Ponzi Scheme Detection via Oversampling-based Long Short-Term Memory for Smart Contracts

Lei Wang[a], Hao Cheng[a], Zibin Zheng[b], Aijun Yang[a], Xiaohu Zhu[c]

[a]*Department of Management Science and Engineering, Nanjing Forestry University, Longpan Road 159, Nanjing 210037, China.*
E-mails: leiwangchn@163.com (L. Wang); 15150622539@163.com (H. Cheng); ajyang81@163.com (A. Yang).
[b]*School of Software Engineering, Sun Yat-sen University, China.*
E-mail: zhzibin@mail.sysu.edu.cn.
[c]*Center for Safe AGI, Lingshi Road 695, Shanghai, 200000, China.*
E-mail: xhzhu.nju@gmail.com.

Abstract

The application of blockchain technology is growing rapidly, which has aroused great attention in the academic and industrial fields. Based on blockchain 2.0, Ethereum is a mainstream smart contract development and operation platform. The trading process of Ethereum users is facing a serious threat of financial fraud. In particular, the Ponzi scheme is a classic form of fraud. Relevant works have investigated the issue of Ponzi schemes smart contract detection on Ethereum based on machine learning approaches. Nevertheless, the detection approaches still fall short in dealing with the big data-space Ponzi scheme smart contract detection application based on the class-imbalanced training data. We propose PSD-OL, a Ponzi schemes detection approach based on oversampling-based Long Short-Term Memory (LSTM) for smart contracts in this paper. PSD-OL takes the contract account features and the contract code features together into consideration. Oversampling technique is utilized to fill the class-imbalanced Ponzi scheme smart contracts' sample feature data. An LSTM model is trained by learning from the feature data for future Ponzi scheme detection. Experimental results conducted on the well-known XBlock dataset demonstrate the effectiveness of the proposed method.

Keywords: Blockchain, Ethereum, Long Short-Term Memory (LSTM), Oversampling, Ponzi Schemes, Smart Contract

1. Introduction

Bitcoin, as the first electronic cryptocurrency, makes anonymous payments on the Internet possible, therefore makes the transfer of value easier [1]. As the underlying technique of bitcoin, blockchain has become a research hotspot at present, and the application scenarios are increasing rapidly [2, 3]. In particular, blockchain is a point-to-point distributed accounting technology, which is essentially an internet shared database. It records all the value transfer processes of bitcoin transactions based on a cryptography algorithm [4, 5]. Therefore, bitcoin has been operating steadily without any central institutions managing it.

To facilitate the bitcoin-based transactions, the smart contract is proposed. Specifically, the smart contract establishes a computer protocol between untrusted participants [6]. Once the written preconditions are met, the contract runs automatically and can not be terminated [7, 8, 9, 10]. In particular, Ethereum provides a suitable platform for smart contracts and opens the era of blockchain 2.0 [11], which has successfully attracted the attention of a large number of investors [12].

Due to the features of intermediary, good traceability and tamper-resistance, blockchain technology has a profound potential for change in finance, science and technology [5, 13]. With the popularization of decentralized applications (DAPP), the number of blockchain smart contracts has been increasing, and the number of digital assets involved in the contracts has increased exponentially [14].

Because of the complexity of this new technology and the lack of supervision, the growing popularity of blockchain trading contracts has attracted a considerable amount of frauds. According to the latest data from blockchain analysis group Chainalysis[1], the hidden encryption fraud in blockchain transactions caused 4.3 billion dollars in losses in 2019, which is only based on the reported encryption fraud.

Most frauds are related to Ponzi scheme. If Ponzi scheme is not taken into account, the losses caused by encryption fraud account for only 0.46% of all cryptocurrency activities. Ponzi scheme is named after the inventor. It is a classic fraud situation in the traditional financial investment field [15, 16]. In short, Ponzi scheme promises high returns to investors but uses the money of new investors to pay interest and short-term high returns to former investors [17]. After it is introduced into blockchain, a new form of smart Ponzi scheme based on blockchain has been created, and it brings huge losses to investors [15]. However, as the preferred platform for most blockchain fraud crimes, Ethereum still lacks effective supervision mechanism.

Plustoken is a cryptocurrency Ponzi scheme disguised as a high-yield investment scheme. The return mechanism of Plus-

[1]https://www.chainalysis.com/

token is illustrated in Fig. 1. In its propaganda, in addition to keeping users from losing money and making sure they can reap the benefits of digital currency appreciation, users can also get 8% to 30% of their principal as their monthly profits only if they agree to turn on smart dog for quantitative trading. To increase the sense of security, the investors can withdraw their coins at any time. However, the designers of the project cleverly set a rule, that investors will have to deduct 5% commission when they withdraw their coins in 28 days but only 1% after 28 days. Such a high fee difference makes greedy users usually withdraw their money later to save 4% of the fee. At the same time, the development of the lower line can also get a high commission. Users directly develop a lower line will be rewarded 100% commission, from the second generation to the 10th generation each reward 10% commission.

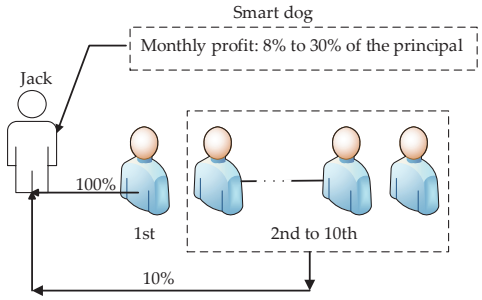

Figure 1: "Return Mechanism" of Plustoken.

Suppose that a user, say Jack, who is attracted by the high yield of digital money, is ready to invest in that. He buys some bitcoins and wants to store. Jack has some choices, he can store his money in an exchange for convenience, a wallet called Imtoken for safe, or Plustoken for higher returns. Putting money on an exchange means sharing control of the cryptocurrency with the exchange. Exchange control of cryptocurrency puts investors foreign exchange accounts at risk. Anything that affects the transaction (e.g., hacking, network problems, etc.) threatens the safety of investors funds. After giving up putting money on the exchange, Jack has to choose between two blockchain wallets, i.e., Plustoken and Imtoken. Considering that some people have already benefited from the Plustoken, Plustoken succeeds in getting his attention and becomes his first choice. Unlike other wallets, Jack only needs to turn on the smart dog, and he can get much more profits. Smart dog can automatically identify the trading volume and price difference of each trading platform, buy a digital currency at a low price on one trading platform, and sell it at a high price on another to earn money. Jack naturally deposits the money into Plustoken, falling victim to this Ponzi scheme. Perhaps he would also encourage his friends to invest in it. Whatever the risk appetite of investors, most investors will be tempted by the high returns promised by the Ponzi scheme when they know little about blockchain and Ponzi scheme [18, 19]. Therefore, in order to ensure the healthy development of blockchain technology, it is very urgent to detect the Ponzi scheme on it. In particular, to reduce the risk of fraud for investors, we need to detect Ponzi scheme in the early stage in real-time.

However, due to the anonymity of the blockchain, it is difficult to detect Ponzi schemes on the blockchain. First, we need enough validated samples. The source code reflects the logic of a smart contract, but source code is difficult to understand and needs to be processed and explained. This makes it difficult to collect samples. Second, there are a large number of smart contracts on Ethereum, which makes the manual checking method inefficient and time-consuming. Moreover, the source codes of most smart contracts on Ethereum may be hidden, only a small portion of the smart contracts on Ethereum has source codes. As illustrated in Fig. 2, the creator and the logic of the smart contract are hidden. We can not know how many Ponzi schemes there are on Ethereum, the features of Ponzi schemes in the source code, and how much of the influence they have. So we need an approach which is not based on source code to detect Ponzi schemes. This poses the following novel challenges for Ponzi schemes detection on Ethereum. (1) We need enough samples that have been verified whether they are Ponzi schemes to train the detection model. (2) Valid features that can be extracted without source code are also necessary, as the source code can be hidden. (3) We must establish a good model in performance to ensure the accuracy of detection. (4) Manual intervened checking has the property of detection delay. The number of smart contracts in Ethereum is increasing rapidly, making it difficult to automatically detect a Ponzi scheme in real-time when it occurs.

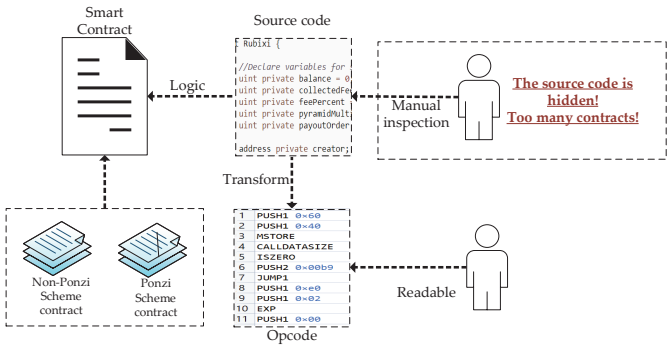

Figure 2: A demonstrating example of smart contracts Ponzi schemes detection.

Because there is a certain mutual conversion relationship among smart contract source code, contract bytecode and human readable operation code (opcode), in the case that there is no way to get all the smart contract source codes, so we can choose to analyze and predict the function of the code effectively by parsing bytecode. First, we can get the bytecodes. Second, we extract the opcodes from the data obtained by using the corresponding relationship of bytecode and opcode provided by the Ethereum yellow paper. Related research can be roughly divided into two categories, one is from the perspective of user accounts. These works check transaction account information to detect malicious account. The other is from the perspective of smart contracts, using opcode features of smart contracts to detect Ponzi schemes. Until the study [20] combined code features with account features, detected Ponzi scheme based on

machine learning and data mining. Currently, the best detection models are XGBoost and random forest. But as the technology evolves, there is still room for improvement in the detection accuracy of Ponzi scheme smart contracts. In addition, Ethereum has a large amount of contract information, the number of smart contracts on it accumulates quickly. In fact, the number of Ponzi scheme smart contracts is very small, compared to the total contracts; resulting in a class-imbalanced learning issue. It is still a challenging issue to effectively find the limited number of Ponzi scheme smart contracts under the big data-space Ethereum smart contracts. Data science approaches are needed for Ponzi schemes detection for Ethereum's smart contracts.

To deal with this above challenge, we propose a Ponzi Scheme detection method via oversampling-based Long Short-Term Memory for smart contracts (or PSD-OL) in this paper. To build an effective model for detection, we first collect enough smart contracts with labels, and then find valid features which can be extracted conveniently. The account features are combined with the contract code features to detect Ponzi scheme on Ethereum. The dataset is constructed based on the extracted account features and opcode frequency. To deal with the limitation of small number of Ponzi scheme smart contracts samples data, we over-sample the data to extract the valid feature data for detection. Long Short-Term Memory (LSTM) is then used to train the detection model for detecting Ponzi scheme smart contracts under the large-scale Ethereum's smart contracts. The contributions of this paper are summarized as follows.

- We present account features and opcode features for detecting Ponzi schemes on Ethereum.
- We propose PSD-OL, a data science-based approach by combining oversampling with LSTM to address the challenge of class-imbalanced large-scale smart contracts real-time detection issue.
- Experiments are carried out to compare the proposed method with four representative methods, and the results verify the effectiveness of the proposed method.

The remainder of the paper is organized as follows. We summarize the related works done on fraud detection and LSTM in Section 2. We give a brief introduction of the Ethereum platform and some key basic concepts in Section 3. A detailed description of the proposed PSD-OL approach is presented in Section 4. Experimental studies and the results analysis are shown in Section 5. Finally, we conclude the paper by laying out some important future directions in Section 6.

2. Related work

In this section, we review some relevant existing works that significantly inspired our PSD-OL approach including fraud detection and long short-term memory network.

2.1. Fraud Detection

Blockchain has sparked a global investment boom, but its technical barriers make it harder for investors to detect forms of fraud such as Ponzi schemes. According to statistics [15], from September 2, 2013 to September 9, 2014, the amount of money involved in bitcoin related fraud cases was as high as 7 million. In 2013, T. Moore et al. [21] identified that bitcoin-based frauds involved more than 4 million dollars which disguised as high yield investment plans. In 2017, M. Bartoletti et al. [15] analyzed the commonness of Ponzi scheme smart contracts from three aspects: descriptive information, source code and transaction record, and conducted an empirical study on the identification of Ponzi fraud on the blockchain for the first time in their report, the samples were distinguished by manual identification, and the related contracts were analyzed at different research levels in detail. In 2018, K. Toyoda et al. [22] used machine learning to demonstrate that over 80 percent of high yield accounts can be identified and classified based on the frequency of bitcoin transactions, which can guide the screening of malicious accounts which may have frauds. M. Vasek et al. [16] divided the scams in bitcoin into four types: Ponzi schemes, mining schemes, wallet schemes and fraudulent transactions. After that, W. Chen et al. [20] used machine learning to extract features from Eehereum smart contract opcodes based on analysis results by M. Bartoletti. Smart contract on Ethereum is distinguished from the normal contract by the classification model, and the method is proved to be more accurate than the manual comparison. Later, W. Chen et al. [23] used the method of random forest (RF) to detect Ponzi scheme smart contracts with higher accuracy than other machine learning methods.

2.2. Long Short-Term Memory

The Long-short Term Memory Network (LSTM) [24], which is a popular neural Network model, has successfully solved the defects of the original recurrent neural Network. It has been successfully applied in many fields, such as speech recognition [25, 26], picture description [27], fraud detection [28], etc. When the recurrent neural network (RNN) processes the long-distance data, it often has the problem of gradient explosion and gradient disappearance, which makes the RNN model can not remember the previous information. So the prediction accuracy is reduced.

To sum up, there are a lot of research achievements about blockchain, Ponzi scheme and LSTM. However, there has been little exploration of fraud on the blockchain. As for the current Ponzi scheme smart contracts detection methods, the performance of the better is XGBoost and Random Forest. Considering the related research has just begun, detection methods and accuracy still have room for further exploration. More importantly, the number of smart contracts to be detected on Ethereum is growing, but the number of Ponzi scheme smart contracts is very small, compared to the total contracts. This results in a class-imbalanced learning issue. We need an effective approach to process the data to obtain sufficient samples of the Ponzi scheme smart contracts for learning model training. Meanwhile, the increasing number of contracts also

challenges the ability of models to process large-scale data. In summarize, data science methodologies are essentially needed for Ponzi scheme smart contracts detection applications.

In this paper, we propose PSD-OL approach, which is a Ponzi scheme detection method based on oversampling-based Long Short-Term Memory. Account features and code features are extracted from contract call information and contract codes, and the two features are combined to detect Ponzi scheme smart contracts on Ethereum. At the same time, we consider the class-imbalanced training data, and solve it through the means of oversampling. The processed data is then built into the training data set. The training data set is used to train the LSTM model to detect Ponzi scheme smart contracts with time regulates. After training the model, the test set is used as input to get the recognition result to investigate the effectiveness of the model. The result is compared with the contract label value, and the detection accuracy of PSD-OL method is verified after that.

3. Preliminaries

This section briefly introduces some basic concepts involved in the research, including Ethereum, smart contract, Ponzi scheme, and imbalanced data classification. First, we introduce Ethereum and smart contract. Then we analyze the development and common features of smart Ponzi scheme to introduce the harm of Ponzi scheme. Finally the class-imbalanced data processing approaches are introduced, which significantly influenced our PSD-OL approach.

3.1. Ethereum and smart contracts

Ethereum is a distributed platform based on the open source blockchain technique [29, 30]. At the end of 2013, B. Vitalik introduced smart contract to blockchain in his Ethereum white paper, expanding the use of blockchain outside the monetary realm. The second generation of blockchain was born. As the representative of the second generation blockchain platform, Ethereum provides a very complete framework for the development of smart contracts [31]. In addition, there are enough API interfaces on it to make sure that users can develop all kinds of blockchain applications [32, 33, 34]. Ethereum is now the mainstream platform of smart contract development and operation. It allows users to run programs in a trusted environment with the blockchain. Ethereum has built an Ethereum Virtual Machine (EVM), which allows code to be validated and executed on the blockchain, ensuring that code runs the same way on the machine of everyone. The code is included in the smart contracts.

N. Szabo defined the smart contract as "a set of commitments defined in digital form, including agreements in which the parties to a contract can execute those commitments" [35]. He argued that complex transactions can be accomplished by using computational codes instead of machines. But in the absence of trusted environments, digital systems and technology, smart contracts were not widely used at that time. Until the rise of blockchain technology makes the further development of smart contracts. The blockchain has the advantages of decentralization and immutability–making the smart contract solve the trust problem perfectly [36]. Smart contract is a computer protocol between distrustful participants that is automatically enforced on the blockchain when the default conditions are met. The execution of a smart contract can not be terminated and is not dependent on any trusted authority [37].

The smart contract can be applied in various fields [10, 38, 39]. This principle has also been applied to the financial sector. French insurance company AXA is using the public Ethereum blockchain to offer automated flight delay compensation to air travelers. They use the public Ethereum blockchain to record insurance product purchases and trigger automatic payments by using smart contracts on the blockchain. Ethereum smart contracts also connect to a global air traffic database to constantly monitor the flight data. When a flight is delayed by more than two hours, the compensation mechanism will be automatically implemented and sent directly to the policyholders credit card account, which is independent of AXAs decision. After the emergence of blockchain, the developers conceive of embedding smart contracts into blockchain to avoid malicious tampering of contract conditions [40]. The smart contract is the traditional contract after digitization, which runs on the blockchain database. To summarize, smart contract specifies a clause in its program code, including its trigger condition, and execute it as soon as the condition is met.

Smart contracts are usually written in a high-level language. Solidity is a contract-oriented high-level language, which is important on the Ethereum Virtual Machine (EVM) platform [41]. To deploy Ethereum smart contract, the source code of smart contract needs to be converted into EVM bytecode through virtual machine. An EVM bytecode consists of a series of bytes, each corresponds to an operation. There is a mnemonic form for each operation, the mnemonic form of an EVM bytecode is called opcode. The Ethereum yellow paper contains a complete list of bytecode and opcode. The dis-assembler is used to get opcode of contracts from bytecode, opcodes consist of a series of opcodes and operands. The conversion relationship among source code, bytecode, and opcode for a smart contract (ID:0xe82719202e5965cf5d9b6673b7503a3b92de20be) is shown in Fig 3.

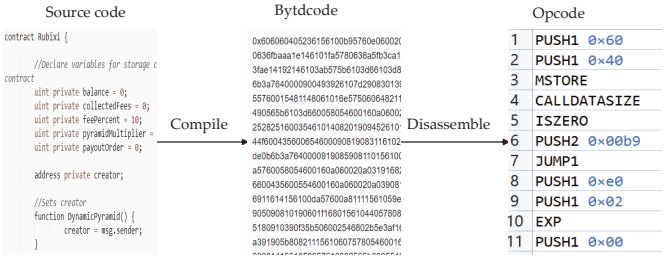

Figure 3: Smart contract code transformation.

3.2. Ponzi Scheme

Ponzi scheme is a typical form of deception, named after the originator of a large-scale scheme nearly 100 years ago. Ponz-

i scheme is investment activity that involves fraud. Instead of generating profits through any legitimate business or financial transaction, the Ponzi scheme distributes a portion of the money invested by subsequent investors directly to the former investors, to create the illusion of continued returns and attract more follow-up investors to ensure continued investment [42]. The essence of the Ponzi scheme is to give back to the previous round of investors the amount invested by subsequent investors as a return on their investment activities. In the process, many participants, especially those who involved later in a Ponzi scheme, typically invest far more than they gain.

Smart contract seems to be an attractive tool for Ponzi schemes as it is automatically enforced and cannot be terminated on blockchain. More importantly, the promoters stay anonymous [17]. Because the writing of a smart contract involves many aspects, such as privacy and law, it is very difficult to detect the whole process of smart contract trading. Any new technology is vulnerable to fraud. Compared to the traditional software, smart contracts in the financial security is more difficult to control as the promoters stay anonymous.

Nowadays, many Ponzi schemes disguise themselves under the veil of smart contracts. We refer these Ponzi schemes as the corresponding smart Ponzi schemes. Since participants' confidence in getting consistent return is the key factor in successful operation of a Ponzi scheme, so the Ponzi scheme has to attract a steady stream of participants. In summarize, smart Ponzi scheme has the following features.

- Incredible returns. It is where most people lose their ability to think calmly.
- Membership class system. Advanced investors can get follow-up investors part of the income according to the entry sooner or later class.
- False project description, such as completely decentralized, 100 percent safe transparent and other exaggerated descriptions.

3.3. Imbalanced Data Classification

In the Ponzi scheme detection system, the number of legal cases (positive samples) will be far greater than the number of fraud cases (negative samples). In a classifier model, the classifier will have a high classification accuracy as long as all the samples are recognized as legal cases. In contrast, for the limited number of fraud cases in the whole smart contrasts, the high accuracy of the classifier in the Ponzi scheme detection system does not make sense [43]. Ponzi scheme detection is essentially an imbalanced classification problem. In practice, most anomaly detection will encounter imbalanced training data set issue due to the number of abnormal data will be far less than that of normal data. At present, the related methods of imbalanced classification are mainly studied from data level and algorithm level. The data level processing is to re-sampling the data, including under-sampling and oversampling [44]. The main idea is to reduce the impact of data imbalance on classifier by adjusting the imbalanced proportion of data.

In particular, the oversampling method improves the classification performance by adding the number of samples of minority class [45]. Simple random oversampling method is to make a simple copy on samples of minority class, but it does not add extra information of samples of minority class. It is easy to lead to overfitting. In contrast, the Synthetic Minority Oversampling Technique (SMOTE) is an improved method based on the random oversampling method [46], which avoids the above problem. In this paper, SMOTE algorithm is used to process the data to deal with the impact of data imbalance issue.

4. PSD-OL Approach

In this section, we present PSD-OL approach, including the overview of PSD-OL approach, the data acquisition, the feature extraction, the data preparation based on oversampling and the LSTM model training.

4.1. Overview of the Approach

As illustrated in Fig 4, we first give the overall framework of PSD-OL approach. The opcode and account features of smart contract are extracted as the relevant features. A dataset is built based on the feature data. The dataset is preprocessed before it is used. The data are oversampled to extract the valid feature data, and then normalized using MinMaxScaler function. After that, the dataset is divided into training set and test set. The training set accounts for 70% and the test set for 30%, respectively. The training set is then used to train the Long Short-Term Memory (LSTM) model for Ponzi scheme smart contract detection. We take 83 features of the smart contracts to be tested as the input of the LSTM model. A dense layer is added to convert the dimensions of output h_t of the LSTM model. The sigmoid activation function is used to calculate the predicted value of a smart contract, which is used to judge whether the smart contract is a Ponzi scheme contract. After the training of the model, the test set is used to validate the effectiveness of the model. The effectiveness of Ponzi scheme detection of LSTM model can be obtained by measuring their performance metrics such as precision and recall. If the model is validated by the test set, it can then be used for Ponzi schemes detection for smart constraints.

4.2. Data Acquisition

In the consideration of smart contract account features, this paper analyzes the transaction records between the contract caller and the contract. We extract the account features and apply them to model calculation. When we consider the smart contract code features, smart contract source codes are necessary, but they may be hidden. In fact, implementing smart contracts on Ethereum requires only bytecode, so we choose to parse bytecode for efficient analysis. There is a certain mutual conversion relationship among contract source code, contract bytecode and human-readable opcodes. If we want to obtain the code features of the contract, we must first obtain the bytecode of the smart contract, and then develop a translator tool to

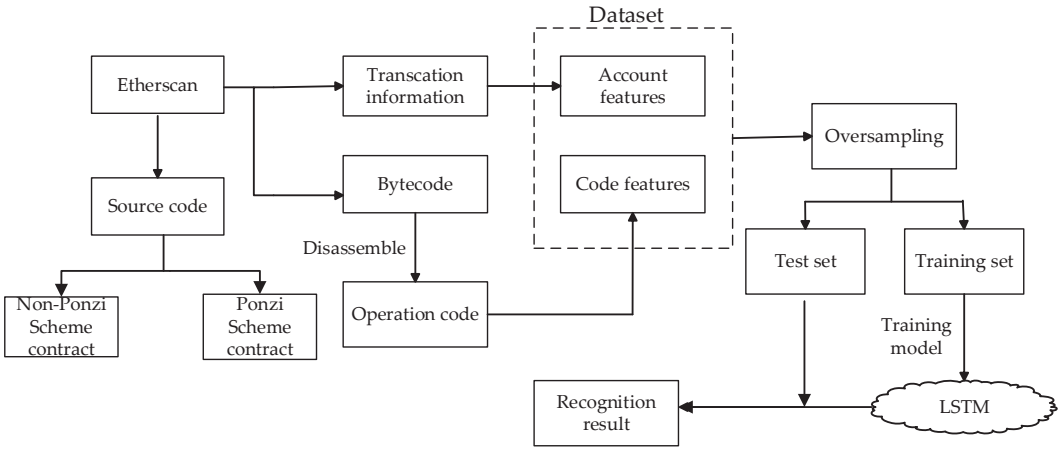

Figure 4: The framework of PSD-OL approach.

extract the opcode from the acquired bytecodes data based on Ethereum yellow paper.

As shown in Table 1, when a transaction occurs between contract and participant, a call record is generated. Each transaction block number, participant, direction, Ether amount, time stamp can be extracted from the contract call record.

The block number (BNum) in the Ethereum record is used to mark the position of the contract information on the Blockchain. TimeStamp (Time) is used to mark the time when the contract occurred. Transaction Hash (TxHash) is the transaction number that can be used to search the successful transaction information on the blockchain website. By FromIsContract and ToIsContract, 1 represents the contract, 0 represents the investor. A transaction from an investor to the contract is called an investment transaction. A transaction from the contract to an investor is a payment transaction. Contract amount value is the amount involved in a contract expressed in scientific notation. Only when the value of Error is None, the transaction will run smoothly, while the rest indicates that the transaction did not succeed. The most common error is "Out Of Gas".

4.3. Feature Extraction

To establish a detection model, the feature of the acquired data is extracted. We will extract code features and account features, respectively, to guarantee the effectiveness of the detection model.

4.3.1. Code Features

The contract opcode contains the logic part of the contract source code to a large extent. It can reflect the contract logic from the EVM point of view. Almost all the triggers in the contract are represented in the opcode. An Ethereum smart contract can only be enforced if the default conditions are met. Thus, Ponzi scheme smart contract often embed mechanisms which allow the frauds to go smoothly into the source code, and the underlying problems of the contracts are also characterized in the opcodes. To effectively distinguish the Ponzi scheme smart contract and the normal smart contract in real-time, this paper analyzes the types and frequency of the contract opcodes, extracts the features of the contract codes. We statistics the frequency of the appearance of different opcodes in the smart contract as the code features. At last, we collect 76 code features for each smart contract. As shown in Fig. 5, we analyze the difference of opcodes between Ponzi scheme smart contracts and normal smart contracts. In particular, the normal contract ID is 0x1caf0d0384aca96b9d4c43afa5400c1e08c22d4 and the Ponzi scheme smart contract ID is 0x3f4dd010f9a9b9d95f1f53837d7d7e9f3fac8.

As can be seen in Fig. 5, there is a great difference between the Ponzi scheme smart contract opcodes and the normal smart contract opcodes without considering the data processing opcodes that occur most frequently, such as PUSH and DUP. The main difference is that Ponzi scheme smart contracts contain more treaty functions (e.g., JUMP, JUMPI, JUMPDEST, etc.) and call functions (e.g., CALL, CALLVALUE, CALLDATALOAD, etc.) than normal contracts. The above analysis indicates that opcode features may be feasible in detecting Ponzi scheme smart contracts.

Figure 5: The Opcode word cloud of Ponzi (right) and non-Ponzi scheme (left) contracts.

4.3.2. Account Features

Comparing with the normal smart contracts, there are some differences in the trading behavior of the Ponzi scheme smart contracts on Ethereum. In particular, there are some obvious features in the process of the circulation of Ether (the corresponding coin of Ethereum) in its contract transaction. According to the results of manual inspection, we summarize the following basic features.

Table 1: Contract invocation information

BNum	Time	Txhash	From	To	Fisc	Tisc	Value	Error
50107	$144\cdots$	$0x\cdots$	$0x109\cdots$	$0x881\cdots$	1	0	1E+18	NONE
50249	$144\cdots$	$0x\cdots$	$0x109\cdots$	$0x109\cdots$	0	1	1E+20	OUT OF GAS

- The Rate of return for a small number of contract participants is very high, and basically all returns are concentrated in one or two participants. Most of the participants have high returns in Ponzi scheme smart contracts are the founders themselves.

- The balance of many Ponzi scheme smart contracts has been kept very low, generally taking the operation of the rapid distribution of the investment obtained.

The contract and the transactions between its participants are shown in Fig. 6. Ponzi scheme smart contracts have their particularity in the direction of the flow of Ether, the amount of transactions, the balance of contracts, etc.. By visualizing the changes in the balance of contract participants in the Rubixi, the prominent features of a Ponzi scheme smart contract can be obtained. In particular, the x axis is the time stamp of the change in the balance of Ethereum smart contract participants, shows the time stamp of the call record (i.e., the number of seconds from the start of the contract to the call); the participant serial number is on the y axis. The value on the y axis represents the time in which the contract participants first entered into the contract. The radius of the dot is the amount of the balance. A series of dots at each level represent the change in a participant's balance. As illustrated in Fig. 6, early investors will execute the payment transactions (the participant receives Ether from the contract) and later investors will execute the investment transactions (the participant sends Ether to the contract) during the whole transaction process of the smart contract.

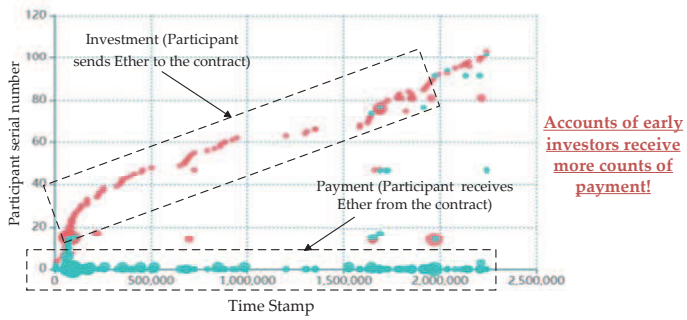

Figure 6: The Ether flow graph of Rubixi.

Fig. 6 presents the transactions involved by each participant along the timeline, as well as the changes in the balance of the participants. Each time the corresponding amount of Ether is represented by the two-point radius and color change of the participants. Contracts and transactions between participants are placed on a horizontal line in the order of the transaction time stamps. Participants are ranked according to the time stamp of their first transaction with the contract. Therefore, a participant with the serial number 1 is usually the creator of the contract. Usually, in normal economic activity, an investment transaction should be followed by a payment transaction, but this is not the case in contracts involving Ponzi scheme. In Rubixi, a Ponzi scheme smart contract, changes in the balance of participants in the contract indicate that early participants received more payments, and that the bulk of the payments were concentrated in the hands of one or two participants. More than 200 transactions were selected from the contracts, while payment transactions were largely focused on interacting with the top 10 participants. Most of the payments in Ponzi scheme contract calls are concentrated on the previous zigzags and on the investors who invested earlier, while the payments in normal contracts are more random.

To sum up, Ponzi scheme smart contracts have the following salient features in contract invocations: (1) Relatively low balance levels. (2) Payment transactions tend to focus on earlier participants. (3) Some participants have more payment transactions than investment transactions.

Based on these observations and features, we extracted the following seven key features from the contract account.

- Balance rate (R_Bal): balance as a percentage of total investments.

- Investment number (NI): the number of investments.

- Payment munber (NP): the number of payments.

- Payment time (R_PT): suppose there are n participants in a Ponzi scheme, i represents the serial number of the investor, e_i represents the earning obtained by the ith participant, and E represents the total earning obtained by all investors. Then

$$R\_PT = \frac{\sum_{i=1}^{n} (e_i \times \frac{i}{n})}{E} \tag{1}$$

  It is used primarily to indicate the order in which the investor receives a high return. The earlier the investor receives a return, the closer the value is to 0. The later the investor receives a return, the closer the value is to 1. For Ponzi scheme smart contracts, this value is usually very small. It is because investors who make big gains are often concentrated in the early stage to attract follow-up investors.

- Difference index (D_ind): this index is used to measure the difference in the number of payments and investments

made by participants. Suppose that the smart contract has p participants, m_i and n_i are used to represent the number of investment and payment transactions of the ith participant, respectively. In order to calculate the difference index, we first calculate V_i $(= n_i - m_i)$ to get the difference in the number of investment and payment transactions of participants. Then,

$$D\_ind = f(x) = \begin{cases} 0, & p < 3 \\ s, & p \geq 3 \end{cases} \tag{2}$$

where s is the skewness of $\{V_1, V_2, ...V_p\}$. For Ponzi scheme smart contracts, the value of D_ind is generally negative because many participants have investments but receive little.

- *R_MAXtimes* (*R_MT*): rate of the maximum times of payments to participants.
- *R_MAXnum* (*R_MN*): rate of the maximum numbers of payments to participants.

Tables 2 and 3 show the average, median, and standard deviation statistics of the extracted features, compare the statistics of Ponzi scheme smart contracts with those of normal contracts. The difference of account features between the two types of contracts is obtained, and we also analyze the value of the extracted features in the detection of Ponzi scheme smart contracts.

As can be seen from Tables 2 and 3, there is a significant statistical difference in the seven account features between the Ponzi scheme smart contracts and normal smart contracts, particularly in terms of the first five features. For example, the average and Median of *R_Bal* of normal contract is actually higher than that of Ponzi scheme smart contract. This shows that the balance of the Ponzi scheme smart contracts has been maintained at a relatively low level, a lower standard deviation also indicates that there are more contracts with lower balances in Ponzi scheme smart contracts.

In addition, when we consider *R_MT* and *R_MN*, although both remain high in mean and median, the higher standard deviation of the normal contract indicates that this phenomenon is more common in Ponzi scheme smart contracts. Since the Ponzi scheme smart contract generally has a specific largest beneficiary, such as the founder, *R_MT* and *R_MN* will maintain a relatively high level.

4.4. Data Preparation based on Oversampling

We construct a dataset D based on extracted account features and contract code features. Suppose that there are N smart contracts in the dataset $\{(\mathrm{x}_i, y_i)|i = 1, 2, ..., N\}$. x_i is the extracted features corresponding to the ith smart contract, and y_i is the classification label. When $y_i = 0$, the contract is a Non-Ponzi scheme smart contract; when $y_i = 1$, the contract is a Ponzi scheme smart contract. The normal sample set of majority class is D_n and the fraud sample set of minority class is D_f.

To solve the problem of sample imbalance problem in Ponzi scheme detection, we utilize SMOTE algorithm to oversample the minority class fraud sample set D_f. Specifically, we select a random sample x_i from D_f. The Euclidean distance is used to calculate the distance of x_i to all samples in D_f to get five samples within the same class nearest to it. Then we choose a sample x_a from the five samples, and generate a random number between 0 and 1. Finally, synthesize a new sample according to the following formula.

$$x_{new} = x_i + rand(0, 1) \times (x_a - x_i) \tag{3}$$

Repeat the above steps until the sample is balanced to get a new sample set X.

In addition, the diversity of feature selection results in a large difference in the size and distribution range of each selected feature. Model parameters may be dominated by features with large or small distribution range. To solve this problem, features need to be normalized and transformed into a rating, so as to eliminate each value's dimension and unify the tendency. We utilize the min-max normalization to map the feature values into the range of [0, 1] [47, 48]. The zoom method is as follows.

$$q_{ik} = \frac{f_{ik} - f_{i\min}}{f_{i\max} - f_{i\min}} \tag{4}$$

where f_{ik} is the feature value of the kth smart contract among the selected ith feature, $\mathrm{f}_{i\min}$ and $\mathrm{f}_{i\max}$ represent the minimum and maximum values of the selected feature, respectively.

4.5. LSTM

To deal with the big data computing challenge of large-scale Ponzi scheme smart contracts detection application and guarantee the effectiveness of the detection results, LSTM is used in this paper to learn from the samples' features with the time evolution. The trained LSTM model will be used for detecting Ponzi scheme for smart contracts. Specifically, LSTM is a special kind of Recurrent Neural Network (RNN), which is mainly used to solve the problem of gradient vanishing and gradient explosion in long sequence training. Compared with the traditional recurrent neural network (or RNN), the LSTM model is still based on current input x_t and output of the previous time stamp h_{t-1} to calculate current output h_t, but it is more carefully designed for the internal structure. Long term memory is maintained through specialized gate units, i.e., the forget gate F_t, the input gate I_t, the output gate O_t, and memory unit c_t. The forget gate controls how much information in the previous memory unit is forgotten, and the input gate controls how much the new state of the current computation is updated into the memory unit, the extent to which output gate controls the current output depends on the current memory unit. The structure of the LSTM is shown in figure 7. Where [] means that the two vectors are connected, $\times$ represents the corresponding multiplication of each element in the matrix, σ is a sigmoid function. In the formulas below, c_t is a cell state, w is a weight coefficient, b is a bias, h_t is the output of the network. LSTM acts like a conveyor belt, carrying information from one neuron to the next.

In this paper, we take the processed smart contract code features and account features as input to the LSTM model. In the LSTM block, given the output of the previous time stamp is

Table 2: Ponzi scheme smart contract account features statistics

	R_Bal	NI	NP	*R_PT*	*D_ind*	*R_MT*	*R_MN*
Mean	0.19	66.63	37.78	0.46	-1.20	0.56	0.24
Medium	0.00	12.00	8.00	0.43	0.00	0.58	0.11
Rd	0.35	287.17	90.52	0.31	10.12	0.34	0.31

Table 3: Normal smart contract account features statistics

	R_Bal	NI	NP	*R_PT*	*D_ind*	*R_MT*	*R_MN*
Mean	0.22	3843.85	2630.85	0.47	-3198.11	0.65	0.51
Medium	0.00	6.00	3.00	0.48	0.00	0.90	0.50
Rd	0.38	55269.91	48796.21	0.49	105064.84	0.40	0.45

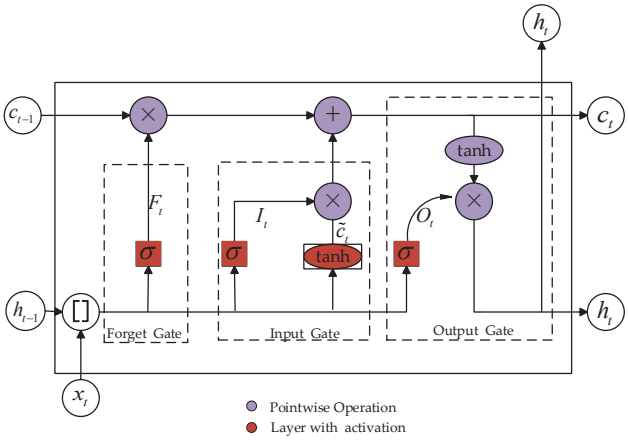

Figure 7: Structure of LSTM.

h_{t-1}, the reconstructed x_t is taken as the input of the current time stamp, the state of the current time stamp is computed by

$$\tilde{c_t} = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{5}$$

The LSTM uses two gates to control the contents of unit state c_t, one is forget gate, which determines how much of the unit state of the last moment, c_{t-1} is retained to the current moment c_t. The other is input gate, it determines how much of the input to the current time network x_t is saved to the unit state c_t. Then the input gate and the forget gate at the current time stamp are computed by

$$I_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{6}$$

and

$$F_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{7}$$

where $\sigma(\cdot)$ is the activation function that controls the flow of information. The state of the current time stamp is updated by

$$c_t = I_t \times \tilde{c_t} + F_t \times c_{t-1} \tag{8}$$

In this way, we combine the current memory of the LSTM with the long-term memory c_{t-1} to form a new unit state c_t. Because of the control of the forget gate, it can save information from a long time ago, and because of the control of the input gate, it can avoid the current irrelevant content into memory. Next, we look at output gate, which control the effect of long-term memory on current output. The output gate is governed by

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{9}$$

The final output of the LSTM is determined by both the output gate and the unit state c_t. Finally, the output of the current time stamp is computed as follows.

$$h_t = O_t \times \tanh(c_t) \tag{10}$$

After that, a dense layer is added to convert the dimensions of output h_t of the LSTM model. The activation function sigmoid is used to calculate the predicted value of smart contract, which is used to judge whether the smart contract is a Ponzi scheme contract. The output is compared with the label y_i of the smart contract, and the performance of the model in the training set is measured by the loss function *binary_crossentropy*. After learning and judging all the data in the training set, the loss value is fed back to the Adam Optimizer to optimize and update the weights in each hidden layer for a new round of training. This makes the model have more accurate classification ability to the training set. Finally, the smart contract is detected on the test set to obtain the detection accuracy of the model for Ponzi scheme smart contracts.

5. Experiments

In this section, a set of experiments are conducted to evaluate PSD-OL approach. To assess the effectiveness of the proposed approach for Ponzi scheme detection, experiments were performed on a validated dataset. We first present the experimental settings and evaluation metrics. Then, we introduce the experiments done using the proposed method for Ponzi scheme smart contracts detection and compared the performance with the representative methods.

5.1. Dataset

In terms of dataset selection, this paper selects a validated sample set of 3019 contracts from previous studies [20], which can be obtained from the open dataset XBlock. The dataset

includes 168 Ponzi scheme smart contracts and 2851 normal smart contracts.

According to the address of the contract, this paper crawls the bytecode and transcation information of the corresponding contract on etherscan.io. Data of contract information and contract call information on the Ethereum blockchain are filtered out from the data blocks of the corresponding time period. In addition, to obtain the code features, we disassemble the bytecode into opcodes, and then calculate the ratio of each opcode to the total number of opcodes. Meanwhile, by analyzing the transaction records between the contract caller and the contract, we extract the account features which represent the characteristics of smart Ponzi schemes Ether flow. We combine account features and code features to detect Ponzi fraud in smart contracts in this paper, which means that we select 83 features (7 account features and 76 code features) for each smart contract to build the data set.

5.2. Evaluation metrics

To evaluate the performance of the model accurately, and to facilitate performance comparisons with other methods for detecting Ponzi scheme smart contracts, we propose three metrics, i.e., precision, recall, and F-score. The specific definitions of these 3 metrics are presented as follows.

$$\text{Precision} = \frac{true\ positive}{true\ positive + false\ positive} \tag{11}$$

$$\text{Recall} = \frac{true\ positive}{true\ positive + false\ negative} \tag{12}$$

$$F - score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

5.3. Approaches under comparison

The existing Ponzi scheme detection works are mainly based on traditional machine learning methods such as eXtreme Gradient Boosting (XGBoost), random forests(RF), etc.. Logistic Regression(LR) and Wasserstein distance(WD) are also representative solutions to the problem of data classification. Therefore, in order to evaluate the effectiveness of the proposed approach, we not only used the PSD-OL method to detect Ponzi scheme smart contracts, but also implemented the following 4 different approaches for comparison purpose.

- **XGBoost** method is one of the most popular machine learning algorithms, which has proved to be an excellent method to solve this type of classification problem in the past research [20, 49]. We regard the fraud detection problem as the two-classification problem between normal samples and fraud samples, and use XGBoost method to detect Ponzi scheme smart contracts.

- **Random Forest (RF)** is a typical classification algorithm proposed by Leo Breiman in 2001, which uses bootstrap resampling technique to generate a new training decision tree by repeatedly sampling samples from the original training sample set [50, 23]. Then according to the above steps, decision trees are generated to form a random forest. The classification results of the new data depend on the score formed by the voting of the classification tree. In essence, it is an improvement decision tree algorithm, which merges several decision trees together, and the establishment of each tree depends on the independent samples.

- **Wasserstein distance (WD)** is commonly known as earth mover's distance (EMD) (and also referred to as the "transportation metric"), which is used to indicate the degree of similarity between two distributions [51, 52]. In this paper, we calculate the distance between the sample to be detected and the Ponzi scheme sample, and determine whether it is a Ponzi scheme by setting a threshold.

- **Logistic Regression (LR)** can be used as a classification model in machine learning, and it is a typical classification algorithm [53]. Due to the simplicity and efficiency of the algorithm, it is widely used in practice. We can think of the classification task as estimating the probability of the occurrence of an event as p, and use the probability of the occurrence of an event to achieve the classification. It fits the data into a logistic function and compares its value to a set threshold to determine its category.

5.4. Impact of parameter Epochs

We investigated the impact of the parameter to the detection performance. It is difficult to select the optimal value for each combination of variables due to the large number and the wide range of variables. On the premise that other variables are determined, change the value of some variables in order to find a relatively optimal parameter combination. By LSTM, one epoch means that every sample in the training set is trained one time. It is not enough to transfer the data set once in the neural network, and we need to transfer the complete data set several times in the same neural network. As the epoch increases, the model changes from under fitting to over fitting, so we need to set a proper epoch.

In this experiment, *binary_crossentropy* is chosen as the measure of the loss function, we set *Batch_size* = 128, *hidden_layers* =32. Then we set the epoch's initial value to a large number as 500 and note the actual epoch value when the training process convergences. We observe the accuracy of the impact of actual epoch value changes. The training loss and testing loss as the actual epoch value changes are shown in Fig. 8. The training accuracy and testing accuracy are shown in Fig. 9.

From the two figures above, we can see that, initially, as the actual epoch grows, the training accuracy and the testing accuracy increase. By the time the actual epoch reaches 400, the accuracy rate starts to stabilize. The training losses keep getting smaller, but the testing losses stay stable after the actual epoch's value reaches 400. After that, we end up setting the epoch's initial value to 400.

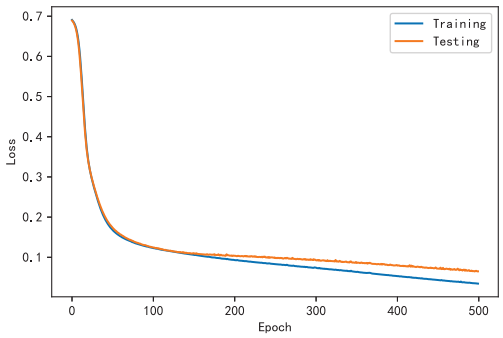

Figure 8: Training Loss and Testing Loss.

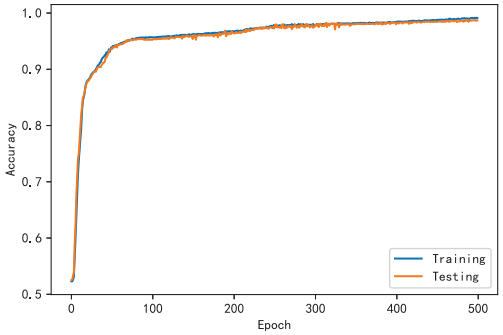

Figure 9: Training Accuracy and Testing Accuracy.

5.5. Impact of parameter Batch_size

Batch_size is related to the gradient descent direction of the LSTM model in each learning process. A small *Batch_size* will make the model difficult to converge, the learning time will be too long. On the other hand, a too large *Batch_size* requires high memory, and will make the number of rounds of convergence relatively longer. Therefore, the selection of batch-size needs to make a consideration between training time and the number of training rounds needed for convergence.

In this experiment, we set *epoch* = 400, *hidden_layers* =32, the number of *Batch_size* is gradually increased within {1, 32, 64, 128, 256, 512}. Figs. 10 and 11 show the time it takes for the model to complete its training and the training accuracy.

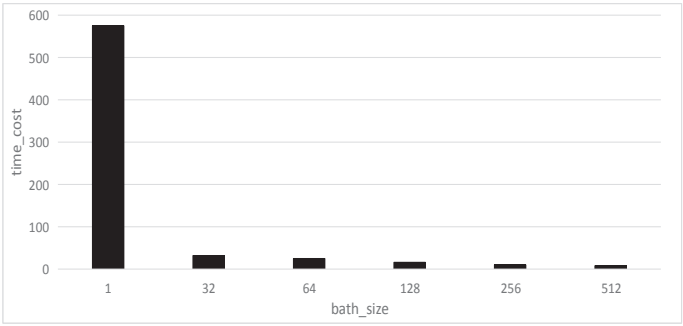

Figure 10: Training time on different number of *batch_size*.

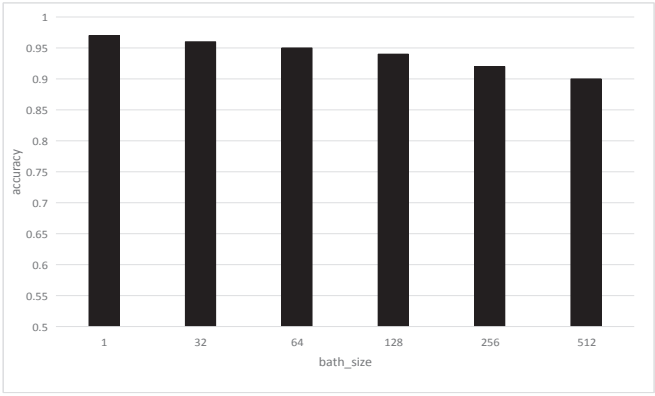

Figure 11: Training accuracy on different number of *batch_size*.

We can see that, as *batch_size* increases, accuracy and the computation overhead declines. When the number of *Batch_size* is over 32, the change of the time it costs is not very obvious. But when the number is over 128, the accuracy declines obviously. By weighing the time needed to train the model and the training accuracy, we ended up setting the bitch-size to 128.

5.6. Impact of the scale of training set

We also consider the effect of the size of the test data set on model training. If The data set partition is good, it may speedup the model training time, or it will greatly affect the deployment of the model application. In this experiment, we change the ratio of the training set and the test set in this paper to study the effect of data set size on the model, while *epoch* = 400, *hidden_layers* =32, *batch_size*=128. Fig. 12 presents the impact of the scale of training set.

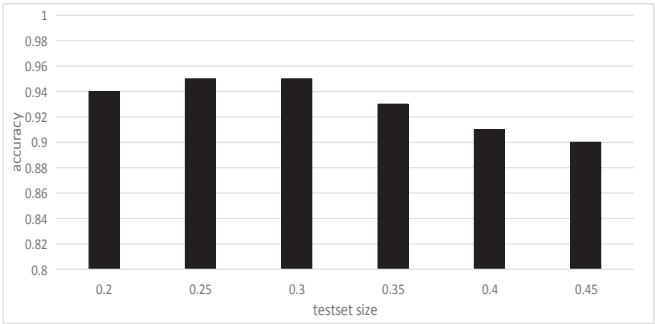

Figure 12: Accuracy on different ratio of test set.

We can see that, as the ratio of the test set increases, accuracy of the model declines. When the ratio of test set reaches 0.25 and 0.3, the accuracy is highest. As the test set size gets larger, the training set size gets smaller. The small size of training set will directly affect the classification performance, because almost all classification experiments prove that a large enough training set can bring higher classification accuracy.

5.7. Impact of the number of neurons

When the number of neurons is too small, a small number of neurons need to represent a large number of features, which

will lead to under fitting. In contrast, using too many neurons can also cause performance declines. When the neural network has too many nodes, the information contained in the training set is not enough to train all the neurons in the hidden layer, so it will lead to over fitting. Even if the training data contains enough information, too many neurons in the hidden layer will increase the training time, so it is difficult to achieve the desired effect. Obviously, the selection of a suitable number of hidden layer neurons is very important.

To investigate the effect of the number of neurons in each layer on the efficiency of PSD-OL, in this experiment, we set $epoch$ = 400, $hidden\_layers$ =32, the number of neurons is gradually increased within $\{8, 16, 32, 64, 128\}$. Fig. 13 shows the training accuracy under different number of neurons. When the number of neurons in each layer is relatively small, the training accuracy increases with the increase of the number of neurons, and the increase slows down when the number of neurons reaches 32. So we ended up with 32 neurons per layer.

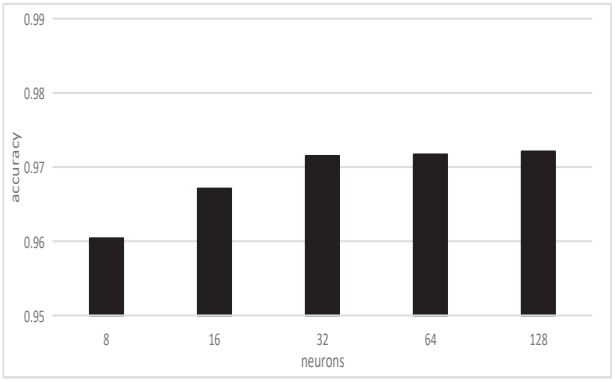

Figure 13: Training accuracy affected by different number of neurons.

5.8. Performance Comparison

This experiment aims at comparing the effectiveness of different approaches. In the PSD-OL approach proposed in this paper, the data set is divided into training set and test set in the ratio of 7:3. In addition, we set LSTM model parameter epoch= 400, $Batch\_size$ = 128, and the number of neurons=32. We trained the models of different approaches by the training set. After the training, the recognition accuracy of different approaches for smart contract detection is tested on the test set. The effectiveness of the above methods for detecting Ponzi scheme smart contracts is measured by using precision, recall, and F-score. The experimental results are shown in Table 4.

As can be seen from Table 4, first, compared with other classification models, the proposed method (PSD-OL) significantly improves the performance of all indicators. In particular, the F-score increased to 0.96, which indicates that PSD-OL is an ideal method for smart Ponzi scheme detection. On the other hand, a higher recall rate will greatly improve efficiency, because it will successfully detect many real Ponzi scheme smart contracts. After the expansion of the dataset, facing the challenge of big-data space Ponzi scheme detection, the neural network we introduced does have better performance.

Table 4: Performance comparison of different methods.

Algorithm	Precision	Recall	F-score
WD	0.33	0.21	0.25
LR	0.83	0.82	0.82
RF	0.95	0.64	0.76
XGBoost	0.92	0.70	0.79
PSD-OL	0.97	0.96	0.96

The model uses SMOTE algorithm to oversample the data. In order to reflect the importance of oversampling technology, we have also drawn the confusion matrix when the technology is used or not, respectively. By analyzing the confusion matrix in Figs. 14 and 15, the accuracy of the model's classification of minority samples is improved after sampling. When the data distribution is very unbalanced, the model will identify all the smart contracts to be detected as Non Ponzi scheme smart contract, forming invalid recognition. As can be seen from the results, oversampling technology solves this problem and the model can accurately detect Ponzi scheme. It is reason why the efficiency of model detection can be greatly improved.

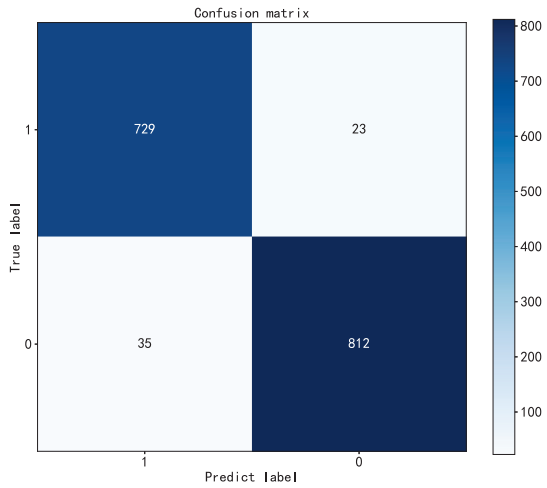

Figure 14: Model confusion matrix with oversampling.

6. Conclusion and Future Work

With the development of blockchain technology, all kinds of trading activities based on blockchain become more and more frequent. Ponzi scheme smart contracts, which imply fraud, is a great hidden risk in the trading process of blockchain-based Ethereum platform. In this study, we propose PSD-OL, which combines account features with contract code features of smart contracts to detect Ponzi Scheme on Ethereum. In PSD-OL, we integrate the SOMTE algorithm and LSTM to construct the detection model and deal with the big data analytics research challenge. The experimental results conducted on well-known open dataset XBlock show that PSD-OL has high accuracy, improves precision, recall and F-score. It has certain application value in the detection of Ponzi scheme smart contracts on Ethereum.

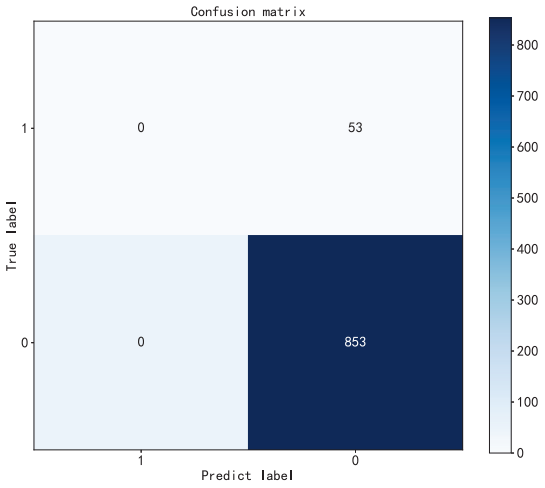

Figure 15: Model confusion matrix without oversampling.

With the development of detection technique, creators of Ponzi scheme may also constantly change their fraud methods. In the future, we plan to integrate concept drift to investigate the change of fraud features. Based on the sampling technique, we will extract the drifting fraud information for model training from the stream data of up-to-date samples which have different features. Moreover, we will try more deep learning methods to see if there are any benefits for Ponzi scheme smart contracts detection.

Acknowledgments

This work was partially supported by the Humanity and Social Science Youth Fund of Ministry of Education of China (No.18YJCZH170), Six talent peaks project in Jiangsu Province in 2019 (No. RJFW-029), Innovative training program for College Students in Jiangsu Province (No. 202110298077Y). L. Wang is the corresponding author.

References

[1] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system (2008).
[2] E. Portmann, Rezension "blockchain: Blueprint for a new economy", H-MD Prax. Wirtsch. 55 (6) (2018) 1362–1364.
[3] A. B. Gómez, J. L. López-Sánchez, M. Acevedo-Aguilar, Blockverse: A cloud blockchain-based platform for tracking in affiliate systems, Int. J. Interact. Multim. Artif. Intell. 6 (3) (2020) 24–31.
[4] K. Huang, X. Zhang, Y. Mu, F. Rezaeibagha, X. Du, Scalable and redactable blockchain with update and anonymity, Inf. Sci. 546 (2021) 25–41.
[5] Y. Guo, Z. Huang, J. Guo, X. Guo, H. Li, M. Liu, S. Ezzeddine, M. J. Nkeli, A bibliometric analysis and visualization of blockchain, Future Gener. Comput. Syst. 116 (2021) 316–332.
[6] W. Xiong, L. Xiong, Anti-collusion data auction mechanism based on smart contract, Inf. Sci. 555 (2021) 386–409.
[7] Z. Zheng, S. Xie, H. Dai, W. Chen, X. Chen, J. Weng, M. Imran, An overview on smart contracts: Challenges, advances and platforms, Future Gener. Comput. Syst. 105 (2020) 475–491.
[8] W. Zou, D. Lo, P. S. Kochhar, X.-B. D. Le, X. Xia, Y. Feng, Z. Chen, B. Xu, Smart contract development: Challenges and opportunities, IEEE Transactions on Software Engineering (early access article) PP (2019) 1–1.
[9] X. L. Yu, O. I. Al-Bataineh, D. Lo, A. Roychoudhury, Smart contract repair, ACM Trans. Softw. Eng. Methodol. 29 (4) (2020) 27:1–27:32.
[10] A. Juels, A. E. Kosba, E. Shi, The ring of gyges: Investigating the future of criminal smart contracts, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, ACM, 2016, pp. 283–295.
[11] CoinDesk, Understanding ethereum-blockchain research report, `http://www.coindesk.com/research/understanding-ethereum-report/` (2017).
[12] Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, Blockchain challenges and opportunities: a survey, Int. J. Web Grid Serv. 14 (4) (2018) 352–375.
[13] H. S. Jennath, A. V. S., S. Asharaf, Blockchain for healthcare: Securing patient data and enabling trusted artificial intelligence, Int. J. Interact. Multim. Artif. Intell. 6 (3) (2020) 15–23.
[14] T. Gayvoronskaya, C. Meinel, Blockchain - Hype or Innovation, Springer, 2021.
[15] M. Bartoletti, S. Carta, T. Cimoli, R. Saia, Dissecting ponzi schemes on ethereum: Identification, analysis, and impact, Future Gener. Comput. Syst. 102 (2020) 259–277.
[16] M. Vasek, T. Moore, There's no free lunch, even using bitcoin: Tracking the popularity and profits of virtual currency scams, in: Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, Vol. 8975, Springer, 2015, pp. 44–61.
[17] M. Vasek, T. Moore, Analyzing the bitcoin ponzi scheme ecosystem, in: Financial Cryptography and Data Security - FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, Vol. 10958, Springer, 2018, pp. 101–112.
[18] J. Neisius, R. Clayton, Orchestrated crime: The high yield investment fraud ecosystem, in: 2014 APWG Symposium on Electronic Crime Research, eCrime 2014, Birmingham, AL, USA,, IEEE, 2014, pp. 48–58.
[19] T. Moore, J. Han, R. Clayton, The postmodern ponzi scheme: Empirical analysis of high-yield investment programs, in: Financial Cryptography and Data Security - 16th International Conference, FC 2012, Kralendijk, Bonaire, Vol. 7397, Springer, 2012, pp. 41–56.
[20] W. Chen, Z. Zheng, J. Cui, E. C. H. Ngai, P. Zheng, Y. Zhou, Detecting ponzi schemes on ethereum: Towards healthier blockchain technology, in: Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, ACM, 2018, pp. 1409–1418.
[21] T. Moore, N. Christin, Beware the middleman: Empirical analysis of bitcoin-exchange risk, in: Financial Cryptography and Data Security - 17th International Conference, FC 2013, Okinawa, Japan, Vol. 7859, Springer, 2013, pp. 25–33.
[22] K. Toyoda, T. Ohtsuki, P. T. Mathiopoulos, Multi-class bitcoin-enabled service identification based on transaction history summarization, in: IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), iThings/GreenCom/CPSCom/SmartData 2018, Halifax, NS, Canada, IEEE, 2018, pp. 1153–1160.
[23] W. Chen, Z. Zheng, E. C. Ngai, P. Zheng, Y. Zhou, Exploiting blockchain data to detect smart ponzi schemes on ethereum, IEEE Access 7 (2019) 37575–37586.
[24] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[25] W. Zhou, R. Schlüter, H. Ney, Full-sum decoding for hybrid hmm based speech recognition using LSTM language model, in: 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, IEEE, 2020, pp. 7834–7838.
[26] A. A. Alvarez, F. Gómez-Martin, Motivic pattern classification of music audio signals combining residual and LSTM networks, Int. J. Interact. Multim. Artif. Intell. 6 (6) (2021) 208–214.
[27] P. Tang, H. Wang, S. Kwong, Deep sequential fusion LSTM network for image description, Neurocomputing 312 (2018) 154–164.
[28] Y. Sun, Y. Wu, Y. C. Xu, Using an ensemble LSTM model for financial statement fraud detection, in: 24th Pacific Asia Conference on Information Systems, PACIS 2020, Dubai, UAE, June 22-24, 2020, 2020, p. 144.
[29] S. Ferretti, G. D'Angelo, On the ethereum blockchain structure: A complex networks theory perspective, Concurr. Comput. Pract. Exp. 32 (12).
[30] D. Lin, J. Wu, Q. Yuan, Z. Zheng, Modeling and understanding ethereum transaction records via a complex network approach, IEEE Trans. Circuits Syst. 67-II (11) (2020) 2737–2741.

[31] T. Chen, Y. Zhu, Z. Li, J. Chen, X. Li, X. Luo, X. Lin, X. Zhang, Understanding ethereum via graph analysis, in: 2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018, IEEE, 2018, pp. 1484–1492.
[32] F. Leal, A. E. Chis, H. González-Vélez, Performance evaluation of private ethereum networks, SN Comput. Sci. 1 (5) (2020) 285.
[33] N. Jia, Q. Kong, H. Huang, How similar are smart contracts on the ethereum?, in: Blockchain and Trustworthy Systems - Second International Conference, BlockSys 2020, Dali, China, Vol. 1267 of Communications in Computer and Information Science, Springer, 2020, pp. 403–414.
[34] D. Guo, J. Dong, K. Wang, Graph structure and statistical properties of ethereum transaction relationships, Inf. Sci. 492 (2019) 58–71.
[35] N. Szabo, Smart contracts: Building blocks for digital markets, `http://www.fon.hum.uva.nl/` (1996).
[36] J. A. T. Casallas, J. M. C. Lovelle, J. I. R. Molano, Smart contracts with blockchain in the public sector, Int. J. Interact. Multim. Artif. Intell. 6 (3) (2020) 63–72.
[37] T. Genet, T. P. Jensen, J. Sauvage, Termination of ethereum's smart contracts, in: Proceedings of the 17th International Joint Conference on e-Business and Telecommunications, ICETE 2020 - Volume 2: SECRYPT, Lieusaint, Paris, France, ScitePress, 2020, pp. 39–51.
[38] K. Christidis, M. Devetsikiotis, Blockchains and smart contracts for the internet of things, IEEE Access 4 (2016) 2292–2303.
[39] A. Norta, Creation of smart-contracting collaborations for decentralized autonomous organizations, in: Perspectives in Business Informatics Research - 14th International Conference, BIR 2015, Tartu, Estonia, Vol. 229, Springer, 2015, pp. 3–17.
[40] S. Linoy, N. Stakhanova, S. Ray, De-anonymizing ethereum blockchain smart contracts through code attribution, Int. J. Netw. Manag. 31 (1).
[41] G. Zheng, L. Gao, L. Huang, J. Guan, Ethereum Smart Contract Development in Solidity, Springer, 2021.
[42] M. Artzrouni, The mathematics of ponzi schemes, Math. Soc. Sci. 58 (2) (2009) 190–201.
[43] V. H. Barella, L. P. F. Garcia, M. C. P. de Souto, A. C. Lorena, A. C. P. L. F. de Carvalho, Assessing the data complexity of imbalanced datasets, Inf. Sci. 553 (2021) 83–109.
[44] R. Barandela, J. S. Sánchez, V. García, F. J. Ferri, Learning from imbalanced sets through resampling and weighting, in: Pattern Recognition and Image Analysis, First Iberian Conference, IbPRIA 2003, Puerto de Andratx, Mallorca, Spain, Vol. 2652, Springer, 2003, pp. 80–88.
[45] M. Pérez-Ortiz, P. A. Gutiérrez, P. Tiño, C. Hervás-Martínez, Oversampling the minority class in the feature space, IEEE Trans. Neural Networks Learn. Syst. 27 (9) (2016) 1947–1961.
[46] A. Fernández, S. García, F. Herrera, N. V. Chawla, SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary, J. Artif. Intell. Res. 61 (2018) 863–905.
[47] L. Wang, Y. Zhang, X. Zhu, Concept drift-aware temporal cloud service apis recommendation for building composite cloud systems, J. Syst. Softw. 174 (2021) 110902.
[48] L. Wang, Architecture-based reliability-sensitive criticality measure for fault-tolerance cloud applications, IEEE Trans. Parallel Distributed Syst. 30 (11) (2019) 2408–2421.
[49] R. A. Stein, P. A. Jaques, J. F. Valiati, An analysis of hierarchical text classification using word embeddings, Inf. Sci. 471 (2019) 216–232.
[50] S. Hakak, M. Alazab, S. Khan, T. R. Gadekallu, P. K. R. Maddikunta, W. Z. Khan, An ensemble machine learning approach through effective feature extraction to classify fake news, Future Gener. Comput. Syst. 117 (2021) 47–58.
[51] Z. Ma, J. Ma, Y. Miao, X. Liu, Privacy-preserving and high-accurate outsourced disease predictor on random forest, Inf. Sci. 496 (2019) 225–241.
[52] L. Wang, Y. Zhang, S. Chen, Computation offloading via sinkhorn's matrix scaling for edge services, IEEE Internet Things J. 8 (10) (2021) 8097–8106.
[53] H. Liu, S. Zhang, X. Wu, MLSLR: multilabel learning via sparse logistic regression, Inf. Sci. 281 (2014) 310–320.