Towards Learning to Discover Money Laundering Sub-network in Massive Transaction Network

Ziwei Chai[1, 2], Yang Yang[*1], Jiawang Dan[3], Sheng Tian[3], Changhua Meng[3], Weiqiang Wang[3], Yifei Sun[1]

[1] Zhejiang University, Hangzhou, China
[2] Beijing Huairou Laboratory, Beijing, China
[3] Ant Group, Hangzhou, China
{zwchai, yangya, yifeisun}@zju.edu.cn, {yancong.djw, tiansheng.ts, changhua.mch, weiqiang.wwq}@antgroup.com

Abstract

Anti-money laundering (AML) systems play a critical role in safeguarding global economy. As money laundering is considered as one of the top group crimes, there is a crucial need to discover money laundering sub-network behind a particular money laundering transaction for a robust AML system. However, existing rule-based methods for money laundering sub-network discovery is heavily based on domain knowledge and may lag behind the modus operandi of launderers. Therefore, in this work, we first address the money laundering sub-network discovery problem with a neural network based approach, and propose an AML framework AMAP equipped with an adaptive sub-network proposer. In particular, we design an adaptive sub-network proposer guided by a supervised contrastive loss to discriminate money laundering transactions from massive benign transactions. We conduct extensive experiments on real-word datasets in AliPay of Ant Group. The result demonstrates the effectiveness of our AMAP in both money laundering transaction detection and money laundering sub-network discovering. The learned framework which yields money laundering sub-network from massive transaction network leads to a more comprehensive risk coverage and a deeper insight to money laundering strategies.

Introduction

Money laundering (ML), aiming at giving illegally-gained proceeds (i.e."dirty money") a legitimate appearance (i.e. "clean"), has been one of the major threat for economy and society for decades . It's estimated that the amount of money laundered globally every year accounts for 2-5% of global GDP and is expected to continue to grow (Kute et al. 2021). Typically, the illegitimate funds to be laundered are transferred around, sometimes through numerous complex financial transactions, until the illegal source of the funds is disguised (Force 1999). Figure 1 shows a typical case of money laundering transaction sub-network (ML sub-network) observed in a real-world online payment system of *AliPay*[1]. Money laundering transaction chains may often be hidden

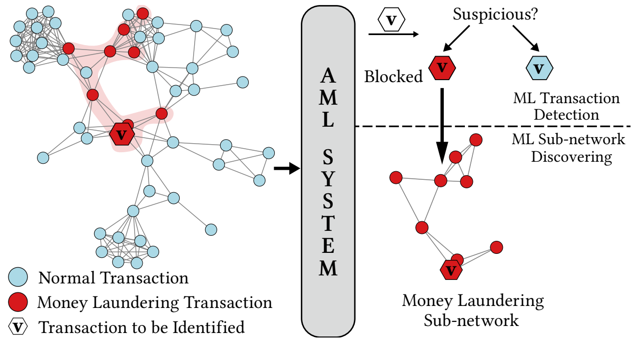
Normal Transaction
Money Laundering Transaction
Transaction to be Identified
AML SYSTEM
Suspicious?
Blocked
ML Transaction Detection
ML Sub-network Discovering
Money Laundering Sub-network

Figure 1: An illustration of real-world transaction network in Alipay. A robust AML system is expected to discover ML sub-network of a suspicious ML transaction for comprehensive risk coverage.

among benign transactions and difficult to be uncovered. According to financial institutes rules, the flow of money must be fully tracked. As a consequence, for a comprehensive risk coverage, a robust anti-money laundering (AML) system is expected not only to identify and block a certain suspicious ML transaction (*ML transaction detection*), but also to unveil the suspicious ML sub-network from massive transaction networks (*ML sub-network discovery*).

Many existing methods aiming to build AML systems have been extensively studied. They are mainly divided into two categories: 1). *Rule-based methods* have been very popular among the commercial institutions in the past few decades (Chen et al. 2018). The rule-based AML system is based on embedded rules which were developed by consultants and domain experts. The key issue of rule-based systems is that keeping the rules up-to-date all the time is extremely hard. 2). *Machine/Deep learning-based methods* learn models by exploring massive amounts of historical transaction data. For example, Kingdon (2004) proposes an extension of support vector machine (SVM) to detect unusual customer behaviour. Deng et al. (2009) propose an active learning procedure through a sequential design method. As graph neural networks (GNNs) have become the de facto tool for performing machine learning tasks on graphs in recent years, Weber et al. (2019) use a GNN for detecting ML transactions.

However, aforementioned methods are specific to AML

[*]Corresponding author.

[1]One of the biggest online payment system in the world that belongs to Ant Group

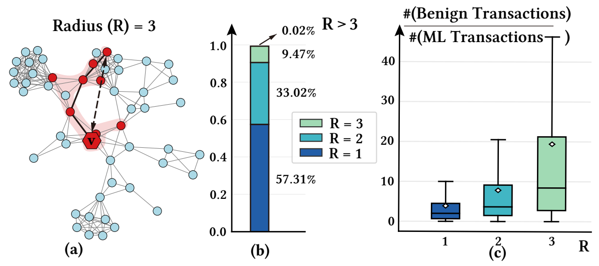

Figure 2: Observations in real-world transaction network of AliPay. (a) illustrates the *radius* of the ML sub-network of transaction $\mathbf{v}$. The radius (R) denotes the maximum distance between $\mathbf{v}$ to all other transactions involved in the ML sub-network. (b) shows the radius distribution of ML sub-networks in transaction network. (c) shows that for an ML sub-network of radius R, how the ratio of the amount of benign transaction (●) to the amount of ML transactions (●) in the R-hop subgraph distributes.

systems operating on the ML transaction detection task, which identify specific suspicious transactions. To discover ML sub-networks, it still heavily depends on methods embedded with pre-defined rules designed from domain knowledge (Dreżewski, Sepielak, and Filipkowski 2015; Li et al. 2017), which is costly on large-scale transaction network and may lag behind the modus operandi of launderers.

To bridge the gap between existing AML systems and industrial risk coverage requirements, it's essential to design a well-behaved AML system that can further discover potential ML sub-network when identifying suspicious ML transactions. It requires the AML system to capture the underlying pattern of ML sub-network involved in a money laundering case. From a real-world online payment network in AliPay, we summarize the following two major challenges: 1). *Multi-hop money laundering chain*. Illegitimate funds laundered may often transfer through multiple transactions to form a complex multi-hop ML sub-network. As Figure. 2(b) illustrates, more than 40% ML sub-networks have a radius larger than 1 (See the definition of radius in Figure. 2(a)). Therefore, to fully track the ML sub-network, we may often need to capture multi-hop underlying patterns and explore exponentially-increased transactions; 2). *Camouflaged Launderers*. Launderers may conduct numerous benign transactions simultaneously to hide themselves. Thus an ML transaction is often connected with a large number of benign transactions in the transaction network. For an ML transaction $\mathbf{v}$ involved in an ML sub-network of radius $\mathbf{R}$, we need to explore $\mathbf{v}$'s $\mathbf{R}$-hop ego-subgraph $\mathcal{G}_{\mathbf{v}}^{\mathbf{R}}$ to uncover the ML sub-network. And we find that the majority of transactions in $\mathcal{G}_{\mathbf{v}}^{\mathbf{R}}$ is benign (Figure. 2(c)). The camouflage strategy brings additional challenges to discriminating ML transactions from massive benign transactions.

Therefore, in this paper, we present the AMAP, a novel neural network based framework for <u>A</u>nti-<u>M</u>oney laundering, equipped with an <u>A</u>daptive sub-network <u>P</u>roposer. Our framework aims at simultaneously addressing the two objectives, ML transaction detection and ML sub-network discovery. Our designed adaptive sub-network proposer is guided by a supervised contrastive loss considering the "camouflaged launderers", which empowers the model to discriminate ML transactions from benign ones. The adaptive sub-network proposer starts from a node and expands the sub-network iteratively to explore underlying patterns of multi-hop money laundering chains. Our framework then utilizes a dual-view fusion classifier, leveraging the potential ML sub-networks yielded by the adaptive sub-network proposer, to identify ML transactions. Compared to AML methods in the literature, AMAP not only predicts ML transactions, but also yields the ML sub-network of a suspicious ML transaction, which leads to a more comprehensive risk control and a deeper understanding of ML strategies.

To summarize, Our main contributions are as follows:

- To our best knowledge, we are the first to address the money laundering sub-network discovering problem by a deep neural network based approach, which is hard to resolve for existing rule-based methods.
- We propose an AML framework AMAP equipped with a novel adaptive sub-network proposer. With designed supervised contrastive loss and iterative generation mechanism, our proposer is endowed with the ability in discriminating ML transactions from massive benign transactions and capturing multi-hop underlying patterns.
- Our AML framework is extensively evaluated in a real-world online payment system. The experimental results show the superiority of our proposed framework on both ML transaction detection and ML sub-network discovery.

Preliminaries

Problem Definition

Transaction Network is a attributed network $\mathcal{G} = \{\mathcal{V}, \boldsymbol{X}, \boldsymbol{Y}, \mathcal{E} = \{\mathcal{E}^+, \mathcal{E}^-\}\}$, where each node $v \in \mathcal{V}$ denotes a transaction with a corresponding feature vector $\mathbf{x}_v \in \mathbb{R}^d$. $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ denotes the feature matrix. $\boldsymbol{Y} \in \mathbb{R}^{|\mathcal{V}|}$ is an indicator vector representing whether node v is a ML transaction or not. An edge $e_{uv} \in \mathcal{E}$ between two transactions u and v indicates the two transactions are related. $\mathcal{E}^+$ denotes the set of edges between money-laundering transactions, $\mathcal{E}^- = \mathcal{E} \setminus \mathcal{E}^+$ and $\mathcal{E}^+ \cap \mathcal{E}^- = \varnothing$.

Money Laundering Sub-network Given $\mathcal{G}^+ = \{\mathcal{V}^+, \boldsymbol{X}^+, \mathcal{E}^+\}$ be the edge-induced subgraph of $\mathcal{G}$ induced by edge set $\mathcal{E}^+$, we define $\mathcal{G}_v^+ = \{\mathcal{V}_v^+, \mathcal{E}_v^+\}$ as the money laundering sub-network of a ML transaction v, where $\forall u \in \mathcal{V}_v^+, |d(u,v)| \leq k$, where $d(u,v)$ is the graph distance on $\mathcal{G}_v^+$ between u and v.

Objectives of Anti-money Laundering Given the observed $\mathcal{G}^t = \{\mathcal{V}^t, \boldsymbol{X}^t, \boldsymbol{Y}^t, \mathcal{E}^t\}$ as the training set, our objectives are to learn an AML model for 1). predicting whether a given transaction v is an ML transaction or not, and 2). yielding the money laundering sub-network $\mathcal{G}_v^+$ if v is identified as an ML transaction. We denote the sub-objectives 1) and 2) as ML transaction detection and ML sub-network discovery, respectively. We follow the inductive setup where v belongs to a new-observed $\mathcal{G} = \{\mathcal{V}, \boldsymbol{X}, \mathcal{E}\}$, and $\mathcal{G}^t$ and $\mathcal{G}$ have no overlap, which conforms with the settings of real-world AML systems.

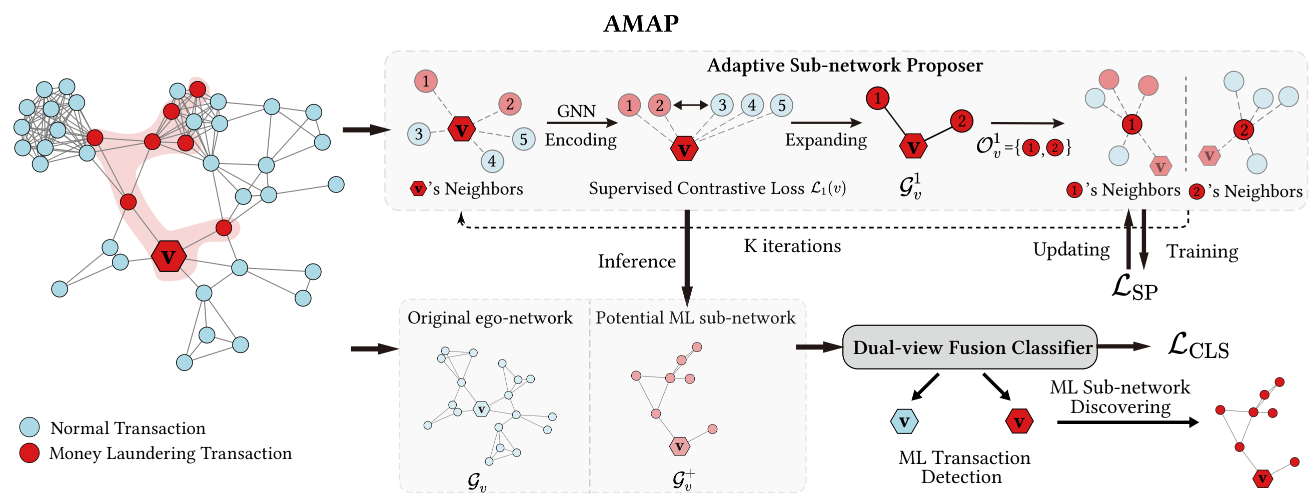

Figure 3: Illustration of AMAP. For convenience, we illustrate both training and inference process on transaction **v**. In practice, the framework could be trained batch-wise and infer inductively. The adaptive sub-network proposer starts from the initial node, and expands the sub-network iteratively. The process of expansion is guided by a supervised contrastive loss. After training the adaptive sub-network proposer, the module is used to propose a potential ML sub-network $\mathcal{G}_v^+$ for inferred transaction **v**. Then a dual-view fusion classifier leverages both the potential ML sub-network and original ego-network to identifying transaction **v**. $\mathcal{G}_v^+$ is outputted as the ML sub-network of **v** when **v** is identified as an ML transaction.

The Proposed Approaches

Overview

The overall framework of AMAP is shown in Figure 3. Given the transaction network as input, we train the adaptive sub-network proposer to generate the potential ML sub-network of a transaction from the massive transaction network. Then for a transaction to be identified, we apply the adaptive sub-network proposer to obtain a potential ML sub-network, which is fed into the dual-view fusion classifier to boost classification performance. The potential ML sub-network will also be the output of AMAP, if AMAP identifies an ML transaction.

Adaptive Sub-network Proposer

Our designed sub-network proposer module aims to generate a potential critical sub-network (i.e. ML sub-network) of a transaction from massive transaction network. The module works in an iterative way. Given the already-generated subgraph in the previous iteration, the module explores the candidate set of nodes connected to the sub-network and expands the suspicious transaction nodes. To distinguish between ML transactions and benign transactions involved in the candidate set, we adopt a loss function in the form of supervised contrastive learning, which pulls together the already-generated subgraph and ML transaction nodes in the embedding space, while simultaneously pushes apart the sub-network from benign nodes. Formally, we can define the module by examining the k-th iteration ($k = 1, \cdots, K$) as follows:

Sub-network Encoding. Here we focus on node v. Given the already-generated sub-network $\mathcal{G}_v^{k-1}$ at the last $(k-1)$-th iteration, we use a GNN on the graph $\mathcal{G}_v^{k-1}$ to get node representations (G_v^0 contains v only). More concretely, the propagation process of the m-th GNN layer is implemented with the following operations:

$$\mathbf{h}_u^{(m)} = \mathrm{UPDATE}\left(\mathbf{h}_u^{(m-1)},\ \mathrm{AGG}\left(\left\{\mathbf{h}_n^{(m-1)} : n \in \mathcal{N}(u)\right\}\right)\right)$$
$$\mathbf{h}_u = \mathbf{h}_u^{(M)} \tag{1}$$

where M is the number of stacked GNN layers and $\mathcal{N}$ is neighboring function. We use the output of the final layer $\mathbf{h}_u$ as node u's representation. And the representation of sub-network $\mathcal{G}_v^{k-1}$ is obtained by aggregating node representations by a READOUT function:

$$\mathbf{h}_{\mathcal{G}_v^{k-1}} = \mathrm{READOUT}\left(\left\{\mathbf{h}_u \mid u \in \mathcal{G}_v^{k-1}\right\}\right) \tag{2}$$

Denoting the nodes expanded into the sub-network in the $(k-1)$-th step as $\mathcal{O}_v^{k-1}$ ($\mathcal{O}_v^0$ contains v only), the candidate node set for the k-th expanding is the neighbors of nodes in $\mathcal{O}_v^{k-1}$:

$$\mathcal{C}_k \triangleq \cup\left\{\mathcal{N}(u) \mid u \in O_v^{k-1}\right\} \tag{3}$$

Our goal is to learn to distinguish between ML and benign transactions in $\mathcal{C}_k$, given the already-generated sub-network. For each node u in O_v^{k-1}, the final representation $\hat{\mathbf{h}}_u$ is defined as the combination of $\mathbf{h}_{\mathcal{G}_v^{k-1}}$ and $\mathbf{h}_u$ to capture both global and local information:

$$\hat{\mathbf{h}}_u = \left(\mathbf{h}_u \oplus \mathbf{h}_{\mathcal{G}_u^{k-1}}\right)\mathbf{W}^s \tag{4}$$

where $\oplus$ denotes concatenating operation and $\mathbf{W}^s \in \mathbb{R}^{2d' \times d'}$ denotes the projection matrix.

Supervised Contrastive Learning. Now for node u in O_v^{k-1}, we have its neighbors to be added into the sub-network in k-th step, where we aim to include ML transaction nodes but exclude benign nodes. To achieve this goal, a supervised contrastive loss (Khosla et al. 2020) is adopted and the loss in k-th step reads

$$\mathcal{L}_k(u) = \mathbb{E}_{n^+ \sim \mathcal{N}^+(u)} \log \sigma \left(\mathcal{S}(n^+, u)\right) + \mathbb{E}_{n^- \sim \mathcal{N}^-(u)} \log \sigma \left(-\mathcal{S}(n^-, u)\right) \tag{5}$$

where $\mathcal{N}^+(v)$ are node u's neighbors which are ML transactions and $\mathcal{N}^-(v)$ are node v's neighbors which are benign. In the case that $\mathcal{N}^+(v)$ or $\mathcal{N}^-(v)$ is empty, the corresponding item in Eq. (5) will not be calculated. $\mathcal{S}(n, u)$ is a similarity function which measure the similarity between node u and its neighbor n. Formally, we implement $\mathcal{S}(n, u)$ in the latent representation space as follows:

$$\mathcal{S}(n, u) = \frac{\mathbf{\Phi}_\theta(\mathbf{x}_n) \cdot \hat{\mathbf{h}}_u}{\|\mathbf{\Phi}_\theta(\mathbf{x}_n)\|_2 \left\|\hat{\mathbf{h}}_u\right\|_2} \tag{6}$$

where $\mathbf{\Phi}_\theta$ is a multilayer perceptron (MLP) that transforms node n's feature $\mathbf{x}_n \in \mathbb{R}^d$ into the latent space $\mathbb{R}^{d'}$, .

Adaptive Thresholding with Virtual Node. At inference time, after calculating the similarity scores through Eq. (6), the module use the scores to select suspicious neighbors (i.e., ML transactions) and expand them into the sub-network. Previous approaches set a global threshold to filter the neighbors as a hyper-parameter (Liu et al. 2021a; Dou et al. 2020). These methods apply global thresholds to all the nodes. However, the optimal threshold for each node could be node-dependent, i.e. an over-sampling threshold for one node could be under-sampling for another node.

To alleviate the problem, we introduce a virtual threshold node (VTN) to serve as an adaptive threshold, which separates the nodes of ML transactions from the benign ones. Formally, the virtual threshold node is implemented as a learnable parameter $\mathbf{h}_{\text{VTN}} \in \mathbb{R}^{d'}$. Thus the supervised contrastive loss in Eq. 5 can be revised in the following form:

$$\begin{aligned} \mathcal{L}_k^1(u) &= \mathbb{E}_{n^+ \sim \mathcal{N}^+(u)} \log \sigma \left(\mathcal{S}(n^+, u) - \mathcal{S}(n_{\text{VTN}}, u)\right) \\ \mathcal{L}_k^2(u) &= \mathbb{E}_{n^- \sim \mathcal{N}^-(u)} \log \sigma \left(\mathcal{S}(n_{\text{VTN}}, u) - \mathcal{S}(n^-, u)\right) \\ \mathcal{L}_k(u) &= \mathcal{L}_k^1(u) + \mathcal{L}_k^2(u) \end{aligned} \tag{7}$$

where n_{VTN} denotes the virtual threshold node and $\mathcal{S}(n_{\text{VTN}}, u)$ is the similarity function and implemented as the cosine similarity of $\mathbf{h}_{\text{VTN}}$ and $\hat{\mathbf{h}}_u$. $\mathcal{L}^1(u)$ encourages the ML transaction neighbors to have higher similarity score than the virtual node, while $\mathcal{L}^2(u)$ pulls the similarity score of benign neighbors to be lower than the virtual node. $\mathcal{L}^1(u)/\mathcal{L}^2(u)$ will not be used if node u has no ML/benign neighbors. Figure 4 illustrates how the virtual threshold node serves as an adaptive threshold by separating ML transaction nodes and benign nodes. The introduction of the VTN empowers the model to learn a personalized threshold for each node u. One can also use an independent VTN at each iteration to improve the model capacity.

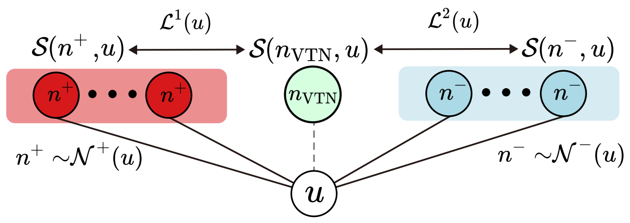

Figure 4: An artificial illustration of our proposed loss for adaptive thresholding. A virtual threshold node (●) is introduced to separate ML transaction nodes (●) and benign nodes (●) in the neighbors of node u.

Training. The overall loss function of the adaptive sub-network proposer is the summation of the K iteration:

$$\mathcal{L}_{\text{SP}} = \sum_{v \in \mathcal{V}_t} \sum_{k=1}^{K} \gamma^{k-1} \sum_{u \in O_v^{k-1}} \mathcal{L}_k(u) \tag{8}$$

where γ is the balance parameter for each iteration and $\mathcal{V}_t$ is the training node set.

The training/inference process of the adaptive sub-network proposer is summarized in Algorithm 1. In training phase, the ground truth is used to build already-generated sub-network for each iteration (Line 9). The iterations on node v will terminate if no nodes are expanded into the sub-network of node v. It is worth noting that to reduce the influence of sample imbalance (the majority of nodes is benign and has no ML transaction neighbors, where the training is performed only for the first iteration), we employ under-sampling technique to train our adaptive sub-network proposer. specifically, we randomly sample the same number of benign nodes as the ML transaction nodes to build the training set $\mathcal{V}_t$.

Dual-View Fusion Classifier

After training the the proposer, we can use it to generate a potential ML sub-network for a node in the transaction network. Ideally, the proposer could extend a ML transaction node into its ML sub-network, and extend none for a benign node, of which the output could be used for classifying ML transactions. However, in practice, the output of the adaptive sub-network proposer could be noisy due to factors such as miss-labeling data, etc. Therefore, to achieve a more robust result, we employ an additional classifier, which fuses the information from the potential ML sub-network and original network, to recognize ML transactions.

More concretely, we focus on node v here. With the adaptive sub-network proposer, we get v's potential ML sub-network $\mathcal{G}_v^+$. We use a GNN to get v's representation $\mathbf{h}_v^p$ on $\mathcal{G}_v^+$ by the means described in Eq. 1. For getting more comprehensive information, we also employ another GNN to encode v' as $\mathbf{h}_v^o$ in the original graph $\mathcal{G}$. We use the attention mechanism $att(\mathbf{h}_v^p, \mathbf{h}_v^o)$ to learn their corresponding importance (α_v^p, α_v^o) as follow:

$$(\alpha_v^p, \alpha_v^o) = att(\mathbf{h}_v^p, \mathbf{h}_v^o) \tag{9}$$

Algorithm 1: Adaptive Sub-network Proposer

```
Input: Transaction Network: $\mathcal{G} = \{\mathcal{V}, \mathcal{X}, Y, \mathcal{E}\}$;
       Training Node Set: $\mathcal{V}_t$; Training Epochs:
       $N_{\text{epoch}}$; Iteration Number: $K$; Test Node: $v$;
Output: Proposed Sub-network $\mathcal{G}_v^+$ for test node $v$
1  ▷ Training phase:
2  for $e = 1, \cdots, N_{epoch}$ do
3      for $u \in \mathcal{V}_t$ do
4          $\mathcal{G}_u^0 \leftarrow \{u\}, \mathcal{O}_u^0 \leftarrow \{u\}$
5          for $k = 1, \cdots, K$ do
6              for $o \in \mathcal{O}_u^{k-1}$ do
7                  $\hat{\mathbf{h}}_o \leftarrow$ Eq. 4
8                  $\mathcal{L}_k(o) \leftarrow$ Eq. 7
9                  $\mathcal{O}_u^k \leftarrow \mathcal{N}^+(o) \cup \mathcal{O}_u^k$
10             end
11             $\mathcal{G}_u^k \leftarrow \mathcal{G}_u^{k-1} \cup \mathcal{O}_u^k$
12         end
13     end
14     $\mathcal{L}_{\text{SP}} \leftarrow$ Eq. 8;
15     Back-propagation to update parameters;
16 end
17 ▷ Inference phase: Initialization $\mathcal{G}_v^0 \leftarrow \{v\}$
18 $\mathcal{O}_v^0 \leftarrow \{v\}$
19 for $k = 1, \cdots, K$ do
20     for $o \in \mathcal{O}_v^{k-1}$ do
21         $\hat{\mathbf{h}}_o \leftarrow$ Eq. 4;
22         for $u \in \mathcal{N}(o)$ do
23             $\mathcal{S}(o, u_{\text{VTN}}^k), \mathcal{S}(o, u) \leftarrow$ Eq. 6
24             if $\mathcal{S}(o, u) > \mathcal{S}(o, u_{VTN}^k)$ then
25                 $\mathcal{O}_v^k \leftarrow \{u\} \cup \mathcal{O}_v^k$
26             end
27         end
28     end
29     $\mathcal{G}_v^k \leftarrow \mathcal{G}_v^{k-1} \cup \mathcal{O}_v^k$
30 end
31 $\mathcal{G}_v^+ \leftarrow \mathcal{G}_v^K$
32 Return: $\mathcal{G}_v^+$.
```

where α_v^p and α_v^o indicates the attention values of node v with representations $\mathbf{h}_v^p \in \mathbb{R}^h$ and $\mathbf{h}_v^o \in \mathbb{R}^h$ from different graph views $\mathcal{G}_v^+$ and $\mathcal{G}$, respectively.

The attention mechanism is instantiated as follow. Firstly, we transform the representation through a nonlinear transformation, and then use one shared attention vector $\mathbf{q} \in \mathbb{R}^{h' \times 1}$ to get the attention weight ω_v^i:

$$\omega_v^i = \boldsymbol{q}^T \cdot \tanh\left(\mathbf{W}^a {\mathbf{h}_v^i}^T + \mathbf{b}\right) \quad \text{for } i \in \{p, o\} \tag{10}$$

where $\mathbf{W}^a \in \mathbb{R}^{h' \times h}$ is the weight matrix and $\mathbf{b} \in \mathbb{R}^{h' \times 1}$ is the bias vector.

We then normalize the attention weights with softmax function to get the final attention values:

$$\alpha_v^i = \frac{\exp\left(\omega_v^i\right)}{\sum_{i \in \{p,o\}} \exp\left(\omega_v^i\right)} \quad \text{for } i \in \{p, o\} \tag{11}$$

Then we combine the two embeddings to obtain the fused embedding $\mathbf{z}$

$$\mathbf{z}_v = \sum_{i \in \{p,o\}} \alpha_v^i \mathbf{h}_v^i \tag{12}$$

We use the output embedding $\mathbf{z}_v$ in Eq. 12 as the final embedding of node v. Based on this, we regard the ML transaction detection problem as a node classification problem, and use cross entropy to model it.

$$\begin{aligned} \mathcal{L}_{\text{cls}} &= -\sum_{v \in \mathcal{V}} \left[y_v \log p_v + (1 - y_v) \log (1 - p_v)\right] \\ p_v &= \operatorname{softmax}\left(\operatorname{MLP}\left(\mathbf{z}_v\right)\right) \end{aligned} \tag{13}$$

where MLP transforms the node embedding into prediction scores and p_v is the probability of node v belonging to ML transactions.

Experiments

In this section, we perform evaluations on the effectiveness of our proposed framework AMAP. Note that AMAP simultaneously addresses two subtasks, namely ML transaction detection and ML sub-network discovery. Therefore, we evaluate AMAP on the two subtasks in separate in this section. More specifically, we aim to answer the following research questions:

- **Q1:** How does our overall framework perform against state-of-the-art baselines on real-world ML transaction detection task?
- **Q2:** Can our designed sub-network proposer effectively ML sub-network discovery of ML transactions? How does it compete against alternative methods like GNN explanation methods.
- **Q3:** Does the designed mechanism of the AMAP work effectively and as expected?

Experimental Setup

Datasets We use the real-world datasets in Alipay, an online payment service provided by Ant Group, which serves more than 1 billions of users. We extract two sub-datasets, named M6 (sampled from transactions occurring between 02/22/22 and 02/28/22) and M12 (sampled from transactions occurring between 02/16/22 and 02/28/22). We extract 400 attributes for each transaction, including profiles of transaction sender/receiver, transaction amount and so on. The dataset is desensitized and encrypted and does not contain any Personal Identifiable Information (PII). The dataset is only used for academic research, it does not represent any real business situation. During the experiment, adequate data protection was carried out to prevent the risk of data copy leakage, and the dataset was destroyed after the experiment. For both datasets, we test the model performance on transactions occurring between 03/02/22 and 03/05/22.

Methods	Dataset	M6			M12		
	Metric	ROC	AUC-PR	GMeans	ROC	AUC-PR	GMeans
Feature-based	SVM	$0.8803_{\pm 0.0186}$	$0.6335_{\pm 0.0711}$	$0.7561_{\pm 0.0230}$	$0.8774_{\pm 0.0162}$	$0.6500_{\pm 0.0511}$	$0.7791_{\pm 0.0294}$
	GBDT	$0.8869_{\pm 0.0377}$	$0.6252_{\pm 0.0443}$	$0.7611_{\pm 0.0217}$	$0.8901_{\pm 0.0306}$	$0.6333_{\pm 0.0061}$	$0.7812_{\pm 0.0114}$
GNN	GCN	$0.8791_{\pm 0.0066}$	$0.6275_{\pm 0.0145}$	$0.7508_{\pm 0.0190}$	$0.8832_{\pm 0.0034}$	$0.6459_{\pm 0.0077}$	$0.7635_{\pm 0.0343}$
	GAT	$0.8682_{\pm 0.0312}$	$0.6102_{\pm 0.0139}$	$0.7464_{\pm 0.0063}$	$0.8572_{\pm 0.0247}$	$0.6160_{\pm 0.0126}$	$0.7289_{\pm 0.0053}$
	GraphSAGE	$0.9082_{\pm 0.0221}$	$0.7026_{\pm 0.0196}$	$0.8239_{\pm 0.0241}$	$0.8937_{\pm 0.0385}$	$0.6685_{\pm 0.0294}$	$0.7746_{\pm 0.0274}$
	GIN	$0.9040_{\pm 0.0115}$	$0.6868_{\pm 0.0137}$	$0.8097_{\pm 0.0106}$	$0.8860_{\pm 0.0412}$	$0.6562_{\pm 0.0328}$	$0.7621_{\pm 0.0299}$
	GeniePath	$0.9002_{\pm 0.0024}$	$0.6892_{\pm 0.0092}$	$0.8098_{\pm 0.0023}$	$0.8970_{\pm 0.0125}$	$0.6797_{\pm 0.0335}$	$0.8002_{\pm 0.0176}$
	FAGCN	$0.9116_{\pm 0.0563}$	$0.7003_{\pm 0.0984}$	$0.8273_{\pm 0.0897}$	$0.9080_{\pm 0.0176}$	$0.6915_{\pm 0.0322}$	$0.8026_{\pm 0.0047}$
GAD	GraphConsis	$0.9003_{\pm 0.0227}$	$0.6961_{\pm 0.0413}$	$0.8139_{\pm 0.0302}$	$0.8856_{\pm 0.0314}$	$0.6445_{\pm 0.0532}$	$0.7956_{\pm 0.0320}$
	CARE-GNN	$\underline{0.9267}_{\pm 0.0193}$	$0.7443_{\pm 0.0392}$	$0.8167_{\pm 0.0229}$	$0.9231_{\pm 0.0186}$	$0.7043_{\pm 0.0359}$	$0.8306_{\pm 0.0379}$
	H^2-FDetector	$0.9217_{\pm 0.0041}$	$\underline{0.7520}_{\pm 0.0292}$	$\underline{0.8364}_{\pm 0.0098}$	$0.9347_{\pm 0.0078}$	$\underline{0.7454}_{\pm 0.0262}$	$0.8426_{\pm 0.0106}$
	PC-GNN	$0.9151_{\pm 0.0241}$	$0.7465_{\pm 0.0094}$	$0.8291_{\pm 0.0014}$	$\underline{0.9355}_{\pm 0.0392}$	$0.7373_{\pm 0.0593}$	$\underline{0.8561}_{\pm 0.0401}$
Ours	AMAP	$\mathbf{0.9513}_{\pm 0.0040}$	$\mathbf{0.7724}_{\pm 0.0175}$	$\mathbf{0.8795}_{\pm 0.0078}$	$\mathbf{0.9678}_{\pm 0.0033}$	$\mathbf{0.7810}_{\pm 0.0336}$	$\mathbf{0.9016}_{\pm 0.0108}$

Table 1: Experimental results (Mean ± Std.) of compared methods on the M6 and M12 datasets

Evaluation on ML Transaction Detection (Q1)

ML Transaction Detection Baselines We compare AMAP with three categories of baseline: 1) Feature-based methods, including SVM (Chang and Lin 2011) and GBDT (Friedman 2001); 2) GNN-based models, including GCN (Kipf and Welling 2017), GraphSAGE (Hamilton, Ying, and Leskovec 2017), GAT (Veličković et al. 2018) GIN (Xu et al. 2018) GeniePath (Liu et al. 2019) and FAGCN (Bo et al. 2021); 3) Graph anomaly detection (GAD) model, including GraphConsis (Liu et al. 2020), CARE-GNN (Dou et al. 2020), PC-GNN (Liu et al. 2021b) and H^2-FDetector (Shi et al. 2022).

ML Sub-network Discovery Baselines Existing rule-based methods focusing on ML sub-network discovery are built on pre-defined rules derived from domain knowledge (Drężewski, Sepielak, and Filipkowski 2015), which is heavily dataset-specific and could not serve as a general comparison method. However, there are two categories of methods that could serve as alternatives to addressing ML sub-network discovery, namely GNN explanation methods and inherently interpretable GNNs. GNN explanation methods identify important graph sub-structure for GNN model prediction results by post-hoc explanation models. Inherently interpretable GNNs reach the same goal by designing inherently interpretable model architecture. In this paper, we compare our methods with state-of-the-art GNN explanation methods, including GNNExplainer (Ying et al. 2019), PG-EXplainer (Luo et al. 2020) SubgraphX (Yuan et al. 2021) and ReFine (Wang et al. 2021), and inherently interpretable GNN methods, including GAT and GSAT (Miao, Liu, and Li 2022).

Implementation Details We use the under-sampled datasets to train our adaptive sub-network proposer as mentioned in Sec. . According to the observation in Fig. 5(c), we set the iteration number K to 3 for both M6 and M12. Then the dual-view fusion classifier is trained with the outputs of the adapative sub-netowrk proposer.

Evaluation Metrics The ML transaction detection datasets are inherently imbalanced, and although ML transactions are in the minority, they are more concerned. In this paper, we adopt three widely-used metrics in the imbalanced setting: AUC-ROC (ROC), AUC-PR and GMeans (Liu et al. 2021b; Shi et al. 2022). All the results are averaged over 10 times tests with different random seeds.

Effectiveness Results Table 1 presents the experimental results and we can make following observations. Overall, AMAP consistently improves the performance for all metrics on both datasets than results of baselines methods, with average 3.02%↑, 3.74%↑ and 6.08%↑ improvement respectively in ROC, AUC-PR and GMean. Among baseline methods, graph anomaly detection methods (CARE-GNN, PC-GNN and H^2-FDetector) outperforms feature-based methods and GNN-based models. We attribute these improvements by GAD methods to their intuitions of distinguishing between anomalous and normal direct neighbors. The highlighted results in the table are from AMAP, which keeps advantage over all the GAD comparison methods consistently. The potential ML sub-network from the adaptive sub-network proposer endows AMAP with the knowledge about the underlying patterns of multi-hop ML sub-networks.

Evaluation on ML Sub-network Discovery (Q2)

Setup We train both baselines and our methods on the M6 datasets. For testing, we sample from ML transactions occurring between 03/02/22 and 03/05/22. For each ML transaction v, we build its 3-hop ego-graph $\mathcal{G}_v^3$. ML transactions in $\mathcal{G}_v^3$ are considered to be the ground truth. We collect ML transactions occurring on 03/01/22 as the validation set. For all the methods, we apply a grid search to tune their own hyper-parameters that archives the best validation performance.

Evaluation Metric We formulate ML sub-network discovery as a binary classification problem in our setup. As Figure. 2(c) indicates, the majority of transactions in $\mathcal{G}_v^3$ of

	F1-SCORE	ACC	POST-HOC
GAT	$0.1391_{\pm.0032}$	$0.3827_{\pm.0113}$	
GNNEXPLAINER	$0.2828_{\pm.0107}$	$0.6889_{\pm.0276}$	✓
REFINE	$0.3423_{\pm.0022}$	$0.7096_{\pm.0079}$	✓
PGEXPLAINER	$0.4317_{\pm.0025}$	$0.7309_{\pm.0184}$	✓
GSAT	$0.4190_{\pm.0305}$	$0.7455_{\pm.0587}$	
SUBGRAPHX	OOT	OOT	✓
AMAP	$\mathbf{0.6764}_{\pm.0032}$	$\mathbf{0.8836}_{\pm.0129}$	

Table 2: Experimental results (Mean ± Std.) on ML sub-network discovery. OOT indicates run-out-of-time.

a ML transaction v may often be benign. Therefore, we report F1-score and Predictive Accuracy (ACC) to evaluate the model's performance. The average F1-score are calculated as the harmonic mean of average precision and average recall following (Forman and Scholz 2010).

Since comparison methods may give possibly different prediction results, we resample the test ML transaction set and only preserve those samples for which all model predictions are correct for a fair comparison.

It's worth mentioning that it's a standard way for explanation methods to use a threshold β or a selection ratio ρ to give the prediction result (Ying et al. 2019). The choice of β or ρ is critical for the performance. We apply a grid search to find the threshold $\beta^\star$ or selection ratio $\rho^\star$ which achieves the best F1-score on the validation set and report the result with $\beta^\star$ or $\rho^\star$ for these methods. We report averaged results over 10 times tests with different random seeds.

Result Comparison and Analysis As shown in Table 2, our method significantly outperform the baselines by a large margin (56.68%↑ in F1-score and 18.52%↑ in ACC). This demonstrates the effectiveness of our adaptive network proposer on discovering ML sub-networks. We attribute these improvements to the explicit ML sub-network modeling: 1) By taking ML sub-networks as explicit supervision information, AMAP is able to capture the underlying patterns of ML sub-network, while baseline methods implicitly infer the ML sub-networks from the prediction results. However, it may be agnostic by what a predictive model determines that a node is belonging to ML transactions; and 2) Conducting the contrastive learning between ML transaction nodes and benign nodes makes AMAP better stratify the discriminative information between ML and benign transactions.We also provide visualizations of the ML sub-networks discovered by AMAP and runner-up methods in Figure 5. We observe that AMAP significantly reduces the number of false positive in the examples than runner-up methods. Also, AMAP yields a connected ML sub-network because it expands from the initial node iteratively, while runner-up methods may give disconnected results.

Ablation Studies (Q3)

We conduct ablation studies from two aspects: First, the superiority of virtual threshold node (Eq. (7)) over global threshold. Second, the effectiveness of Dual-fusion module.

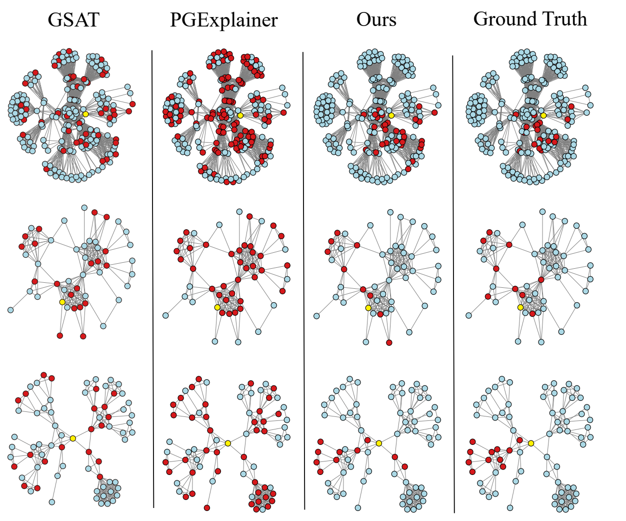

Figure 5: Visualization of ML Sub-network Discovery. (●) indicates the test node.

	F1	ACC	ROC	PR	GMEANS
AMAP	0.6764	0.8836	0.9513	0.7724	0.8795
AMAP-NO-V	0.4919	0.7911	0.9128	0.7370	0.8331
AMAP-NO-F	-	-	0.9244	0.7411	0.8414

Table 3: Ablation studies on M6. We report metric on both ML transaction detection and ML sub-network discovery.

- **AMAP-NO-V**: Replace the virtual threshold node with global threshold searched on the validation set. Training the adaptive sub-network proposer with loss function in Eq. (5) without virtual threshold node.
- **AMAP-NO-F**: Remove the attention fusion module in Eq. (9). This means only the potential ML sub-networks are used for ML transaction detection.

As shown in Table 3, the performance on ML sub-network discovery drops significantly when we replace the virtual threshold node with global threshold. This implies applying virtual threshold node effectively alleviates the lack of adaptivity of using global threshold. Also, removing the attention fusion module causes performance drop on ML transaction detection. It's worth noting that AMAP-NO-F yields competitive results compared with baseline methods in Table 1. It indicates that solely using the adaptive sub-network proposer boosts the performance on ML transaction detection significantly.

Conclusion

In this paper, we present a novel AML framework AMAP. This is the first work in which a neural network based approach has ever been employed to the ML sub-network discovery problem. Our AMAP is designed for discriminating ML transactions from massive benign transactions and capturing multi-hop underlying patterns. We hope this work will bring insights towards a more robust AML system.

Acknowledgments

This work was partially supported by Zhejiang NSF (LR22F020005), the National Key Research and Development Project of China (2018AAA0101900), and the Fundamental Research Funds for the Central Universities.

References

Bo, D.; Wang, X.; Shi, C.; and Shen, H. 2021. Beyond Low-frequency Information in Graph Convolutional Networks. In *AAAI*.

Chang, C.-C.; and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2: 27:1–27:27.

Chen, Z.; Khoa, L. D.; Teoh, E. N.; Nazir, A.; Karuppiah, E. K.; and Lam, K. S. 2018. Machine Learning Techniques for Anti-Money Laundering (AML) Solutions in Suspicious Transaction Detection: A Review. *Knowl. Inf. Syst.*, 57(2): 245–285.

Deng, X.; Joseph, V. R.; Sudjianto, A.; and Wu, C. J. 2009. Active learning through sequential design, with applications to detection of money laundering. *Journal of the American Statistical Association*, 104(487): 969–981.

Dou, Y.; Liu, Z.; Sun, L.; Deng, Y.; Peng, H.; and Yu, P. S. 2020. Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters. In *CIKM*.

Dreżewski, R.; Sepielak, J.; and Filipkowski, W. 2015. The application of social network analysis algorithms in a system supporting money laundering detection. *Inf. Sci.*, 295: 18–32.

Force, F. A. T. 1999. What is money laundering. *Policy Brief July 1999*.

Forman, G.; and Scholz, M. 2010. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *SIGKDD Explor.*, 12: 49–57.

Friedman, J. H. 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29: 1189–1232.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2020. Supervised Contrastive Learning. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 18661–18673. Curran Associates, Inc.

Kingdon, J. 2004. AI fights money laundering. *IEEE Intelligent Systems*, 19(3): 87–89.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

Kute, D. V.; Pradhan, B.; Shukla, N.; and Alamri, A. 2021. Deep Learning and Explainable Artificial Intelligence Techniques Applied for Detecting Money Laundering–A Critical Review. *IEEE Access*, 9: 82300–82317.

Li, X.; Cao, X.; Qiu, X.; Zhao, J.; and Zheng, J. 2017. Intelligent Anti-Money Laundering Solution Based upon Novel Community Detection in Massive Transaction Networks on Spark. *2017 Fifth International Conference on Advanced Cloud and Big Data (CBD)*, 176–181.

Liu, Y.; Ao, X.; Qin, Z.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2021a. Pick and Choose: A GNN-Based Imbalanced Learning Approach for Fraud Detection. In *WWW*.

Liu, Y.; Ao, X.; Qin, Z.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2021b. Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection. In *WWW*.

Liu, Z.; Chen, C.; Li, L.; Zhou, J.; Li, X.; Song, L.; and Qi, Y. 2019. GeniePath: Graph Neural Networks with Adaptive Receptive Paths. In *AAAI*.

Liu, Z.; Dou, Y.; Yu, P. S.; Deng, Y.; and Peng, H. 2020. Alleviating the Inconsistency Problem of Applying Graph Neural Network to Fraud Detection. In *SIGIR*.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized Explainer for Graph Neural Network. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 19620–19631. Curran Associates, Inc.

Miao, S.; Liu, M.; and Li, P. 2022. Interpretable and Generalizable Graph Learning via Stochastic Attention Mechanism. In *ICML*.

Shi, F.; Cao, Y.; Shang, Y.; Zhou, Y.; Zhou, C.; and Wu, J. 2022. H2-FDetector: A GNN-based Fraud Detector with Homophilic and Heterophilic Connections. In *WWW*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*.

Wang, X.; Wu, Y.; Zhang, A.; He, X.; and Chua, T.-S. 2021. Towards Multi-Grained Explainability for Graph Neural Networks. In *NeurIPS*.

Weber, M.; Domeniconi, G.; Chen, J.; Weidele, D. K. I.; Bellei, C.; Robinson, T.; and Leiserson, C. E. 2019. Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. *KDD Workshop on Anomaly Detection in Finance*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How Powerful are Graph Neural Networks. In *ICLR*.

Ying, Z.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, 9244–9255.

Yuan, H.; Yu, H.; Wang, J.; Li, K.; and Ji, S. 2021. On Explainability of Graph Neural Networks via Subgraph Explorations. In *ICML*.