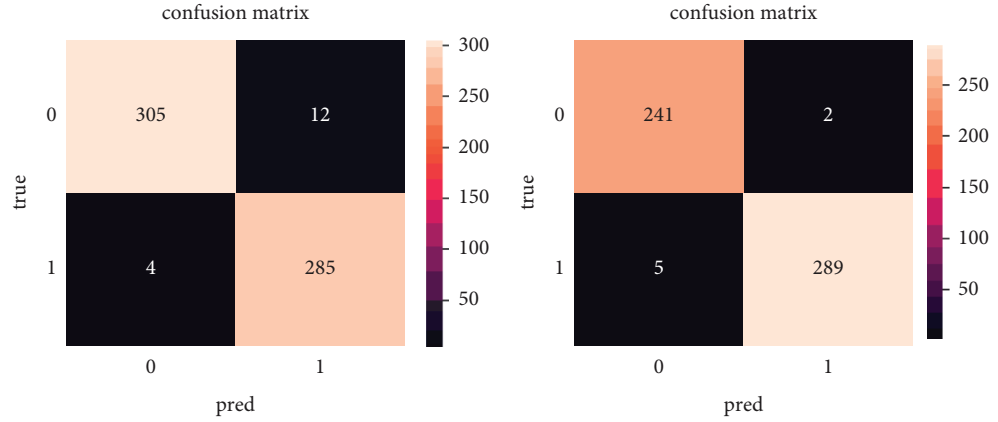


Figure 14: SMOTE (a) and SMOTEENN (b) confusion matrix.

Table 4: Comparison with and without CNN feature extractor.

Method	Precision (%)	Recall (%)	F1-score (%)	Test time
SMOTEENN + CNN_RF	98	98	98	0.65
SMOTEENN + RF	97	97	98	0.59

indicators of SMOTEENN + CNN_RF and SMOTE + CNN_RF, for example, Table 3. First, as shown in Figure 14, the data processed by SMOTEENN is less than that processed by SMOTE because the ENN data cleaning operation in the SMOTEENN algorithm removes the duplicate data generated by SMOTE. Second, Table 3 shows that SMOTEENN + CNN_RF is slightly higher than SMOTE + CNN_RF in terms of precision and recall. In addition, SMOTEENN + CNN_RF is also better than SMOTE + CNN_RF in the test time of the model. Therefore, the model combined with the SMOTEENN algorithm is more suitable for Ponzi scheme contract detection in Ethereum.

5.4.3. Comparison with and without CNN Feature Extractor. In addition to the above experimental comparisons, to verify the rationality and superiority of the process of introducing CNN feature extraction, we conducted a comparative experiment between SMOTEENN + CNN_RF and SMOTEENN + RF. The experimental results are shown in Table 4. The experimental results show that the model without a CNN feature extractor is faster in the model's test time. However, it is still slightly lower in precision and recall detection indicators. On the whole, SMOTEENN + CNN_RF is more suitable for Ponzi scheme detection.

6. Conclusions and Future Work

The decompiled bytecode and account features constitute this study's dataset for Ethereum-style scam detection. The diversity of mixed features and the degree of influence of mixed features on Ponzi scheme detection performance is different. Hence, feature extraction is redundant and imprecise from the acquired source data. Therefore, we use a CNN model that combines features to automatically extract the key and powerful features and identify a Ponzi scheme. Then, the features extracted by the CNN were input into an RF classifier as new input data for classification and prediction. In addition, because of the severe imbalance of sample data in the dataset we obtained, the SMOTEENN-mixed sampling algorithm was used to pre-process the data. The results show that our proposed PD-SECR (CNN-RF joint detection method: integrating unbalanced data and pre-processing algorithm) is more suitable for Ponzi scheme detection on Ethereum.

In future research, we plan to use generative adversarial networks (GANs) to detect contracts. GANs can effectively solve the problem of the high-dimensional feature distribution of data, which means that they can be used for anomaly detection in line with the application scenario of the identification and detection of Ponzi scheme contracts. Because GANs are a type of unsupervised learning, they can realize the clustering of normal contracts, which effectively solves the sample imbalance problem in the process of identifying contracts in the Ponzi scheme.

Data Availability

The data used to support the findings of the study can be obtained from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62102209), the Shandong Provincial Natural Science Foundation of China (ZR2020KF035), the National Key Research and Development Program of China (2018YFB0804104), and the Shandong Provincial Key Research and Development Program (2021CXGC010107 and 2020CXGC010107).

References

[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[2] W. Zheng, Z. Zheng, X. Chen, K. Dai, P. Li, and R. Chen, "Nutbaas: a blockchain-as-a-service platform," *IEEE Access*, vol. 7, pp. 134422–134433, 2019.

[3] C. Linnhoff-Popien, R. Schneider, and M. Zaddach, *Digital Marketplaces Unleashed*, Springer, Berlin, Heidelberg, 2018.

[4] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[5] V. Buterin, *A Next-Generation Smart Contract and Decentralized Application Platform*, white paper, 2014.

[6] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F. Y. Wang, "Blockchain-enabled smart contracts: architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.

[7] M. Vasek and T. Moore, "There's no free lunch, even using Bitcoin: tracking the popularity and profits of virtual currency scams," *International conference on financial cryptography and data security*, vol. 8975, pp. 44–61, 2015.

[8] S. King and S. Nadal, *Ppcoin: Peer-To-Peer Crypto-Currency with Proof-Of-Stake*, 2012.

[9] S. Corbet and, Ed., *Understanding Cryptocurrency Fraud: The Challenges and Headwinds to Regulate Digital Currencies*, Walter de Gruyter GmbH & Co KG, vol. 2, , 2021.

[10] M. Vasek and T. Moore, "Analyzing the bitcoin ponzi scheme ecosystem," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, vol. 10958, pp. 101–112, Springer, Berlin, Heidelberg, 2018.

[11] P. Monamo, V. Marivate, and B. Twala, "Unsupervised learning for robust Bitcoin fraud detection," in *Proceedings of*

the Information Security for South Africa (ISSA), pp. 129–134, Johannesburg, South Africa, August 2016.

[12] M. Bartoletti, B. Pes, and S. Serusi, "Data mining for detecting bitcoin ponzi schemes," in *Proceedings of the 2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pp. 75–84, Zug, Switzerland, June 2018.

[13] P. Nerurkar, S. Bhirud, D. Patel, R. Ludinard, Y. Busnel, and S. Kumari, "Supervised learning model for identifying illegal activities in Bitcoin," *Applied Intelligence*, vol. 51, no. 6, pp. 3824–3843, 2021.

[14] W. Chen, Z. Zheng, and J. Cui, "Detecting ponzi schemes on ethereum: towards healthier blockchain technology," in *Proceedings of the 2018 world wide web conference*, pp. 1409–1418, Lyon, France, April 2018.

[15] M. Bartoletti, S. Carta, T. Cimoli, and R. Saia, "Dissecting Ponzi schemes on Ethereum: identification, analysis, and impact," *Future Generation Computer Systems*, vol. 102, pp. 259–277, 2020.

[16] Y. Zhang, W. Yu, Z. Li, S. Raza, and H. Cao, "Detecting ethereum Ponzi schemes based on improved LightGBM algorithm," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 2, pp. 624–637, 2022.

[17] S. Fan, S. Fu, and H. Xu, "Al-SPSD: anti-leakage smart Ponzi schemes detection in blockchain," *Information Processing & Management*, vol. 58, 2021.

[18] H. Han, W. Y. Wang, and B. H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning," in *Proceedings of the International conference on intelligent computing*, vol. 3644, pp. 878–887, Springer, Berlin, Heidelberg, August 2005.

[19] T. Moore, J. Han, and R. Clayton, "The postmodern Ponzi scheme: empirical analysis of high-yield investment programs," in *Proceedings of the International Conference on financial cryptography and data security*, vol. 7397, pp. 41–56, Springer, Berlin, Heidelberg, March 2012.

[20] Y. Lou, Y. Zhang, and S. Chen, "Ponzi contracts detection based on improved convolutional neural network," in *Proceedings of the 2020 IEEE International Conference on Services Computing (SCC)*, pp. 353–360, Beijing, China, March 2020.

[21] L. Wang, H. Cheng, Z. Zheng, A. Yang, and X. Zhu, "Ponzi scheme detection via oversampling-based Long Short-Term Memory for smart contracts," *Knowledge-Based Systems*, vol. 228, Article ID 107312, 2021.

[22] S. Wang, M. Huang, and Z. Deng, "Densely connected CNN with multi-scale feature attention for text classification," in *Proceedings of the International Joint Conferences on Artificial Intelligence Organization (IJCAI)*, pp. 4468–4474, Melbourne, Australia, August 2018.

[23] B. Guo, C. Zhang, J. Liu, and X. Ma, "Improving text classification with weighted word embeddings via a multi-channel TextCNN model," *Neurocomputing*, vol. 363, pp. 366–374, 2019.

[24] S. Yang, L. Gu, X. Li, T. Jiang, and R. Ren, "Crop classification method based on optimal feature selection and hybrid CNN-RF networks for multi-temporal remote sensing imagery," *Remote Sensing*, vol. 12, no. 19, 2020.

[25] G. Kwak, C. Park, K. Lee, Si Na, Hy Ahn, and N. W. Park, "Potential of hybrid CNN-RF model for early crop mapping with limited input data," *Remote Sensing*, vol. 13, no. 9, p. 1629.

[26] N. Szabo, "Smart contracts: building blocks for digital markets," *EXTROPY: The Journal of Transhumanist Thought*, vol. 18, no. 2, p. 28, 1996.

[27] V. Buterin, "Ethereum white paper," *GitHub repository*, vol. 1, pp. 22-23, 2013.

[28] A. Ekblaw, A. Azaria, and J. D. Halamka, "A Case Study for Blockchain in Healthcare:"MedRec" prototype for electronic health records and medical research data," *Proceedings of IEEE open & big data conference*, vol. 13, p. 13, 2016.

[29] M. Du, Q. Chen, J. Xiao, H. Yang, and X. Ma, "Supply chain finance innovation using blockchain," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1045–1058, 2020.

[30] O. Novo, "Blockchain meets IoT: an architecture for scalable access management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.

[31] G. Praveen, V. Chamola, V. Hassija, and N. Kumar, "Blockchain for 5G: a prelude to future telecommunication," *Ieee Network*, vol. 34, no. 6, pp. 106–113, 2020.

[32] W. Shang, K. Sohn, and D. Almeida, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *Proceedings of the International conference on machine learning. PMLR*, pp. 2217–2225, Las Vegas, Nevada, 2016.