Mining trading patterns of pyramid schemes from financial time series data

Future Generation Computer Systems 134 (2022) 388–398

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Mining trading patterns of pyramid schemes from financial time series data

Fang Lv [a,b], Wei Wang [a,b], Linxuan Han [a,b], Di Wang [a,b], Yulong Pei [c], Junheng Huang [a,b], Bailing Wang [a,b,*], Mykola Pechenizkiy [c,*]

[a] School of Computer Science and Technology, Harbin Institute of Technology, Weihai 264209, China
[b] Cyberspace Security Institute, Harbin Institute of Technology, Weihai 264209, China
[c] Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Received 10 January 2021
Received in revised form 17 February 2022
Accepted 20 February 2022
Available online 11 March 2022

Keywords:
Financial time series data
Recursive data mining
Sequence de-noising
Contrast analysis
Trading patterns mining

ABSTRACT

The current studies relating to pyramid schemes are mostly about qualitative analysis, whereas the quantitative analysis is still rare owing to the insufficiency in knowledge of their specific trading modes. Often, the trading modes of pyramid schemes are inconspicuous in financial data, making it difficult to be identified in the data. In this study, we propose a quantitative framework for mining trading patterns of pyramid schemes from financial time series data. The framework includes two parts: **Lo**ng **R**ange **S**equence **D**e-noising (LoRSD) algorithm and Contrast **T**rading **P**attern **M**ining (Contrast TPM) algorithm. LoRSD distinguishes noise items by folding the statistical frequent items and removes the infrequent items recursively. In Contrast TPM, we first identify the frequent one-itemset by comparing the pyramid-related samples with the general samples. Subsequently, a random model is added in the comparative analysis to generate the frequency conditions for mining pyramid scheme patterns. Instead of setting user-defined support thresholds, we adopt contrastive samples as benchmarks in determining the frequency conditions. Our extensive experiments on the financial data set including behaviour of a real-world pyramid scheme demonstrate the effectiveness of our framework in sequence de-noising and mining trading patterns of pyramid schemes from financial time series data.

1. Introduction

A pyramid scheme generally refers to a hierarchical business model that recruits members via a promise of payments or service for enrolling others into a scheme, rather than supplying investments or sale of products [1]. This model can be used to commit fraud by funneling money from bottom to the top of the pyramid. Pyramid scheme based scams differ in the number of layers, the number of members in each layer or the payment rules, resulting in a very flexible structure. This structure makes it hard to detect the abnormal trading behavior of pyramid schemes, thereby allowing it spreads inconspicuously across the globe and threatening the global financial security [2]. In this paper, we investigate the possibility of discovering the knowledge of pyramid schemes from financial time series data.

In financial time series data, the trading activities of pyramid scheme members can be classified into two parts: *purchasing activity* (paying the membership due) and *redeeming activity* (receiving the rebate). The payment follows a specific rule defined by the pyramid scheme which stipulates that the amount of money that each member receives is determined by one's level and the dues that one paid, including one's personal dues as well as dues from one's subordinate members. As illustrated in Fig. 1, for every member, each redeeming activity is caused by one of the previous purchasing activities, implying that the combination of membership due and rebate occur frequently in the financial time series data involving pyramid schemes. The payment due and rebate together form what we hereinafter refer to as trading patterns (TPs) of pyramid scheme. Even though the pyramid schemes can be different in funneling dues and allocating rebates, they are in consistence with TPs. Intuitively, discovering the TPs contributes to the solution of discovering the knowledge of pyramid schemes.

Currently, there are many studies on pyramid schemes from different perspectives. The TPs of the adoption of a known pyramid scheme case is investigated in [3] within a diffusion-of-innovation framework. The study of [4] concludes that the pyramid schemes benefit only a small minority of investors while increasing financial difficulties for the majority of participants and causing severe social conflicts. The micro-structure and the network construction process of the pyramid scheme is researched by analyzing its finance flow network in [5]. In anomaly detection, classifiers for detecting abnormal nodes [6] and communities [1] are designed based on the characteristics in their financial data. These studies have deepened our understanding of the pyramid

* Corresponding authors.
E-mail addresses: f.lyu.hit.tue@gmail.com (F. Lv), wbl@hit.edu.cn (B. Wang).

https://doi.org/10.1016/j.future.2022.02.017
0167-739X/

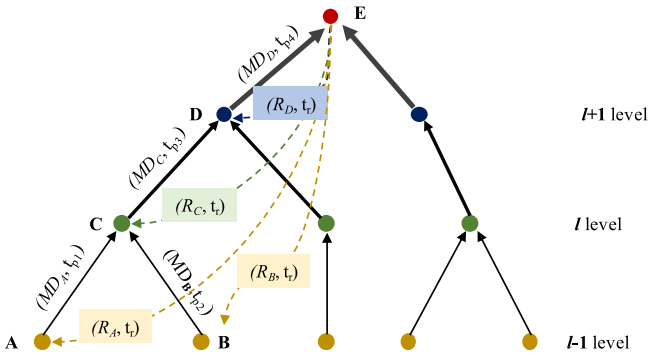


Fig. 1. An example of the trading principle of the pyramid schemes. A to E represent members from different levels. The solid lines represent the purchasing activity (paying membership due (MD)) and the dashed lines represent the redeeming activity (receiving the rebate (R). t_{pi} (where $i \in \{1, 2, 3, 4\}$) and t_r respectively denote the time when paying membership due and receiving the rebate. Note that $\{t_{p1}, t_{p2}\} < t_{p3} < t_{p4} < t_r$. In the pyramid scheme example shown above, members A and B are the subordinate members of C while C is a subordinate member of D. A pays MD_A to C at time t_{p1} and B pays MD_B to C at time t_{p2}. All the dues received by C (MD_C) from its subordinate members (i.e., sum of MD_A and MD_B) are submitted to D at t_{p3}. At time t_r, A, B, C and D receive R_A, R_B, R_C and R_D respectively from E.

scheme from a broader perspective. Many of the current studies focus on qualitative analysis [5] or case analysis [3] while there have been limited quantitative studies [6,7] in the published literatures, due to their insufficiency of the prior knowledge like trading patterns to our best knowledge. In our study, we aim to remove this hindrance by mining TPs from financial time series data using sequential pattern mining techniques.

Sequential pattern mining [8,9] is a widely researched field for extracting patterns from time series data or sequences. Among the various definitions of sequential pattern, two that are of interest to us are periodic pattern and episode pattern. Periodic pattern refers to patterns that the number of events between two occurrences of the pattern is never greater than the maximum period threshold [10] while an episode is defined to be a partially ordered set of events for consecutive and fixed-time intervals in a sequence [11]. TPs overlap with the definitions of periodic and episode patterns, because it refers to a partially ordered set of events that periodically occur in both single time series and the whole time series regardless of the period threshold constraint in periodic patterns and time interval in episode patterns. For sequential pattern mining methods, the support threshold could be user-defined or determined by contrast analysis, such as contrasting different categories of samples. We employ contrast analysis in determining the support thresholds for mining TPs, considering TPs occur more frequently in the financial time series data generated by pyramid scheme members than that of normal accounts. To improve the efficiency of data mining methods, many pruning strategies are used for reducing their search spaces. For financial time series, de-noising the irrelevant items before mining TPs is necessary, as the volume of amount items can be infinite and many of them are infrequent.

Considering the characteristics of TPs discussed in Fig. 1 and the differences among TPs, periodic patterns and episode patterns, we propose a sequential pattern mining framework with contrast analysis. This framework is composed of two components: **Lo**ng **R**ange **S**equence **D**e-noising (LoRSD) algorithm for sequence de-noising and Contrast **T**rading **P**attern **M**ining (Contrast TPM) algorithm for pyramid pattern identification. The main contributions of this article are summarized as follows:

- We propose a new sequence de-noising algorithm called LoRSD with the intention of reducing the volume of the itemset while maintaining the potential trading patterns. LoRSD folds long range frequent sub-sequences while discarding the infrequent items in each sequence.
- We devise algorithm Contrast TPM for mining TPs under our defined contrast frequency conditions, which are formalized by contrasting time series from different categories instead of employing user-defined support thresholds.
- We conduct extensive experiments on the financial time series data set of a realistic pyramid scheme, demonstrating the effectiveness of LoRSD and Contrast TPM on mining trading patterns of pyramid schemes.

The remainder of this paper is organized as follows. We discuss the related work in Section 2. In Section 3, we first describe the definitions and the problem in mining trading patterns. Secondly, the proposed framework containing LoRSD and Contrast TPM components is introduced in detail. In Section 4, we construct experiments to evaluate our framework, and Section 5 is our conclusion.

2. Related works

Our research aims to mine TPs from financial time series data, which are mainly related to three aspects of studies: periodic pattern mining (PPM), serial episode mining and contrast sequential pattern mining.

The PPM is to discover valid periodic patterns in a time-related data set [12]. The mining strategies and the constraints of periodic vary from the difference of applications. In [13], the periodic occurrence was used as a criterion for the proposed periodic high-utility sequential patterns miner (PHUSPM), which could be used to discover all periodic patterns with high profits in the sequential database. In the process of periodic, PHUSPM merges the multiple sequences into a single sequence and then employs the LQS-Tree [14] and two pruning strategies for mining pattern effectively. However, PHUSPM does not guarantee the periodicity of patterns in all sequences. There were many research works for studying the periodicity of patterns in each sequence with consideration of their frequencies in the sequences [10,15], e.g., Philippe et al. [10] defined the periodic pattern when the periodicity appears in a certain ratio of sequences. Instead of mining frequent patterns, other studies focused on mining user-defined periodic patterns [16] appeared, e.g., the algorithm MRCPPS [15] could be used to find rare correlated periodic patterns in multiple sequences depending on the defined mining constraints and data structure. MRCPPS has been successful in discovering all rare correlated periodic patterns, however, its user-defined support thresholds make the obtained patterns have a widely varied periodicity. The pattern-growth algorithm SPP-Growth [17] was proposed to discover patterns with a stable periodicity.

The serial episode [18], representing an ordered sequence of event types, is one of the two categories of the episode patterns. It has been of great interest in many practical applications [19–21] for its ability in capturing causative chains of event types in the data. In alarm management [21], episode mining was applied in discovering the possible combination of the alerts. A meaningful serial episode should correspond to different type of ordered events in different applications, therefore, many occurrence counting strategies have been studied, such as minimal occurrences [22], non-overlapping occurrence [23,24], and with utility measure [25]. Avinash et al. [24] proposed a depth-first search based algorithm for closed serial episode discovery with gap and span constraints. Intuitively, the thresholds of these user-defined strategies influence the effective and time complexity of episode mining methods directly.

A discriminative sequential pattern [26] (a kind of contrast pattern) refers to a sub-sequence that possess significant differences in occurrence frequency in sequences of different classes.

Table 1
The instance of trading sequences.

SID	Sequences
s_1	$(d_1, d_2, d_3, d_4, d_5, d_1, d_3, d_2, d_7)$
s_2	$(d_1, d_3, d_5, d_7, d_1, d_2)$
s_3	$(d_2, d_4, d_6, d_3, d_6, d_2, d_4)$
s_4	$(d_4, d_5, d_2, d_1, d_3)$

Discriminative sequential pattern mining has been one widely researched topic, such as in detection [27], classification [28] and behavior analysis domain [29]. In order to understand the internal relationship between taxpayers' sequential behaviors, Zheng et al. [29] proposed a contrast sequential pattern (CSP) mining approach by using a novel CSP-tree structure. To alleviate the general issue, high false positive of current methods, He et al. [9] integrated multiple hypothesis testing correction process into the pattern mining process. Considering the redundancy issue, He et al. [30] presented a conditional discriminative sequential pattern mining for removing subset-induced redundant patterns by imposing an additional constraint on the redundancy of each pattern with respect to its sub-patterns.

In Summary, the major steps of sequential pattern mining methods are: pruning searching space, determining the support threshold and optimizing the mining process. Instead of using constrains for reducing searching space in the process of pattern mining (like gap constrains), in this paper, we prune the infrequent items before mining TPs. TPs mining cannot be solved by existing episode or periodic pattern mining methods, as TPs do not simply belong to each of them. Specifically, the TPs mining method should satisfy the contrast frequency and ordering requirements. Different from the contrast analysis used in [9], we present a suitable filtering criteria strategy [29] for mining TPs. Moreover, we generate series of contrast frequency conditions on the basic of sequential analysis without using user-defined support thresholds.

3. The framework for mining TPs

In this section, we build a framework for mining trading patterns based on contrast analysis. Firstly, we state the problems and definitions following the overview of our framework. Secondly, we construct the LoRSD and the Contrast TPM algorithms respectively. Finally, we analyze the complexity of our framework.

3.1. Problem statements

A trading time series data set containing N data series is represented as $S_N = \{s_1, s_2, \ldots, s_N\}$, where $s_k \in S_N$ denotes the kth time series. Given $s_k = (d_{k1}, d_{k2}, \ldots, d_{km_k})$, where d_{kj}, $j \in [1, m_k]$, denotes an item. All the unrepeated items in the set $\{d_{ij}\}, i = 1, 2, \ldots, N, j = 1, 2, \ldots, m_i$, form an itemset D, and s_k is regarded as a chronologically arranged list of items. Omitting the time stamp, s_k can be simply seen as a sequence. The format of trading sequences in our scenario, for example, can be shown in Table 1.

For s_k, its length equals the sum of the occurrence of each item in it. Notably, here, each item in S_k (as shown in Table 1) represents only one element and cannot be a set of elements. This formation is different from the sequences that were introduced in the work of many other researchers. For instance, in [31], some of the items in $s_x = (\langle a\rangle, \langle a, b, c\rangle, \langle a, c\rangle, \langle d\rangle, \langle c, f\rangle)$ just represent a single element (e.g., $\langle a\rangle$, $\langle d\rangle$), while others contain several elements (e.g., $\langle a, b, c\rangle$). To specify the characteristics of trading patterns in pyramid schemes, we firstly state the definitions of trading sequence and trading item respectively.

Definition 1 (*Trading Sequence*)**.** A trading sequence $s_a = (e_1, e_2, \ldots, e_n)$ denotes the chronological ordered trading record of an account a ranging from timestamp a_{t1} to a_{tn}, where $e_k \in s_a$ represents the kth payment amount from or to account a, which occurs at timestamp a_{tk}.

Definition 2 (*Trading Item*)**.** Given an itemset D, formed by the whole trading sequences, an item is called a discriminating trading item if it is an element in the purchasing due set Φ_{MD} or redeeming rebate set Φ_R, here $\Phi_{MD} \subset D$ and $\Phi_R \subset D$.

As mentioned in Section 1, a sequential pattern is a subsequence from sequences that satisfies filtering criteria about frequency of occurrence in sequences. Given a trading sequence s_a, the ordered item list $s_\alpha = (a_1, a_2, \ldots, a_m)$ is called one of its sub-sequences if there exist integers $1 \le j_1 < j_2 < \cdots < j_m \le n$ such that $a_1 = e_{j_1}, a_2 = e_{j_2}, a_m = e_{j_m}$. Intuitively, in trading sequences, the occurrence of TPs are relatively frequent in sequences formed by pyramid scheme members (positive sequences) than that of non-pyramid scheme members (negative sequences). To discover the TPs that distinguish positive and negative sequences, we constrain TPs with contrast conditions as below.

Definition 3 (*Trading Patterns*)**.** A sub-sequence $s_\mu = (p_{i_1}, p_{i_2}, \ldots, p_{i_k})$ is a trading pattern if **(1)** $\forall p_{i_j} \in \Phi_{MD} \bigcup \Phi_R$; **(2)** it occurs more frequently in positive sequence set $SP = \{s_{a_{p_1}}, s_{a_{p_2}}, \ldots, s_{a_{p_N}}\}$ than in negative sequence set $SN = \{s_{a_{n_1}}, s_{a_{n_2}}, \ldots, s_{a_{n_M}}\}$, where $s_{a_{p_n}} \in SP$ and $s_{a_{n_m}} \in SN$ are the trading sequences of a pyramid scheme account a_{p_n} and a normal account a_{n_m} respectively.

A big search space caused by redundant items is a vital obstacle to discover patterns effectively. In discriminative sequential pattern mining methods, many pruning strategies [29,32] were proposed for solving redundancy issues by using different pattern evaluation measures or adopting different pattern searching data structures [30]. The pruning methods in the search space formed by trading sequences are even trickier than that in the widely researched sequences. The two major reasons are (1) a large capacity itemset caused by a lot of rare items, such as fractions and big trades; and (2) the existence of disturbing frequent amounts that occur frequently than others naturally, such as 100,500. Time complexities of the subsequent pattern mining algorithms would be increased by a large searching capacity, meanwhile. Furthermore, the mining efficiency would be reduced by the disturbing items. A beforehand sequence de-noising process enables the sequential pattern mining to perform on a relatively pure data set. The left of Fig. 2 illustrates the process of our sequence de-noising algorithm with the related definition given below.

Problem 1 (*Sequence De-noising*)**.** Given trading sequences $S = \{s_1, s_2, \ldots, s_M\}$, the task of sequence de-noising is to remove the irrelevant items from each $s_m \in S$ to make it short while retaining the trading items.

This is a general statement. Especially, in our study, it is worth noting that,

- De-noising sequences without disrupting the structure of sequences before knowing the ultimate trading patterns is a difficult task. Moreover, the definition of irrelevant item for trading patterns is ambiguous. Therefore, the remained items should occur frequently in the sub-sequences of trading sequences.
- To solve this problem, we introduce recursive data mining (RDM) method [33] into our sequence de-noising (LoRSD) algorithm because it extracts long range frequent subsequences without using any prior knowledge.

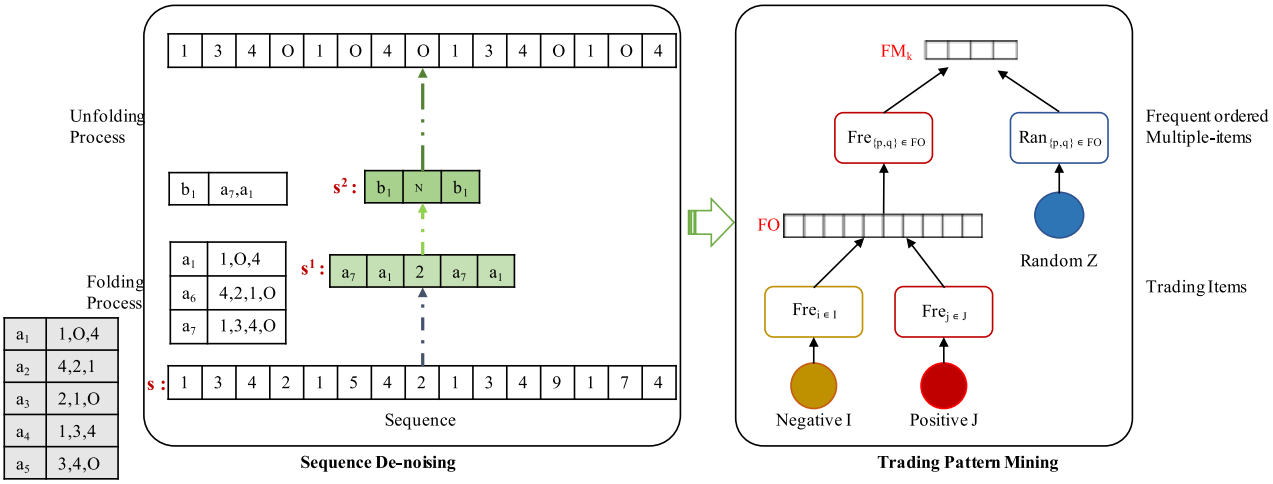


Fig. 2. The framework for mining trading patterns of pyramid schemes.

After being de-noised by LoRSD algorithm, the trading sequences that consist of the most trading items are ideally suited for mining TPs comparing to the raw sequences. The process of mining TPs is shown in the right of Fig. 2 containing two continuous phrases: (1) extracting the trading items, and (2) mining trading patterns. The two components of our framework are discussed in details in the following parts separately.

3.2. Sequence de-noising algorithm

The optimal result of the above-mentioned sequence de-noising method is that all of the trading items are retained while the whole irrelevant items are eliminated. To achieve this goal, we propose a long range sequence de-noising (LoRSD) algorithm.

The RDM [33] algorithm is one of the most suitable fuzzy pattern mining methods and pattern mining methods for sequential noisy data [34,35]. The idea of RDM is originally proposed to distinguish the roles of communicators in a social group by discovering the statistically dominant sequential patterns from stream data. The key properties of RDM algorithm [36] include: (1) no restriction on gap intervals of the items in a sequential pattern, which retains the related items to the greatest extent, (2) tolerant of approximate matching, which can be used to eliminate the noise items without excessive pruning, and (3) the recursive mining process, because of which the structure of the sequences is preserved. The three properties match exactly with the requirements of sequences pretreatment for the subsequent TPs mining. Therefore, the idea of RDM is used in solving Problem 1.

LoRSD is a hierarchical mining process containing two processes: folding process and unfolding process (see the left part of Fig. 2). The folding process conducts recursively with each recursion contains two steps. Firstly, the statistically significant sequential patterns are recorded and assigned by new tokens for the next loop. In details, given a trading sequence $s_a \in S$ and itemset D, for an item $d \in D$ at each index position in $\forall s_a \in S$, LoRSD works on its created sub-sequences s_{fw} which is fragmented by a sliding window $w = (l, g)$ to select its dominant replacement from the candidate sub-sequence set. Here, l denotes the length of w, and g denotes the maximum number of wildcards in w. A wildcard is a symbol for replacing any item in a sub-sequence. The number of wildcards in w ranges from 0 to g in the process of obtaining the dominant replacement of s_{fw}. Hence, the dominant replacement of s_{fw} is the one candidate with the highest significance score $\mathbf{s}^r_{fw}$.

$$\mathbf{s}^r_{fw} = \frac{\prod_{k=1}^{(l-r)} P(x_k)}{(R_a)^{(l-g)}} \tag{1}$$

where, $P(x_k)$ means the probability of item x_k over S, and $R_a = \frac{1}{|I|}$ represents the random distribution of each item over S. The power exponent $l-r$ implies that only the non-wildcard items are considered (i.e., both $P(x_k)$ and R_a of the wildcard equal 1). A s_{fwd_r} is considered to be a valid candidate sub-sequence of s_{fw} if its significance score $s^r_{fwd_r}$ is bigger than the threshold β. According to the number of r, the $d \in D$ in each index position of $s_a \in S$ will generate a valid candidate sub-sequence set fwd. In this paper, β is set as an empirical value 10^{-5} for our data set.

Secondly, LoRSD operates on the re-written sequences until it reaches the stop condition. Given the dominant sub-sequences FW of S, we iterate over each index position in $s_a \in s$ by checking whether its dominant sub-sequence $fwd \neq \varnothing$, if so we replace the sub-sequence fragment (starts from the current position till after l items) with fwd, otherwise move to the next positions. In addition, the dominant sub-sequences are labeled as new symbols for next loop, meanwhile, the items that are not replaced for more than k loops are tagged as noise items (labeled by another wildcard N).

LoRSD repeats the former two steps till there are no more dominant sub-sequences from the current loop. In this hierarchical folding process, a long range combination of items (containing wildcards and new symbols) can be folded into a sliding window and will be considered as a dominant sub-sequence no matter how big the intervals of the items in the original sequence are. For simplify, we introduce how LoRSD uses RDM in solving Problem 1 in Algorithm 1. For more details refer to [33].

Algorithm 1 LoRSD Algorithm

Input: S, l, g, k, β
L, S_{DL}, *noiseItems*, *tokenItems*;
Output: S_{Pru}, *pruneItems*
1: $S_{DL} == S$
2: **while** S_{DL} is not updated **do**
3: *Valids* = **getValid**(S_{DL}, l, g, β)
4: *tokenItems* = **getDominants** (*Valids*, L, l, g)
5: *tokenItems* = **tandem**(S_{DL}, *tokenItems*, L)
6: S_{DL} = **replaceTokens**(S_{DL}, *tokens*)
7: *noiseItems*, S_{DL} = foldNoise(S_{DL}, k)
8: **end while**
9: S_{Pru} = **unfold**(S_{DL}, *noiseItems*, *tokenItems*)
10: **return** S_{Pru}, *noiseItems*

As shown in Algorithm 1 from Line 3 to 6, LoRSD generates dominant sub-sequences of S_{DL} in loop L. For the item in each index position of $s \in S_{DL}$, LoRSD generates its candidate valid sub-sequence and then select the one with the highest significance

sore as its dominant sub-sequence. The neighboring dominant sub-sequences are connected to tandems if possible. Afterward, S_{DL} is replaced by the ultimate dominant sub-sequences by carrying out a traversal operation. Line 7 explains the process of marking the noise items with an N. In the unfolding process (line 9), LoRSD flattens S_{DL} from the last loop with the stored dominant sub-sequences. In this algorithm, the optimal values of parameters g, l, k vary with data sets. We discuss how to select the optimal values in Section 4.4.

3.3. Contrast TPM algorithm

Given noise-free trading sequences, the target of this section is to mine TPs (Definition 3). In this section, we propose a Contrast TPM algorithm on the basis of contrast analysis without using user-defined support parameters.

Determining the support thresholds is a vital component for pattern mining methods [15,37]. Similar to [9], we employ the difference between positive samples and comparable data in mining TPs. Specifically, Contrast TPM obtains trading items by mining frequent one-items (i.e., Definition 4), and then discover TPs by mining frequent ordered multiple-items (i.e., Definition 5).

Definition 4 (*Frequent One-itemset*)**.** Given negative sequences SN and positive sequences SP, $O_n = \{p_1, p_2, \ldots, p_n\}$, $p_i \in SN \bigcup SP$ is called a frequent one-itemset if $\forall p_i \in O_n$ satisfies,

$$f(SP, p_i)/f(SN, p_i) > \alpha. \tag{2}$$

where, $f(x, y)$ is a function for calculating the frequency of y in x, α is the threshold coefficient, meaning $\alpha * f(SP, p_i)$ is the frequent threshold of p_i.

The support threshold of each one-item is dynamically determined by the trading sequences, containing positive and negative sequences.

As discussed in Section 1, TPs should satisfy the order-preserving requirement of trading items stemming from the concomitance of receiving rebates with paying membership dues. For instance, given a trading pattern $\langle md, r\rangle$, the event of submitting md triggers the event of receiving b, and not vice versa. Moreover, the co-occurrence of trading items is more frequently in the trading sequences of pyramid schemes than that in random sequences. Hence, in addition to the negative sequences, the random model is applied into the frequency conditions of multiple-items. The formulation of our frequent ordered multiple-itemset is given as below.

Definition 5 (*Frequent Ordered Multiple-itemset*)**.** Given SP, the previous itemset M_k and comparison model λ, $M_{k+1} = \{m_{i,j} = \langle i, j\rangle | i, j \in M_k\}$, is called a frequent ordered multiple-itemset if each $\forall m_{i,j}$ satisfies the following contrast frequency conditions,

$$\digamma(SP(m_{i,j})) \gtrdot \digamma(\lambda(m_{i,j})) \tag{3}$$

where $\digamma(x)$ represents a function family for calculating the frequency of event x under different conditions, and $SP(y), \lambda(y)$ denote the events that y occurs in SP and λ respectively, and $\gtrdot$ explains the comparison operator.

In Definition 5, $\digamma$ and $\gtrdot$ constrain the frequent conditions from both high co-occurrence and order-preserving perspectives. Notably, its conditions do not constrain the gap or interval between items in different with most of the current studies [24,37]. For $m_{i,j}$, to guarantee its relatively frequent co-occurrence in SP comparing to in λ, the coming three conditions need to be satisfied: $\mathbf{C_1}$ **(Condition 1))**, in SP, at least one of its item occurs more frequently than its random probability (shown in Eq. (4)); $\mathbf{C_2}$**)**, the co-occurrence frequency of i, j in the same sequences is higher than the joint possibility of i and j in a random model (shown in Eq. (5)); $\mathbf{C_3}$**)**, the difference between the sequence distributions of i and j is less than a random value (shown in Eq. (6)).

$$max(C_{SP}(i), C_{SP}(j)) > C_R(x) \tag{4}$$

$$\frac{SC_{SP}(i, j)}{L_S} > Pr_{SP}(i) * Pr_{SP}(j) \tag{5}$$

$$\frac{min(SC_{SP}(i), SC_{SP}(j))}{max(SC_{SP}(i), SC_{SP}(j))} > \frac{min(SC_R(i), SC_R(j))}{max(SC_R(i), SC_R(j))} \tag{6}$$

where, x, i, j denote random event, event i and event j respectively; (i, j) represent the event that i and j occur in the same sequence, called co-occurrence. For SP, $C_{SP}(x)$ represents the total number of times event x occurs in the whole data set, $SC_{SP}(x)$ means the number of sequences that contain x (these two notations also apply to the random model R). Not ably, $\digamma$ comply with the non-overlapping constraint [38] in the counting of the occurrence of multiple-items.

To guarantee the concomitant occurrence of i and j, $m_{i,j}$ should satisfies the coming three order-preserving conditions: $\mathbf{C_4}$**)**, the two events of (i, j) occur in a specific order in most sequences, ensuring i and j are the antecedent and consequent items of a trading pattern respectively. In short, the frequency of event $(j|i)$ is higher than its random probability (shown in Eq. (7)); $\mathbf{C_5}$**)**, if a sequence contains a i, then it tends to contain a j as well. In other words, the conditional probability of (i, j) in sequences is higher than that in the whole data set (shown in Eq. (8)); $\mathbf{C_6}$**)**, the co-occurrence of i and j shows a higher probability in keeping the order $\langle i, j\rangle$. Briefly, the distribution of $(j|i)$ to (i, j) in sequences is higher than that in the whole data set (shown in Eq. (9)).

$$C_{SP}(j|i) > C_{SP}(i, j)/2 \tag{7}$$

$$\frac{SC_{SP}(i, j)}{SC_{SP}(i)} > \frac{C_{SP}(i, j)}{C_{SP}(i)} \tag{8}$$

$$\frac{SC_{SP}(j|i)}{SC_{SP}(i, j)} > \frac{C_{SP}(j|i)}{C_{SP}(i, j)} \tag{9}$$

To sum up, the former defined contrast conditions guarantee the relative higher co-occurrence frequency of ordered multiple-items in both an individual sequence and the whole data set, meanwhile, satisfy the requirement of order-preserving of frequent one-items. By following Definition 4 and 5, Contrast TPM algorithm conducts the frequent one-itemset mining and multiple-itemset mining processes successively. The TPs are separated from the others without using user-defined certain support thresholds by using contrast analysis. The workflow of Contrast TPM algorithm is shown in Algorithm 2.

As shown in Algorithm 2, Contrast TPM contains two phases: frequent one-itemset mining (Line 1 to 7) and frequent ordered multiple-itemset mining (Line 8 to 16). In the first phase, the frequency of each item in positive and negative sequences is calculated (Line 1) and then is used for determining whether it is a discriminative frequent item (Line 2). Subsequently, in the second phase, the PrefixSpan idea [39] is used for mining the frequent multiple-items (two-items for the trading patterns of pyramid schemes) that satisfy Definition 5.

3.4. Computational complexity

The complexities of the two algorithms of our framework are analyzed below.

- Sequence De-noising algorithm. In RDM process, the worse case time complexity of the folding process is $\Theta(N^{\log_w N})$ [40], where N, w are the length of sequences and size of the sliding window respectively.

Algorithm 2 Contrast TPM algorithm

Input: SN, SP, α, μ, ν
Output: *FreOne*,*FreMulItems*
1: FP_1, FN_1 = **getFrequency**(SN, SP)
2: **FreOneItems**(FP_1, FN_1, α)
3: **for** $\forall p_i \in FP_1$ **do**
4: **if** $p_i \vDash$ Formula (2) **then**
5: *FreOne*.append(p_i)
6: **end if**
7: **end for**
8: **PrefixSpanMulItems**(*FreOne*, SP, SN, μ, ν)
9: *Proj* = **getProjections**(*FreOne*, SP)
10: **getStatistics**(*Proj*, SP, SN)
11: **return** $SC_{SP}, C_{SP}, SC_{SN}, C_{SN}$
12: **for all** *Proj* **do**
13: **if** *Proj* $\vDash$ Definition 5 **then**
14: *FreMulItems*.append(*Proj*)
15: **end if**
16: **end for**

- Contrast TPM algorithm. There are two phases in this algorithm. The time complexity of frequent one-itemset mining is linear to the number of items in positive and negative sequences, which is $O(M)$. For the second phase, the PrefixSpan method saves a lot of memory space comparing to other data structures. The time complexity of this phase is linear with the respect to the number of sequential patterns $O(M^k)$ (for the worst-case scenario that M items remain), k means the length of the multiple-item.

In summary, the framework is approximately linear to the volume of items of each process. Therefore, the more effective the previous process is, the more efficient the current one will be.

4. Experiments

In this section, we evaluate the effectiveness of our proposed algorithms empirically. we first introduce the data set and experimental settings, then discuss our experimental results in the trading pattern mining task.

4.1. Data set

We use a real-world trading data set, relating to a pyramid scheme organization, from a security institution in China. The operational organism of this specific pyramid scheme case is discussed comprehensively in [1]. In brief, the participants must spend 3800 yuan to 69,800 yuan to purchase 1–21 virtual products (so as to qualify for membership), and all members are organized into 5 levels for collecting dues and receiving rebates. Our data set covers the time series trading records of 2275 bank accounts. Each record is a chronological trading sequence that consists of the transactions starting from the time when the account was activated. The trading sequences of these bank accounts are labeled as positive (formed by pyramid scheme members) and negative (formed by normal accounts) by providers according to the forensic evidences. The trading items and trading patterns are frequent in the positive sequences whereas are sparse in the negative sequences, the details of which are listed in Table 2 and Table 3 respectively. As shown in Table 2, the numbers of transaction amounts and average lengths of sequences in these two kinds of samples are similar. The number of purchasing dues is 43 in total, 2 of which are rounded from other dues (see Table 3). Furthermore, the sophisticated trading behaviors

Table 2
Statistics of positive and negative sequence samples.

Sample	Number of sequences	Number of transaction amounts	Average length of sequences
Positive	846	4630	54.9
Negative	1804	4144	40.8

are concluded into unit items (i.e., A, B) and times of them. For instance, the purchasing due 69800 is composed of one time of 3800 (i.e., unit A) and 20 times of 3300 (i.e., unit B). The relationship between the two units of redeeming rebates is ***or***, which means only one unit is used for calculating the rebates (with the time decided by the correlated purchasing due).

4.2. Experimental setup

To the best of our knowledge, no existing methods can be directly used to identify trading patterns of pyramid schemes from time series financial data. However, some existing techniques can be extended for our problem, we select the following two representative periodic sequential pattern mining methods as our baselines for demonstrating the effectiveness of our proposed framework.

- MPFPS [10] considers the periodicity of patterns in each sequence and their frequencies in the overall database. It defines the periodic pattern in a sequence and sequence periodic ratio for the whole database with using upper-bound as a pruning strategy for reducing the search space.
- MRCPPS [15] focuses on discovering rare correlated periodic patterns in multiple sequences.
- Contrast TPM($\mathbf{C}_i$). This is a series of variations of our Contrast TPM algorithm, each of which ignores one of the conditions in Definition 5 (e.g., $i = 1$ denotes the first condition is ignored).

4.3. Metric methods

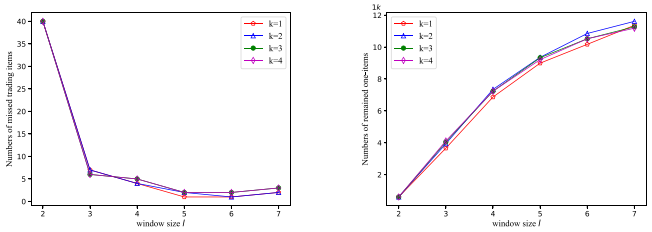
The effectiveness and efficiency of the algorithms are compared by measures of **number of patterns** and **coverage**. The definitions of these two metrics vary along with the specific function of each testing parameter. Therefore, we give the explicit definitions in the corresponding parts of this paper separately. According to the definition of TPs, we only discover the ordered multiple-items with length no more than 2 items by using our Contrast TPM algorithm, while for the other methods, the length of the obtained patterns is determined by how their algorithms work. On MPFPS, we operate the mining process using Breadth-first search strategy with the parameters as $maxStd = 30$, $minRA = 0.01$, $maxPer = 45$, $minSup = 2$. The reason is that with such a setting, MPFPS achieves the best trade-off between effectiveness and efficiency. For MRCPPS, the optimal parameters are set as $maxSup = 50$, $maxStd = 50$, $minBond = 0.01$, $minRa = 0.01$. We determine the optimal values for the parameters in our framework through empirical experiments in the coming section.

4.4. Effectiveness study

In this section, we test how parameter settings influence the effectiveness of our proposed pyramid pattern mining framework and then determine their optimal values. To demonstrate the robustness of the optimal parameters, the time series data is split into a training set and a testing set by a time point of observation, each of which is used for determining the optimal values and verify their efficacy in general financial time series data. The

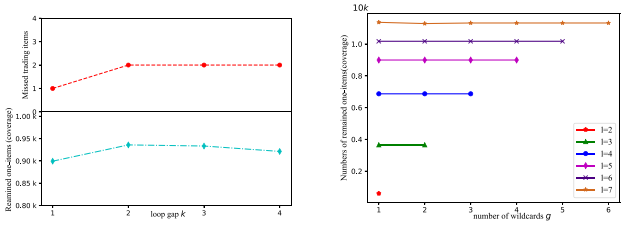
Table 3
Statistics of trading items containing purchasing items and redeeming items.

Trading activities	Total number of trading items	Number of unit A	Times of A	Composition operator	Number of unit B	Times of B
purchasing	43(2)	1	$\mathbb{N} \in [0, 1]$	+	1	$\mathbb{N} \in [0, 20]$
redeeming	102	12	$\mathbb{N}^+ \in [1, 11]$	or	2	$\mathbb{N}^+ \in [1, 21]$



(a) Numbers of missed trading items. (b) Coverage.

Fig. 3. Coverage *w.r.t. l*.



(a) Varying k. (b) Varying g.

Fig. 4. Varying k and g.

division time point is set to the timestamp that is half a year ago till the end of our data set, referring to the certification of the time-sensitive attribute of pyramid scheme behaviors in [41]. Given the training set, the optimal parameters are selected based on the two self-defined metrics. Subsequently, we evaluate the robustness of these optimal values on testing data.

4.4.1. Parameter analysis

There are four influential parameters in our framework, three of which are l, g, k that come from the LoRSD algorithm and the other one is α of Contrast TPM algorithm. The metrics are named as **number of patterns** and **coverage**, where patterns represent the trading item (shown in Table 3) and coverage represents the number of retained items or the ratio of it to the entire original itemset. In this section, We analyze the four parameters on the training data separately.

Varying *l*

Given $g = 1$, we examine the effect of l from LoRSD in the effectiveness of de-noising. The ideal outcome of the de-noising process is that most non-trading items are pruned while all the trading items remain.

The influence of the variation of l on the effectiveness is tested by varying k from 1 to 4, and the results are shown in Fig. 3. For $k \in 1, 2, 3, 4$, the variation tendencies of the curves in each k are similar (shown in Fig. 3(a)). For each k, the numbers of missed trading items show three changing phases, it reduces sharply when l increases from 2 to 3, and then the declining trend slows down, and then it turns to an upward trend.

Fig. 3(b) illustrates the influence of the variation of l to the number of remained items when varying k from 1 to 4. The curves show a similar shape along while changing l. It increases rapidly when raising l from 2 to 4, and turns to a relatively slow increasing speed after that. In short, the **number of patterns** and **coverage** change reversely with l raising. Therefore, a proper l should retain more trading items while prunes more non-trading items. Even though when $l = 2$, the coverage reaches the minimum value 599, the number of missed trading items increases up to 40 (i.e., 27.2% of trading items) which is undesirable. In addition, in the situation of raising l from 4 to 6, take $k = 2$ as an example, the missed trading items reduce from 4 to 1, however, the coverage enlarges from 7349 to 10855. Given $l = 5$, most of ks reach their minimum numbers of missed patterns. Therefore, to trade-off these two indicators, we select $l = 5$ as the optimal parameter for LoRSD.

Varying k

Given $l = 5$, we test how different loop gaps for de-noising influence the effectiveness of the **number of patterns** and **coverage**. As shown in Fig. 4(a), the two curves both increase with varying k from 1 to 2, and decelerate the growth in the following situations. The optimal value of the parameter k is 1. The result signifies LoRSD performs best when it replaces the noise items that have not been covered in the former one loop into the noise symbol.

Varying g

Given $k = 1$, the number of wildcard g relies on l, of which the value ranges from 1 to $l - 1$. In this section, we test how g influences the coverage of LoRSD. As shown in Fig. 4(b), in the y direction, the coverage arguments obviously with increasing l, which is correspond with the conclusion in 4.4.1. In the x direction, the number of remained trading items for $\forall l \in [6, 7]$ decreases slightly with changing g from 1 to 2, while keeps the same for $l \in [3, 4, 5]$. For $l \in [6, 7]$, the coverage contains the same number of trading items when g is increased from 3 to $l-1$. The fluctuation of $l = 7$ shows a tiny reduction in the coverage accompanies by raising g from 2 to 3. The results indicate that the numbers of remained trading items change positively with l, while negatively with g (*iff* given l). Therefore, for $l = 5$, the optimal value of g is 1.

Varying α

Contrast TPM conduct the pattern mining process on the de-noised training data, which is generated by the LoRSD algorithm with the optimal parameters (e.g., $l = 5, k = 1, g = 1$). During the pattern mining process, a parameter α is used as a ratio support threshold for mining relative frequent one-items in positive samples compared to negative ones. We analyze the effectiveness of α while varying it from 0.7 to 1.5, and report the results in Fig. 5.

As is shown in Fig. 5, the x axis represents the values of α, y axes replay the number of missed trading items and the ratio of the remained trading items for a given α (x-axis). The K-shape of the two curves indicates that they are negatively correlated. The ratio decreases sharply with increasing α from 0.5 to 1.1, meanwhile, the number of missed trading items augments at a relatively slow speed. For the training data set, $\alpha = 0.8$ is seen as the optimal trade-off between computational complexity (coverage ratios) and accuracy (remained trading items). In this situation, more items containing trading items are remained, enabling the high effectiveness of the whole TPs mining process.

Table 4
Numbers and coverage of the mined patterns.

Methods	Total patterns (one-items)	Trading items	Binomial patterns[a]	TPs	Multiple patterns[b]	TP ratio (%)
MPFPS	4231(176)	34	59	0	1	0
MRCPPS	574227(721)	79	218	7	1160	3.21
Contrast TPM	44458(2019)	105	1010	**359**	–	**35.54**

[a]Denotes the multiple-items in the length of 2, of which both two items are trading items.
[b]Denotes the multiple-items longer than 2, in which more than 1/2 items are trading items.

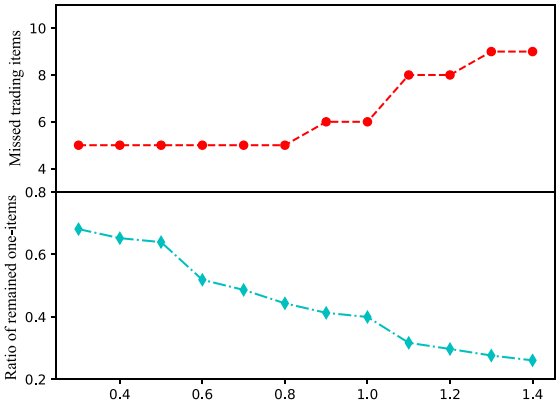


Fig. 5. Numbers of missed trading items and coverage *w.r.t.* α.

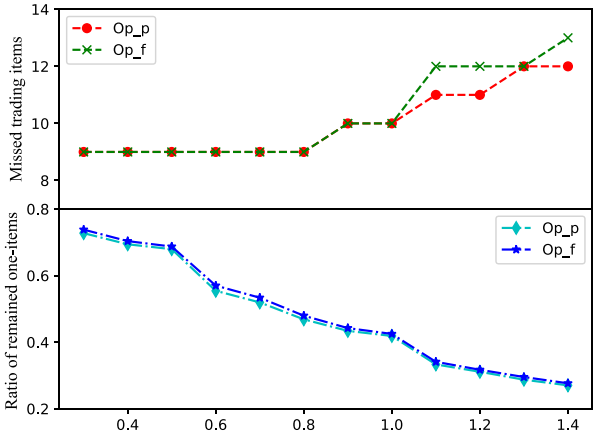


Fig. 6. Performances of the two Optimal αs.

4.4.2. Robustness analysis

The optimal parameters on training data should be robust on general financial time series data. The efficacy of the optimal parameters is confirmed by comparing their performances on testing data. To demonstrate the robustness of the optimal parameters, we search for the optimal values of LoRSD on testing data and then compare the performance differences of these two sets of optimal parameters.

Analyzing the optimal parameters on testing data

The performance of different l and k on testing data is shown in Table 5, in which the attributes items and coverage represent the number of missed trading items and the number of retained amount items (figures in thousands).

The optimal value of l, denoted as l_f, on testing data can be determined by analyzing its relation with items and coverage. As discussed in Section 4.4.1, with the increase of l, the number of missed trading items and coverage change to two inverse directions. The l_f should be the best trade-off of these two indicators. For $l = \{2, 3\}$, the minimum **items** is 16 meaning 10.88% of trading items are missed. Even though the coverage meets the expectation, missing too many trading items will lead to a low recall of the subsequent pattern mining algorithm. For $l = \{6, 7\}$, the majority of the cases are high in coverage and low in items, this may lead to their low accuracies in pattern mining. In contrast, the two indicators of $l = \{4, 5\}$ reach an acceptable balance. Therefore, l_f is set as 4 or 5.

Given $l = \{4, 5\}$, we search for the optimal value of k, denoted as k_f, by considering items and coverage at the same time. When $l = 4$, the items of $k = 2$ reaches its minimum, and coverage does not increase obviously comparing to the cases of $k = \{1, 3, 4\}$. When $l = 5$, the items reduce with enlarging k, while the **coverage** shows a general rising trend. Therefore, $k = \{1, 2\}$ generates the best trade-off results.

Robustness of the optimal l and k

As discussed above, the optimal parameters on training set and testing set are $Op_p = \{l_p = 5, k_p = 1\}$ and $Op_f = \{l_f = \{4, 5\}, k_f = \{1, 2\}\}$ respectively. In this case, Op_p is a subset of Op_f, meaning that the two optimal parameters of training data works well on the testing data.

Robustness of the optimal α

The optimal α of one de-noised data set is considered to be robust if it works well on the other de-noised data set. Given the testing data, LoRSD generates two sets of de-noised trading sequences by using Op_f and Op_p respectively. We analyze the robustness of α by comparing its performance in two de-noised data sets.

Take $Op_f = \{l_f = 4, k_f = 2\}$ as an instance, the performances of α with Op_f and Op_p are depicted in Fig. 6. The performances of α show almost the same changing trend and overlapping in most values. This indicates that α is robustness to Op_f and Op_p. Moreover, given $\alpha = 0.8$, i.e. the optimal α on training data, the two metrics in Fig. 6 show a good trade-off on testing data. It means that α is not sensitive to data set.

4.5. Efficiency study

This section estimates the efficiency of Contrast TPM by conducting statistical analysis on the results of pattern mining, and then tests the efficiency of patterns in detecting fraud bank accounts.

4.5.1. Efficiency of pattern mining

In this subsection, we execute these four algorithms (i.e., MPFPS, MRCPPS, Contrast TPM, Contrast TPM(C_i)), under their optimal setting situations, in mining trading patterns. Here, Contrast TPM and Contrast TPM(C_i) are inputted with de-noised sequences and others are fed with original trading sequences. The number of obtained patterns are given in Table 4, from where the coverage of the results are derived (shown in Tables 5 and 6). The coverage is expounded by the recall and precision [42].

As indicated in Table 4, Contrast TPM derives 44175 frequent patterns containing 2019 frequent one-items and 42439 binomial-item patterns. The results cover **105** trading items and

Table 5
The performances of l and k on testing data.

l	k							
	1		2		3		4	
	Items	Coverage	Items	Coverage	Items	Coverage	Items	Coverage
2	46	0.624	46	0.624	46	0.624	46	0.624
3	19	3.357	17	3.591	16	3.695	16	3.735
4	10	6.047	6	6.310	7	6.276	7	6.191
5	7	7.477	5	7.835	5	7.605	5	7.621
6	6	8.583	6	8.829	7	8.821	7	8.829
7	4	9.077	4	9.401	4	9.383	4	9.383

Table 6
Recall and precision of different methods.

Methods	One-item recall (%)	Binomial recall (%)	TPs recall (%)	One-item precision (%)	Binomial precision (%)
MPFPS	23.45	0.67	0	**19.32**	**8.14**
MRCPPS	54.48	2.49	0.16	10.54	3.90
Contrast TPM	**72.41**	**15.2**	**13.52**	5.2	2.57

Table 7
Comparing the series of Contrast TPM algorithms.

Contrast TPM	Binomial recall (%)	TPs recall (%)	Binomial precision (%)	TPs precision (%)	Dominant ratio (%)
C_1	12.80	8.6	2.55	0.855	33.57
C_2	7.48	5.36	1.97	0.706	35.82
C_3	15.86	10.81	2.41	0.822	34.07
C_4	18.09	15.96	0.69	0.303	44.11
C_5	0.09	5.02	1.92	0.548	28.50
C_6	8.73	7.52	1.942	0.837	43.08
Our	15.2	13.52	2.57	1.141	44.39

1010 (binomial) trading patterns. In particular, the number of trading patterns up to 359, meaning that 35.54% the whole trading patterns are discovered. The coverage of Contrast TPM is expounded by the Recall and Precision in Table 6, in which the frequent one-itemset, binomial pattern set, and trading pattern set are analyzed respectively. For one-itemset, the recall rates of Contrast TPM and MRCPPS are close, and the precision rates of Contrast TPM is far below those of MPFPS and MRCPPS. The reason is principal that Contrast TPM takes one-item mining as a precursor, aiming to remain as many as feasible relative frequent items for the subsequent multiple-item mining process. For the sequential patterns, Contrast TPM recalls 15.2% binomial patterns which is far exceed than those of MRCPPS and MPFPS, however, its precision is less than the others. The small precision value of binomial is caused by its large volume of frequent binomial patterns. In spite of the low precision of binomial patterns, Contrast TPM achieves the highest precision and recall rates in trading patterns.

To illustrate the effectiveness of the frequent conditions, a series of vibrant versions of Contrast TPM are evaluated with the coverage metric. The dominant ratio is an another important metric for evaluating the efficiency of the algorithms, as it means the proportion of TPs to binomial patterns. As shown in Table 7, we display the performances of these six algorithms in mining binomial patterns and TPs to demonstrate the necessity of considering them into mining TPs. Without the co-occurrence conditions (i.e., C_1, C_2, C_3), the algorithms, on average, achieve higher precision values comparing with others. Particularly, omitting C_1, the algorithm gains the second highest precision. The importance of order-preserving condition is confirmed by omitting C_4, in which the algorithm obtains the second highest recalls values both on binomial patterns and TPs. The best precision and dominant ratio of Contrast TPM demonstrating the rationality of our contrast frequent conditions.

4.5.2. Efficiency in fraud detection

The trading sequences of a pyramid scheme members imply obvious periodic trading action and contain specific trading amounts. In other words, a trading sequence that with high coverage of TPs is more likely generated by a pyramid scheme account. Therefore, the TPs obtained from Contrast TPM can be used as classification features for identifying the fraud bank accounts from normal accounts. To guarantee the credibility of our results, this section starts with a discussion about the generality of our data set.

Performance of methods with different l and k on testing data

The trading behaviors of a bank account can be extracted into three categories of features: (1) transaction statistical features, e.g., transaction amount, transaction frequency; (2) network behavioral features, e.g., number of the counterparty; (3) periodic behavioral features, e.g., the monthly ratio of total income to expenditure.

The differences and similarities of the pyramid scheme accounts and normal accounts on these three kinds of features have been fully discussed in our previous study [43]. The conclusion is drawn that there is no statistically significant difference between positive and negative trading sequences. Take the periodic behavior feature as an instance, 13 monthly periodic features are constructed, and then conduct null hypothesis significance testing is conducted for evaluating the differences among the two sample sets. Given significance level $\alpha = 0.05$, the result shows only one of the 13 p-values is bigger than α. The experiments demonstrate that our data is a general data set, and there is no obvious difference between the positive and negative trading sequences.

2. Detecting pyramid scheme members

The trading sequence of a normal account rarely contains binomial patterns, in contrast, that of a pyramid scheme member is more likely in containing the binomial patterns. Therefore, the

Table 8
Performances of different classifiers in using the binomimal patterns as features.

Algorithm	Accuracy	Recall	Precision	F_1 Score
Decision Tree	0.777	0.872	0.823	0.847
Random Forest	0.787	0.893	0.837	0.864
Extra Tree	0.806	0.888	0.838	0.862
Baseline [43]	*0.934*	*0.7783*	*0.7893*	*0.7838*

binomial patterns can be considered as attributes for identifying pyramid scheme accounts.

Given the binomial patterns (in the volume of 51978) are obtained from Contrast TPM, the one-hot encoding method is adopted to generate the feature vector for each account. Afterward, three classifiers are selected in testing the performance of the obtained patterns, which are Decision Tree, Random Forest, and Extra Tree respectively. The baseline is the best result of identifying the pyramid scheme members by using the features generated in [43]. Each classifier conducts 10-fold cross-validation and records the average results on testing data set in Table 8. Taking binomial patterns as the classification features, all of the three classifiers perform better than the baseline in Recall, Precision, and F_1 Score.

In summary, the classification performance of TPs demonstrates that Contrast TPM works well in the fraud detection task.

5. Conclusion

In this paper, we investigate the possibility of discover knowledge for quantitative research about pyramid schemes from financial time series data. We propose a sequential pattern mining framework for identifying trading patterns of pyramid schemes, whose result can be used for subsequent detection tasks. In our framework, contrast analysis is employed in determining the frequency conditions to reduce the reliance on user-defined support thresholds. The infrequent items is de-noised by LoRSD algorithm without destroying the structures of the original trading sequences. Contrast TPM conducts trading pattern mining on the noise-free trading sequences by using contrast analysis. The contrast frequent one-itemset is found out firstly, based on which trading patterns are discovered under the conditions of co-occurrence and order-preserving. The extensive experiments on a real-world data set demonstrate the performance of LoRSD and Contrast TPM in sequence de-noising and trading patterns mining. The results indicate that the inconspicuous activities of pyramid schemes are predictable with the help of sequential pattern mining techniques.

CRediT authorship contribution statement

Fang Lv: Conceptualization, Methodology, Writing – original draft. **Wei Wang:** Investigation, Writing – review & editing. **Linxuan Han:** Software. **Di Wang:** Data Curation, Formal analysis. **Yulong Pei:** Metrology. **Junheng Huang:** Validation, Supervision. **Bailing Wang:** Resources, Data curation, Funding acquisition. **Mykola Pechenizkiy:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work of this paper is supported by the National Key R&D Program of China (2021YFB2012400); the Fundamental Research Funds for the Central Universities, China (Grant No. HIT.NSRIF. 2020098).

References

[1] P. Feng, D. Sun, Z. Gong, A case study of pyramid scheme finance flow network based on social network analysis, Sustainability 11 (16) (2019) 4370.
[2] J. Moisander, C. Groß, K. Eräranta, Mechanisms of biopower and neoliberal governmentality in precarious work: Mobilizing the dependent self-employed as independent business owners, Hum. Relat. 71 (3) (2018) 375–398.
[3] S. Bosley, K.K. McKeage, Multilevel marketing diffusion and the risk of pyramid scheme activity: The case of fortune hi-tech marketing in montana, J. Public Policy Mark. 34 (1) (2015) 84–102.
[4] L. Schiffauer, Dangerous speculation: The appeal of pyramid schemes in rural Siberia, Focaal 2018 (81) (2018) 58–71.
[5] J. Xiong, A method of mining key accounts from internet pyramid selling data, Teh. Vjesn. 26 (3) (2019) 728–735.
[6] F. Lv, W. Wang, Y. Wei, Y. Sun, J. Huang, B. Wang, Detecting fraudulent bank account based on convolutional neural network with heterogeneous data, Math. Probl. Eng. 2019 (1) (2019) 1–11.
[7] F. Lv, J. Huang, W. Wang, Y. Wei, Y. Sun, B. Wang, A two-route CNN model for bank account classification with heterogeneous data, PLoS One 14 (8) (2019) 1–22.
[8] Y. Wu, Y. Tong, X. Zhu, X. Wu, NOSEP: Nonoverlapping sequence pattern mining with gap constraints, IEEE Trans. Cybern. 48 (10) (2017) 2809–2822.
[9] Z. He, S. Zhang, J. Wu, Significance-based discriminative sequential pattern mining, Expert Syst. Appl. 122 (2019) 54–64.
[10] P. Fournier-Viger, Z. Li, J.C.-W. Lin, R.U. Kiran, H. Fujita, Efficient algorithms to identify periodic patterns in multiple sequences, Inform. Sci. 489 (2019) 205–226.
[11] K.-Y. Huang, C.-H. Chang, Efficient mining of frequent episodes from complex sequences, Inf. Syst. 33 (1) (2008) 96–114.
[12] J.-S. Yeh, S.-C. Lin, A new data structure for asynchronous periodic pattern mining, in: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, in: ICUIMC '09, Association for Computing Machinery, New York, NY, USA, 2009, pp. 426–431.
[13] T. Dinh, V.-N. Huynh, B. Le, Mining periodic high utility sequential patterns, in: Asian Conference on Intelligent Information and Database Systems, Springer International Publishing, Springer, Manhattan, New York, 2017, pp. 545–555.
[14] T.H. Duong, D. Janos, V.D. Thi, N.T. Thang, T.T. Anh, An algorithm for mining high utility sequential patterns with time interval, Cybern. Inf. Technol. 19 (4) (2019) 3–16.
[15] P. Fournier-Viger, P. Yang, Z. Li, J.C.-W. Lin, R.U. Kiran, Discovering rare correlated periodic patterns in multiple sequences, Data Knowl. Eng. 126 (2020) 101733.
[16] Y.S. Koh, S.D. Ravana, Unsupervised rare pattern mining: a survey, ACM Trans. Knowl. Discov. Data (TKDD) 10 (4) (2016) 1–29.
[17] P. Fournier-Viger, P. Yang, J.C.-W. Lin, R.U. Kiran, Discovering stable periodic-frequent patterns in transactional data, in: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer-Verlag, Springer, Berlin, Heidelberg, 2019, pp. 230–244.
[18] H. Mannila, H. Toivonen, Verkamo, A. Inkeri, Discovery of frequent episodes in event sequences, Data Min. Knowl. Discov. 1 (3) (1997) 259–289.
[19] M. Wang, Y. Wu, M. Tsai, Exploiting frequent episodes in weighted suffix tree to improve intrusion detection system, in: 22nd International Conference on Advanced Information Networking and Applications - Workshops, Aina Workshops 2008, IEEE, Japan, Okinawa, 2008, pp. 1246–1252.
[20] A. Ng, A.W.-c. Fu, Mining frequent episodes for relating financial events and stock trends, in: Advances in Knowledge Discovery and Data Mining, Springer, Berlin, Heidelberg, 2003, pp. 27–39.
[21] A.A. Ramaki, M. Amini, R.E. Atani, RTECA: Real time episode correlation algorithm for multi-step attack scenarios detection, Comput. Secur. 49 (2015) 206–219.
[22] H. Ohtani, T. Kida, T. Uno, H. Arimura, Efficient serial episode mining with minimal occurrences, in: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, in: ICUIMC '09, Association for Computing Machinery, New York, NY, USA, 2009, pp. 457–464.
[23] H. Zhu, P. Wang, X. He, Y. Li, W. Wang, B. Shi, Efficient episode mining with minimal and non-overlapping occurrences, in: 2010 IEEE International Conference on Data Mining, 2010, pp. 1211–1216.
[24] A. Achar, I. A., P. Sastry, Pattern-growth based frequent serial episode discovery, Data Knowl. Eng. 87 (2013) 91–108.

[25] P. Fournier-Viger, P. Yang, J.C.-W. Lin, U. Yun, HUE-span: Fast high utility episode mining, in: J. Li, S. Wang, S. Qin, X. Li, S. Wang (Eds.), Advanced Data Mining and Applications, Springer International Publishing, Cham, 2019, pp. 169–184.
[26] G. Dong, J. Li, Efficient mining of emerging patterns: Discovering trends and differences, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: KDD '99, Association for Computing Machinery, New York, NY, USA, 1999, pp. 43–52.
[27] Y. Fan, Y. Ye, L. Chen, Malicious sequential pattern mining for automatic malware detection, Expert Syst. Appl. 52 (2016) 16–25.
[28] D. Fradkin, F. Mörchen, Mining sequential patterns for classification, Knowl. Inf. Syst. 45 (3) (2015) 731–749.
[29] Z. Zheng, W. Wei, C. Liu, W. Cao, L. Cao, M. Bhatia, An effective contrast sequential pattern mining approach to taxpayer behavior analysis, World Wide Web 19 (4) (2016) 633–651.
[30] Z. He, S. Zhang, F. Gu, J. Wu, Mining conditional discriminative sequential patterns, Inform. Sci. 478 (2019) 524–539.
[31] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M. Hsu, Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth, in: Proceedings of the 17th International Conference on Data Engineering, IEEE, Washington, DC, USA, 2001, pp. 215–224.
[32] Y. Kameya, An exhaustive covering approach to parameter-free mining of non-redundant discriminative itemsets, in: Big Data Analytics and Knowledge Discovery, Springer International Publishing, Cham, 2016, pp. 143–159.
[33] V. Chaoji, A. Hoonlor, B.K. Szymanski, Recursive data mining for role identification, in: Proceedings of the 5th International Conference on Soft Computing As Transdisciplinary Science and Technology, Association for Computing Machinery, New York, NY, United States, 2008, pp. 218–225.
[34] Y. Abboud, A. Brun, A. Boyer, C3Ro: An efficient mining algorithm of extended-closed contiguous robust sequential patterns in noisy data, Expert Syst. Appl. 131 (2019) 172–189.
[35] T.-Y. Wu, J.C.-W. Lin, U. Yun, C.-H. Chen, G. Srivastava, X. Lv, An efficient algorithm for fuzzy frequent itemset mining, J. Intell. Fuzzy Systems 38 (5) (2020) 5787–5797.
[36] E. Nissan, An overview of data mining for combating crime, Appl. Artif. Intell. 26 (8) (2012) 760–786.
[37] X. Ji, J. Bailey, G. Dong, Mining minimal distinguishing subsequence patterns with gap constraints, Knowl. Inf. Syst. 11 (3) (2007) 259–286.
[38] B. Ding, D. Lo, J. Han, S.-C. Khoo, Efficient mining of closed repetitive gapped subsequences from a sequence database, in: 2009 IEEE 25th International Conference on Data Engineering, IEEE, Piscataway,NJ, 2009, pp. 1024–1035.
[39] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M. Hsu, Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth, in: Proceedings of the 17th International Conference on Data Engineering, IEEE, Washington, DC, USA, 2001, pp. 215–224.
[40] M. Akra, L. Bazzi, On the solution of linear recurrence equations, Comput. Optim. Appl. 10 (2) (1998) 195–210.
[41] W. Wang, J. Tian, F. Lv, G. Xin, Y. Ma, B. Wang, Mining frequent pyramid patterns from time series transaction data with custom constraints, Comput. Secur. 100 (2021) 1–15.
[42] D.M. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, J. Mach. Learn. Technol. 2 (1) (2011) 37–63.
[43] F. Lv, J. Huang, W. Wang, G. Xin, B. Wang, Detecting pyramid scheme accounts with time series financial transactions, in: IEEE Third International Conference on Data Science in Cyberspace, 2018, pp. 722–728.

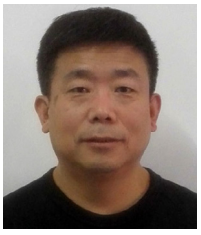
Fang Lv (Lyu) is currently a Ph.D. candidate in the school of Computer Science of Technology, Harbin Institute of Technology, China. She received her Master degree in the School of Computer Science and Technology from Harbin Institute of technology, in 2015. Her research interests include financial security, data mining.

Wei Wang is an assistant Professor in the school of Computer Science of Technology, Harbin Institute of Technology, China. She received her Ph.D. degree in the School of Computer Science and Technology from Harbin Institute of technology, in 2015. Her main interests include data mining, machine learning and nature language process.

Linxuan Han is currently a bachelor in the school of Computer Science and Technology, Harbin Institute of Technology, China. His research interests includes data mining and pattern recognition.

Di Wang is currently a bachelor in the school of Computer Science and Technology, Harbin Institute of Technology, China. His research interests includes Ant algorithm analysis and machine learning.

Yulong Pei is an assistant professor with Department of Mathematics and Computer Science, Eindhoven University of Technology (TU/e). He received his Ph.D. in Computer Science from TU/e in February 2020. His research interests cover graph mining, network embedding, and text mining. He has published several papers in top conferences and journals. He has served as the PC member of top-tier conferences and the regular reviewer for prestigious journals.



Junheng Huang is an Associate Professor in the school of Computer Science and Technology, Harbin Institute of Technology, China. He received his Master degree from the School of Mathematics and Statistics, Lanzhou university, in 1990. His main research interests include financial security, data mining and social network analysis.

Bailing Wang is a professor in the school of Computer Science of Technology, Harbin Institute of Technology, China. He received his Ph.D. degree from the School of Computer Science and Technology from Harbin Institute of technology, in 2006. His main research interests include financial security, information security and cyber networks.

Mykola Pechenizkiy is full professor, chair of Data Mining at the Department of Mathematics and Computer Science, TU Eindhoven. He received his PhD from University of Jyvaskyla, Finland in 2005. His main expertise is in predictive analytics on data evolving over time. He studies foundations of robustness, safety, trust, reliability, scalability, interpretability and explainability of AI. He collaborates with industry on developing novel techniques for informed, accountable and transparent predictive and prescriptive analytics.