CubeFlow: Money Laundering Detection with Coupled Tensors

Xiaobing Sun[1,*], Jiabao Zhang[1,2,*] Qiming Zhao[1,*], Shenghua Liu[1,2,*(✉)], Jinglei Chen[3], Ruoyu Zhuang[3], Huawei Shen[1,2], Xueqi Cheng[1,2]

[1] CAS Key Laboratory of Network Data Science and Technology, Insititute of Computing Technology, Chinese Academy of Sciences, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] China Construction Bank Fintech, China

xiaobingsun1999@gmail.com, zhangjiabao18@mails.ucas.edu.cn, qmzhao@cqu.edu.cn, liushenghua@ict.ac.cn, jl.chen.ray@gmail.com, zhuangruoyu@hotmail.com, {shenhuawei,cxq}@ict.ac.cn

Abstract. Money laundering (ML) is the behavior to conceal the source of money achieved by illegitimate activities, and always be a fast process involving frequent and chained transactions. How can we detect ML and fraudulent activity in large scale attributed transaction data (i.e. tensors)? Most existing methods detect dense blocks in a graph or a tensor, which do not consider the fact that money are frequently transferred through middle accounts. CubeFlow proposed in this paper is a scalable, flow-based approach to spot fraud from a mass of transactions by modeling them as two coupled tensors and applying a novel multi-attribute metric which can reveal the transfer chains accurately. Extensive experiments show CubeFlow outperforms state-of-the-art baselines in ML behavior detection in both synthetic and real data.

1 Introduction

Given a large amount of real-world transferring records, including a pair of accounts, some transaction attributes (e.g. time, types), and volume of money, how can we detect money laundering (ML) accurately in a scalable way? One of common ML processes disperses dirty money into different *source* accounts, transfers them through many *middle* accounts to *destination* accounts for gathering in a fast way. Thus the key problem for ML detection are:

Informal Problem 1. ***Given*** *a large amount of candidates of source, middle, and destination accounts, and the transferring records, which can be formalized as two coupled tensors with entries of (source candidates, middle candidates,*

*Xiaobing Sun, Jiabao Zhang, and Qiming Zhao contribute equally as the first authors. Shenghua Liu is the corresponding author.The work was done when Xiaobing Sun and Qiming Zhao were visiting students at ICT CAS, who are separately from NanKai University and Chongqing University.

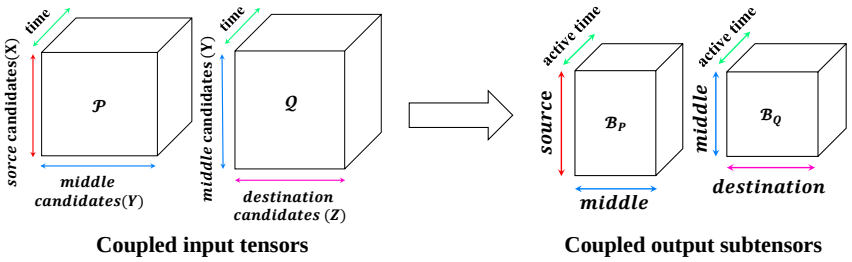


Fig. 1. An example of ML detection. Two coupled input tensors indicate a money flow from X to Y to Z. Modes X, Y and Z denote the candidates of source, middle and destination accounts. The purpose of detection is to find two dense coupled blocks in original coupled tensors, i.e., catching fraudsters involving in two-step ML activities.

time, ⋯), and (middle candidates, destination candidates, time, ⋯), how ***to find*** *the accounts in such a ML process accurately and efficiently.*

Fig. 1 shows an example of ML detection with two coupled tensors indicating a flow from source to middle to destination accounts. Those candidates can be pre-selected by existing feature-based models or in an empirical way. For example, in a bank, we can let source candidates simply be external accounts with more money transferring into the bank than out of the bank, destination candidates be the opposite ones, and middle candidates be the inner accounts.

Most existing dense subtensor detection methods [6,18,19] have been used for tensor fraud detection, but only can deal with one independent tensor. Therefore, they exploit exactly single-step transfers, but do not account for "transfer chains". Such methods based on graph's density [5,14,17], have the same problem and even not be able to leverage multi-attributes. Although, FlowScope [12] designed for dense and multi-step flow, it fails to take into account some important properties (e.g. time) because of the limits by the graph.

Therefore, we propose CubeFlow, a ML detection method with coupled tensors. CubeFlow not only considers the flow of funds (from sources, through middle accounts, to destinations), but also can combine some attributes, such as transferring time to model fraudsters' highly frequent transfers. We define a novel multi-attribute metric for fraudulent transferring flows of ML. CubeFlow considers the suspicious in-and-out balance for middle accounts within short time intervals and detects the chain of fraudulent transfers accurately. The experiments on real-world datasets show that CubeFlow detects various adversarial injections and real ML fraudsters both with high accuracy and robustness.

In summary, the main advantages of our work are:

• **Multi-attribute metric for money-laundering flow:** We propose a novel multi-attribute metric for detecting dense transferring flows in coupled tensors, which measures the anomalousness of typical two-step laundering with suspicious in-and-out balance for middle accounts within many short time intervals.

• **Effectiveness and robustness:** CubeFlow outperforms baselines under various injection densities by descending the amount of money or ascending # of accounts on real-world datasets. And CubeFlow shows its robustness in different proportions of accounts with better accuracy.

• **Scalability:** CubeFlow is scalable, with near-linear time complexity in the number of transferring records.

Our code and processed data are publicly available for reproducibility [4].

2 Related Work

Our work pulls from two main different fields of research: (i) domain-specific ML detection methods; and (ii) general anomaly detection methods in graphs and tensors.

Money laundering detection. The most classical approaches for Anti-ML are those of rule based classification relying heavily on expertise. Khan et al. [9] used Bayesian network designed with guidance of the rules to assign risk scores to transactions. The system proposed by [10] monitored ongoing transactions and assessed their degree of anomaly. However, rule based algorithms are easy to be evaded by fraudsters. To involve more attributes and handle the high-dimensional data, machine learning models such as SVM [21], decision trees [22] and neural networks [15] are applied, while these methods are focused on isolated transaction level. Stavarache et al. [20] proposed a deep learning based method trained for Anti-ML tasks using customer-to-customer relations. However, these algorithms detect the ML activities in supervised or semi-supervised manners, suffering from imbalanced class and lack of adaptability and interpretability.

General-purpose anomaly detection in graphs and tensors. Graphs (i.e. tensors) provide a powerful mechanism to capture interrelated associations between data objects [1], and there have been many graph-based techniques developed for discovering structural anomalies. SpokEn [17] studied patterns in eigenvectors, and was applied for anomaly detection in [8] later. CatchSync [7] exploited two of the tell-tale signs created by fraudsters. And many existing methods rely on graph (i.e. tensor)'s density, e.g., Fraudar [5] proposed a suspiciousness measure on the density, HoloScope [13, 14] considered temporal spikes and hyperbolic topology and SpecGreedy [3] proposed a unified framework based on the graph spectral properties. D-Cube [19], M-Zoom [18] and CrossSpot [6] adopted greedy approximation algorithms to detect dense subtensors, while CP Decomposition (CPD) [11] focused on tensor decomposition methods. However, these methods are designed for general-purpose anomaly detection tasks, which not take the flow across multiple nodes into account.

3 PROBLEM FORMULATION

In general money laundering (ML) scenario, fraudsters transfer money from source accounts to destination accounts through several middle accounts in order

[4] https://github.com/BGT-M/spartan2-tutorials/blob/master/CubeFlow.ipynb

Table 1. Notations and symbols

Symbol	Definition
$\boldsymbol{\mathcal{P}}(X, Y, A_3, \ldots, A_N, V)$	Relation representing the tensor which stands money trans from X to Y
$\boldsymbol{\mathcal{Q}}(Y, Z, A_3, \ldots, A_N, V)$	Relation representing the tensor which stands money trans from Y to Z
N	Number of mode attributes in $\boldsymbol{\mathcal{P}}$ (or $\boldsymbol{\mathcal{Q}}$)
A_n	n-th mode attribute name in $\boldsymbol{\mathcal{P}}$ (or $\boldsymbol{\mathcal{Q}}$)
V	measure attribute (e.g. money) in $\boldsymbol{\mathcal{P}}$ (or $\boldsymbol{\mathcal{Q}}$)
$\boldsymbol{\mathcal{B}}_{\mathbf{P}}$ (or $\boldsymbol{\mathcal{B}}_{\mathbf{Q}}$)	a block(i.e. subtensor) in $\boldsymbol{\mathcal{P}}$ (or $\boldsymbol{\mathcal{Q}}$)
$\mathbf{P}_x$, $\mathbf{P}_y$, $\mathbf{Q}_z$,$\mathbf{P}_{a_n}$	set of distinct values of X, Y, Z, A_n in $\boldsymbol{\mathcal{P}}$ (or $\boldsymbol{\mathcal{Q}}$)
$\mathbf{B}_x$,$\mathbf{B}_y$, $\mathbf{B}_z$, $\mathbf{B}_{a_n}$	set of distinct values of X, Y, Z, A_n in $\boldsymbol{\mathcal{B}}_{\mathbf{P}}$ (or $\boldsymbol{\mathcal{B}}_{\mathbf{Q}}$)
$M_{x,y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}})$	attribute-value mass of $(x, y, a_3, \ldots, a_N)$ in $(X, Y, A_3, \ldots, A_N)$
$M_{y,z,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{Q}})$	attribute-value mass of $(y, z, a_3, \ldots, a_N)$ in $(Y, Z, A_3, \ldots, A_N)$
ω_i	Weighted assined to a node in priority tree
$g(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}})$	Metric of ML anomalousness
$[3, N]$	$\{3, 4, \ldots, N\}$

to cover up the true source of funds. Here, we summarize three typical characteristics of money laundering:

Density: In ML activities, a high volume of funds needs to be transferred from source to destination accounts with limited number of middle accounts. Due to the risk of detection, fraudsters tend to use shorter time and fewer trading channels in the process of ML which will create a high-volume and dense subtensor of transfers.

Zero out middle accounts: The role of middle accounts can be regarded as a bridge in ML: only a small amount of balance will be kept in these accounts for the sake of camouflage, most of the received money will be transferred out. This is because the less balances retained, the less losses will be incurred if these accounts are detected or frozen.

Fast in and Fast out: To reduce banks' attention, "dirty money" is always divided into multiple parts and transferred through the middle accounts one by one. The "transfer", which means a part of fund is transferred in and out of a middle account, is usually done within a very short time interval. This is because the sooner the transfer is done, the more benefits fraudsters will get.

Algorithms which focus on individual transfers, e.g. feature-based approaches, can be easily evaded by adversaries by keeping each individual transfer looks normal. Instead, our goal is to detect dense blocks in tensors composed of source, middle, destination accounts and other multi-attributes, as follows:

Informal Problem 2 (ML detection with coupling tensor)**.** ***Given*** *two money transfer tensors* $\boldsymbol{\mathcal{P}}(X, Y, A_3, \ldots, A_N, V)$ *and* $\boldsymbol{\mathcal{Q}}(Y, Z, A_3, \ldots, A_N, V)$*, with attributes of source, middle and destination candidates as* X, Y, Z*, other coupling attributes (e.g. time) as* A_n*, and a nonnegative measure attribute (e.g. volume of money) as* V*.*

Find: *two dense blocks (i.e. subtensor) of* $\boldsymbol{\mathcal{P}}$ *and* $\boldsymbol{\mathcal{Q}}$*.*

Such that:
- it maximizes density.
- for each middle account, the money transfers satisfy zero-out and fast-in-and-fast-out characteristics.

Symbols used in the paper are listed in Table 1. As common in other literature, we denote tensors and modes of tensors by boldface calligraphic letters(e.g. $\boldsymbol{\mathcal{P}}$) and capital letters(e.g. X, Y, Z, A_n) individually. For the possible values of different modes, boldface uppercase letters (e.g. $\mathbf{P}_x, \mathbf{P}_y, \mathbf{Q}_z, \mathbf{P}_{a_n}$) are used in this paper. Since $\boldsymbol{\mathcal{P}}(X, Y, A_3, \ldots, A_N, V)$ and $\boldsymbol{\mathcal{Q}}(Y, Z, A_3, \ldots, A_N, V)$ are coupled tensors sharing the same sets of modes Y, A_n, we have $\mathbf{P}_y = \mathbf{Q}_y$ and $\mathbf{P}_{a_n} = \mathbf{Q}_{a_n}$. Our targets, the dense blocks (i.e. subtensor) of $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{Q}}$, are represented by $\boldsymbol{\mathcal{B}}_{\mathbf{P}}$ and $\boldsymbol{\mathcal{B}}_{\mathbf{Q}}$. Similarly, the mode's possible values in these blocks are written as $\mathbf{B}_x, \mathbf{B}_y, \mathbf{B}_z, \mathbf{B}_{a_n}$. An entry $(x, y, a_3, \ldots, a_N)$ indicates that account x transfers money to account y when other modes are equal to $a_3, \ldots, a_N$ (e.g. during a_3 time-bin), and $M_{x,y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}})$ is the total amount of money on the subtensor.

4 Proposed Method

4.1 Proposed Metric

First, we give the concept of fiber: *A fiber of a tensor $\boldsymbol{\mathcal{P}}$ is a vector obtained by fixing all but one $\boldsymbol{\mathcal{P}}$'s indices.* For example, in ML process with A_3 representing transaction timestamp, total money transferred from source accounts X into a middle account y at time-bin a_3 is the mass of fiber $\boldsymbol{\mathcal{P}}(X, y, a_3, V)$ which can be denoted by $M_{:,y,a_3}(\boldsymbol{\mathcal{B}}_{\mathbf{P}})$, while total money out of the middle account can be denoted by $M_{y,:,a_3}(\boldsymbol{\mathcal{B}}_{\mathbf{Q}})$.

In general form, we can define the minimum and maximum value between total amount of money transferred into and out of a middle account $y \in \mathbf{B}_y$ with other attributes equal to $a_3 \in \mathbf{B}_{a_3}, \ldots, a_N \in \mathbf{B}_{a_N}$:

$$f_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}}) = min\{M_{:,y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}), M_{y,:,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{Q}})\} \tag{1}$$

$$q_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}}) = max\{M_{:,y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}), M_{y,:,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{Q}})\} \tag{2}$$

Then we can define the difference between the maximum and minimum value:

$$r_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}}) = q_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}}) - f_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_{\mathbf{P}}, \boldsymbol{\mathcal{B}}_{\mathbf{Q}}) \tag{3}$$

Next, our ML metric is defined as follows for spotting multi-attribute money-laundering flow:

Definition 1. *(Anomalousness of coupled blocks of ML) The anomalousness of a flow from a set of nodes $\mathbf{B}_x$, through the inner accounts $\mathbf{B}_y$, to another subset $\mathbf{B}_z$, where other attribute values are $a_n \in \mathbf{B}_{a_n}$:*

$$g(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}) = \frac{\sum_{y\in\mathbf{B}_y, a_i\in\mathbf{B}_{a_i}} \big((1-\alpha) f_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}) - \alpha \cdot r_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q})\big)}{\sum_{i=3}^{N}(|\mathbf{B}_{a_i}|) + |\mathbf{B}_x| + |\mathbf{B}_y| + |\mathbf{B}_z|}$$
$$= \frac{\sum_{y\in\mathbf{B}_y, a_i\in\mathbf{B}_{a_i}} \big(f_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}) - \alpha q_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q})\big)}{\sum_{i=3}^{N}(|\mathbf{B}_{a_i}|) + |\mathbf{B}_x| + |\mathbf{B}_y| + |\mathbf{B}_z|} \tag{4}$$

Intuitively, $f_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P})$ is the maximum possible flow that could go through middle account $y \in \mathbf{B}_y$ when other attributes are $a_n \in \mathbf{B}_{a_n}$. $r_{y,a_3,\ldots,a_N}(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q})$ is the absolute value of "remaining money" in account y after transfer, i.e., retention or deficit, which can be regarded as a penalty for ML, since fraudsters prefer to keep small account balance at any situations. When we set A_3 as time dimension, we consider the "remaining money" in each time bin which will catch the trait of fast in and fast out during ML. We define α as the coefficient of imbalance cost rate in the range of 0 to 1.

4.2 Proposed Algorithm: CubeFlow

We use a near-greedy algorithm CubeFlow, to find two dense blocks $\boldsymbol{\mathcal{B}}_\mathbf{P}$ and $\boldsymbol{\mathcal{B}}_\mathbf{Q}$ maximizing the objective $g(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q})$ in (4).

To develop an efficient algorithm for our metric, we unfold the tensor $\boldsymbol{\mathcal{P}}$ on mode-X and $\boldsymbol{\mathcal{Q}}$ on mode-Z. For example, a tensor unfolding of $\boldsymbol{\mathcal{P}} \in \mathbb{R}^{|\mathbf{B}_x|\times|\mathbf{B}_y|\times|\mathbf{B}_{a_3}|}$ on mode-X will produce a $|\mathbf{B}_x| \times (|\mathbf{B}_y| \times |\mathbf{B}_{a_3}|)$ matrix.

For clarity, we define the index set $\mathbf{I}$, whose size equals to the number of columns of matrix:

$$\mathbf{I} = \mathbf{B}_y \bowtie \mathbf{B}_{a_3} \bowtie \ldots \bowtie \mathbf{B}_{a_N} \tag{5}$$

where $\bowtie$ denotes Cartesian product. Therefore, the denominator of (4) can be approximated by $|\mathbf{B}_x| + |\mathbf{I}| + |\mathbf{B}_z|$.

First, we build a priority tree for entries in $\boldsymbol{\mathcal{B}}_\mathbf{P}$ and $\boldsymbol{\mathcal{B}}_\mathbf{Q}$. The weight (ie. priority) assigned to index i is defined as:

$$\omega_i(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}) = \begin{cases} f_i(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}) - \alpha q_i(\boldsymbol{\mathcal{B}}_\mathbf{P}, \boldsymbol{\mathcal{B}}_\mathbf{Q}), & \text{if } i \in \mathbf{I} \\ M_{i,:,:,\ldots,:}(\boldsymbol{\mathcal{B}}_\mathbf{P}), & \text{if } i \in \mathbf{B}_x \\ M_{:,i,:,\ldots,:}(\boldsymbol{\mathcal{B}}_\mathbf{Q}), & \text{if } i \in \mathbf{B}_z \end{cases} \tag{6}$$

The algorithm is described in Alg 1. After building the priority tree, we perform the near greedy optimization: block $\boldsymbol{\mathcal{B}}_\mathbf{P}$ and $\boldsymbol{\mathcal{B}}_\mathbf{Q}$ start with whole tensor $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{Q}}$. Let we denote $\mathbf{I} \cup \mathbf{B}_x \cup \mathbf{B}_z$ as $\mathbf{S}$. In every iteration, we remove the node v in $\mathbf{S}$ with minimum weight in the tree, approximately maximizing objective (4); and then we update the weight of all its neighbors. The iteration is repeated until one of node sets $\mathbf{B}_x, \mathbf{B}_z, \mathbf{I}$ is empty. Finally, two dense blocks $\hat{\boldsymbol{\mathcal{B}}}_\mathbf{P}, \hat{\boldsymbol{\mathcal{B}}}_\mathbf{Q}$ that we have seen with the largest value $g(\hat{\boldsymbol{\mathcal{B}}}_\mathbf{P}, \hat{\boldsymbol{\mathcal{B}}}_\mathbf{Q})$ are returned.

Algorithm 1: CubeFlow

Input: relation $\mathcal{P}$, relation $\mathcal{Q}$
Output: dense block $\mathcal{B}_{\mathbf{P}}$, dense block $\mathcal{B}_{\mathbf{Q}}$

$\mathcal{B}_{\mathbf{P}} \leftarrow \mathcal{P}$, $\mathcal{B}_{\mathbf{Q}} \leftarrow \mathcal{Q}$;
$\mathbf{S} \leftarrow \mathbf{I} \cup \mathbf{B}_x \cup \mathbf{B}_z$;
$\omega_i \leftarrow$ calculate node weight as Eq. (6) ;
$T \leftarrow$ build priority tree for $\mathcal{B}_{\mathbf{P}}$ and $\mathcal{B}_{\mathbf{Q}}$ with $\omega_i(\mathcal{B}_{\mathbf{P}}, \mathcal{B}_{\mathbf{Q}})$;
while $\mathbf{B}_x, \mathbf{B}_z$ *and* $\mathbf{I}$ *is not empty* **do**
    $v \leftarrow$ find the minimum weighted node in T;
    $\mathbf{S} \leftarrow \mathbf{S} \backslash \{v\}$;
    update priorities in T for all neighbors of v;
    $g(\mathcal{B}_{\mathbf{P}}, \mathcal{B}_{\mathbf{Q}}) \leftarrow$ calculate as Eq. (4);
end
return $\hat{\mathcal{B}}_{\mathbf{P}}, \hat{\mathcal{B}}_{\mathbf{Q}}$ that maximizes $g(\mathcal{B}_{\mathbf{P}}, \mathcal{B}_{\mathbf{Q}})$ seen during the loop.

5 Experiments

We design experiments to answer the following questions:

- **Q1. Effectiveness:** How early and accurate does our method detect synthetic ML behavior comparing to the baselines?
- **Q2. Performance on real-world data:** How early and accurate does our CubeFlow detect real-world ML activity comparing to the baselines?
- **Q3. Performance on 4-mode tensor:** How accurate does CubeFlow compare to the baselines dealing with multi-mode data?
- **Q4. Scalability:** Does our method scale linearly with the number of edges?

Table 2. Dataset Description

Name	Volume	# Tuples
3-mode bank transfer record (from_acct, to_acct, time, money)		
CBank	$491295 \times 561699 \times 576$	$2.94M$
	$561699 \times 1370249 \times 576$	$2.60M$
CFD-3	$2030 \times 2351 \times 728$	$0.12M$
	$2351 \times 7001 \times 730$	$0.27M$
4-mode bank transfer record (from_acct, to_acct, time, k_symbol, money)		
CFD-4	$2030 \times 2351 \times 728 \times 7$	$0.12M$
	$2351 \times 7001 \times 730 \times 8$	$0.27M$

5.1 Experimental Setting

Machine: We ran all experiments on a machine with 2.7GHZ Intel Xeon E7-8837 CPUs and 512GB memory.

Data: Table 2 lists data used in our paper. CBank data is a real-world transferring data from an anonymous bank under an NDA agreement. Czech Financial Data (CFD) is an anonymous transferring data of Czech bank released for Discovery Challenge in [16]. We model CFD data as two 3-mode tensors consisting of entries (a_1, a_2, t, m), which means that account a_1 transfers the amount of money, m, to account a_2 at time t. Specifically, we divide the account whose money transferring into it is much larger than out of the account into X, on the contrary, into Z, and the rest into Y. Note that it does not mean that X, Y, Z have to be disjoint, while this preprocessing helps speed up our algorithm. We also model CFD data as two 4-mode tensors having an additional dimension *k_Symbol* (characterization of transaction, e.g., insurance payment and payment of statement). And we call two CFD data as CFD-3 and CFD-4 resp.

Implementations: We implement CubeFlow in Python, CP Decomposition(CPD) [11] in Matlab and run the open source code of D-Cube [19], M-Zoom [18]and CrossSpot [6]. We use the sparse tensor format for efficient space utility. Besides, the length of time bins of CBank and CFD are 20 minutes and 3 days respectively, and the value of α is 0.8 as default.

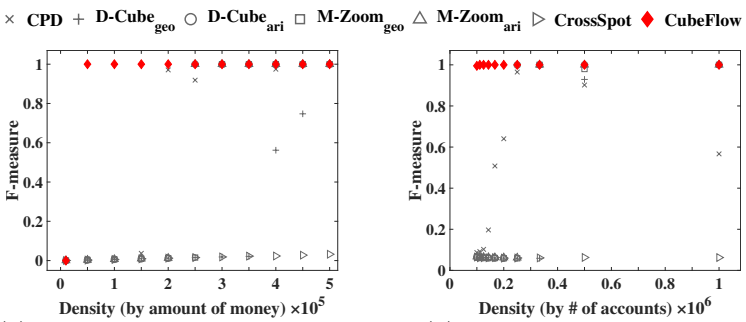
5.2 Q1.Effectiveness

To verify the effectiveness of CubeFlow, we inject ML activities as follows: fraudulent accounts are randomly chosen as the tripartite groups, denoted by B_x, B_y and B_z. The fraudulent edges between each group are randomly generated with probability p. We use Dirichlet distribution (the value of scaling parameter is 100) to generate the amount of money for each edge. And for each account in B_y, the amount of money received from B_x and that of transferred to B_z are almost the same. Actually, we can regard the remaining money of accounts in B_y as camouflage, with amount of money conforms to a random distribution ranging from 0 to 100000 (less than 1% of injected amount of money). To satisfy the trait "Fast in and Fast out", we randomly choose the time from one time bin for all edges connected with the same middle account.

The influence of the amount of money: In this experiment, the number of B_x, B_y, B_z is 5, 10 and 5 resp, and we increase the amount of injected money laundered step by step while fixing other conditions. As shown in Figure. 2(a), CubeFlow detects the ML behavior earliest and accurately, and the methods based on bipartite graph are unable to catch suspicious tripartite dense flows in the tensor.

The influence of the number of fraudulent accounts: Another possible case is that fraudsters may employ as many as people to launder money, making the ML behavior much harder to detect. In this experiment, we increase the number of fraudsters step by step at a fixed ratio (5 : 10 : 5) while keeping the amount of laundering money and other conditions unchanged. As Figure. 2(b) shown, our method achieves the best results.

Robustness with different injection ratios of accounts: To verify the robustness of our method, we randomly pick B_x, B_y and B_z under three ratios



(a) descending amount of money (b) ascending # of accounts

Fig. 2. CubeFlow outperforms baselines under different injected densities by descending amount of money (a) or ascending # of accounts (b) on CFD-3 data.

Table 3. Experimental results on CFD-3 data with different injection ratios of accounts

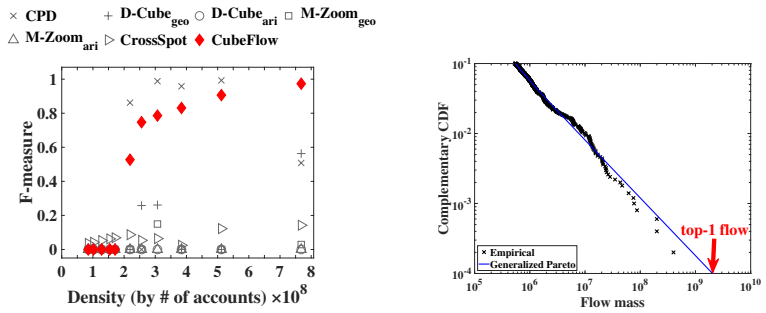
X:Y:Z	CubeFlow	D-Cube$_{geo}$	D-Cube$_{ari}$	CPD	CrossSpot	M-Zoom$_{geo}$	M-Zoom$_{ari}$
5:10:5	**0.940**	0.189	0.553	0.641	0.015	0.455	0.553
10:10:10	**0.940**	0.427	0.653	0.647	0.024	0.555	0.555
10:5:10	**0.970**	0.652	0.652	0.725	0.020	0.652	0.652

as presented in Table 3. The metric for comparison is **FAUC**: the areas under curve of F-measure as in Figure. 2. We normalize the density in horizontal axis to scale FAUC between 0 and 1, and higher FAUC indicates better performance. And we can see from Table 3, CubeFlow achieves far better performance than other baselines under all settings, indicating earlier and more accurate detection for more fraudulent accounts.

5.3 Q2. Performance on real-world data

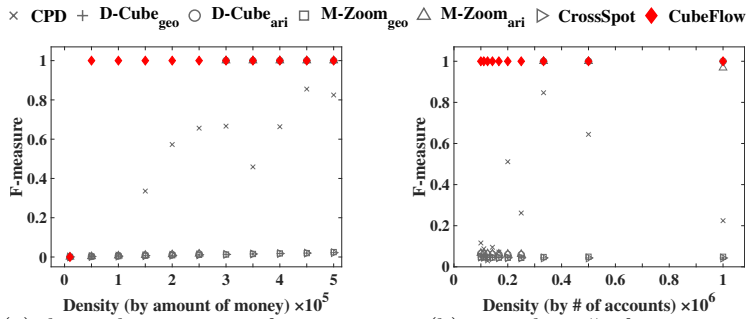
CBank data contains labeled ML activity: based on the X to Y to Z schema, the number of each type of accounts is 4, 12, 2. To test how accurately and early we can detect the fraudsters in CBank data, we first scale down the percentage of dirty money laundered from source accounts to destination accounts, then gradually increase the volume of money laundering linearly back to the actual value in the data. Figure. 3(a) shows that CubeFlow can the catch the ML behaviors earliest, exhibiting our method's utility in detecting real-world ML activities. Note that although CPD works well at some densities, it fluctuate greatly, indicating that CPD is not very suitable for ML behavior detection.

Flow Surprisingness estimation with extreme value theory: Inspired by [4], we use Generalized Pareto (GP) Distribution, a commonly used probability distribution within extreme value theory, to estimate the extreme **tail** of a distribution without making strong assumptions about the distribution it-



(a) CBank data with ground-truth (b) mass distribution of flows

Fig. 3. CubeFlow performs best in real CBank data. (a) CubeFlow detects earliest (less money being laundered) and accurately in ground-truth community. (b) The GP distribution closely fits mass distributions of real flows: Black crosses indicate the empirical mass distribution for flows with same size with top-1 flow detected by CubeFlow, in the form of its complementary CDF (i.e. CCDF).



(a) descending amount of money (b) ascending # of accounts

Fig. 4. CubeFlow outperforms the baselines under different adversarial densities by descending the amount of money (a) or ascending # of accounts (b) on CFD-4 data.

self. GP distributions exhibit heavy-tailed decay (i.e. power law tails), which can approximate the tails of almost any distribution, with error approaching zero [2].

Specifically, we estimate tail of GP distribution via sampling. Given a flow corresponding to two blocks, $\mathcal{B}_\mathbf{P}$, $\mathcal{B}_\mathbf{Q}$, with total mass $M(\mathcal{B}_\mathbf{P}) + M(\mathcal{B}_\mathbf{Q})$, we sample 5000 uniformly random flows from data with same size. For $\epsilon = 0.1$, we fit a GP distribution using maximum likelihood to the largest ϵN masses. The surprisingness of flow is the **CDF** of this GP distribution, evaluated at its mass. As shown in Figure. 3(b), masses of sampled flows follow a GP distribution and tail measure score (i.e. CDF) of top-1 flow detected by CubeFlow is very close to 1 (pointed by red arrow), indicating that this activity is quite extreme(i.e. rare) in CBank data.

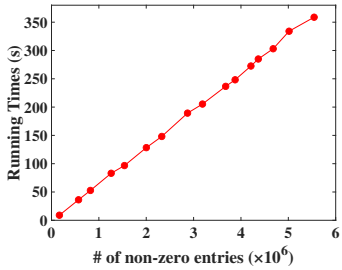


Fig. 5. CubeFlow scales near linearly

5.4 Q3. Performance on 4-mode tensor

To evaluate the performance of CubeFlow dealing with multi-mode data, we conducted similar experiments on CFD-4 data. Figure. 4(a)-4(b) show that our method takes significant advantages over the baselines as our method achieves excellent performance far earlier.

5.5 Q4. Scalability

Scalability: We demonstrate the linearly scalability with of CubeFlow by measuring how rapidly its update time increases as a tensor grows. As Figure. 5 shown, CubeFlow scales linearly with the size of non-zero entries.

6 Conclusion

In this paper, we propose a money laundering detection method, CubeFlow, which is a scalable, flow-based approach to spot the fraud in big attributed transaction tensors. We model the problem with two coupled tensors and propose a novel multi-attribute metric which can utilize different characteristics of money-laundering flow. Experiments based on different data have demonstrated the effectiveness and robustness of CubeFlow's utility as it outperforms state-of-the-art baselines. The source code is opened for reproducibility.

Acknowledgements. This paper is partially supported by the National Science Foundation of China under Grant No.91746301, 61772498, U1911401, 61872206, 61802370. This paper is also supported by the Strategic Priority Research Program of the Chinese Academy of Sciences, Grant No. XDA19020400 and 2020 Tencent Wechat Rhino-Bird Focused Research Program.

References

1. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data mining and knowledge discovery **29**(3) (2015)

2. Balkema, A.A., De Haan, L.: Residual life time at great age. Annals of Probability (1974)
3. Feng, W., Liu, S., Danai, K., Shen, H., Cheng, X.: Specgreedy: Unified dense subgraph detection. In: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD) (2020)
4. Hooi, B., Shin, K., Lamba, H., Faloutsos, C.: Telltail: Fast scoring and detection of dense subgraphs. In: AAAI (2020)
5. Hooi, B., Song, H.A., Beutel, A., Shah, N., Shin, K., Faloutsos, C.: Fraudar: Bounding graph fraud in the face of camouflage. In: SIGKDD. ACM (2016)
6. Jiang, M., Beutel, A., Cui, P., Hooi, B., Yang, S., Faloutsos, C.: A general suspiciousness metric for dense blocks in multimodal data. In: ICDM (2015)
7. Jiang, M., Cui, P., Beutel, A., Faloutsos, C., Yang, S.: Catchsync: catching synchronized behavior in large directed graphs. In: SIGKDD. ACM (2014)
8. Jiang, M., Cui, P., Beutel, A., Faloutsos, C., Yang, S.: Inferring strange behavior from connectivity pattern in social networks. In: PAKDD. Springer (2014)
9. Khan, N.S., Larik, A.S., Rajput, Q., Haider, S.: A bayesian approach for suspicious financial activity reporting. International Journal of Computers and Applications (2013)
10. Khanuja, H.K., Adane, D.S.: Forensic analysis for monitoring database transactions. In: International Symposium on Security in Computing and Communication. Springer (2014)
11. Kolda, T., Bader, B.: Tensor decompositions and applications. SIAM Review (2009)
12. Li, X., Liu, S., Li, Z., Han, X., Shi, C., Hooi, B., Huang, H., Cheng, X.: Flowscope: Spotting money laundering based on graphs. In: AAAI (2020)
13. Liu, S., Hooi, B., Faloutsos, C.: A contrast metric for fraud detection in rich graphs. IEEE Transactions on Knowledge and Data Engineering (2019)
14. Liu, S., Hooi, B., Faloutsos, C.: Holoscope: Topology-and-spike aware fraud detection. In: CIKM. ACM (2017)
15. Lv, L.T., Ji, N., Zhang, J.L.: A rbf neural network model for anti-money laundering. In: ICWAPR. IEEE (2008)
16. Lütkebohle, I.: Bworld robot control software. https://data.world/lpetrocelli/czech-financial-dataset-real-anonymized-transactions/, [Online; accessed 2-November-2018]
17. Prakash, B.A., Sridharan, A., Seshadri, M., Machiraju, S., Faloutsos, C.: Eigenspokes: Surprising patterns and scalable community chipping in large graphs. In: PAKDD. Springer (2010)
18. Shin, K., Hooi, B., Faloutsos, C.: M-zoom: Fast dense-block detection in tensors with quality guarantees. In: PKDD. Springer (2016)
19. Shin, K., Hooi, B., Kim, J., Faloutsos, C.: D-cube: Dense-block detection in terabyte-scale tensors. In: WSDM. ACM (2017)
20. Stavarache, L.L., Narbutis, D., Suzumura, T., Harishankar, R., Žaltauskas, A.: Exploring multi-banking customer-to-customer relations in aml context with poincar\'e embeddings. arXiv preprint arXiv:1912.07701 (2019)
21. Tang, J., Yin, J.: Developing an intelligent data discriminating system of anti-money laundering based on svm. In: ICMLC. IEEE (2005)
22. Wang, S.N., Yang, J.G.: A money laundering risk evaluation method based on decision tree. In: ICMLC. IEEE (2007)