MT²AD: multi-layer temporal transaction anomaly detection in ethereum networks with GNN

Han, Beibei ; Wei, Yingmei ; Wang, Qingyong ; De Collibus, Francesco Maria ; Tessone, Claudio J

DOI: https://doi.org/10.1007/s40747-023-01126-z

Complex & Intelligent Systems (2024) 10:613–626
https://doi.org/10.1007/s40747-023-01126-z

ORIGINAL ARTICLE

MT^{2}AD: multi-layer temporal transaction anomaly detection in ethereum networks with GNN

Beibei Han[1,2] · Yingmei Wei[1] · Qingyong Wang[1] · Francesco Maria De Collibus[2] · Claudio J. Tessone[2,3]

Received: 29 September 2022 / Accepted: 16 May 2023 / Published online: 31 July 2023

Abstract

In recent years, a surge of criminal activities with cross-cryptocurrency trades have emerged in Ethereum, the second-largest public blockchain platform. Most of the existing anomaly detection methods utilize the traditional machine learning with feature engineering or graph representation learning technique to capture the information in transaction network. However, these methods either ignore the timestamp information and the transaction flow direction information in transaction network or only consider single transaction network, the cross-cryptocurrency trading patterns in Ethereum are usually ignored. In this paper, we introduce a Multi-layer Temporal Transaction Anomaly Detection (MT^{2}AD) model in Ethereum network with graph neural network. Specifically, for a given Ethereum token transaction network, we first extract its initial features including the structure subgraph and edge's feature. Then, we model the temporal information in subgraph as a series of network snapshots according to the timestamp on each edge and time window. To capture the cross-cryptocurrency trading patterns, we combine the snapshots from multiple token transactions at a given timestamp, and we consider it as a new combined graph. We further use the graph convolution encoder with attention mechanism and pooling operation on this new graph to obtain the graph-level embedding, and we transform the anomaly detection on dynamic multi-layer Ethereum transaction networks as a graph classification task with these graph-level embeddings. MT^{2}AD can integrate the transaction structure feature, edge's feature and cross-cryptocurrency trading patterns into a framework to perform the anomaly detection with graph neural networks. Experiments on three real-world multi-layer transaction networks show that the proposed MT^{2}AD (0.8789 Precision, 0.9375 Recall, 0.4987 FbMacro and 0.9351 FbWeighted) can achieve the best performance on most evaluation metrics in comparison with some competing approaches, and the effectiveness in consideration of multiple tokens is also demonstrated.

Keywords Anomaly detection · Multi-layer transaction networks · Graph classification · Temporal network · Graph representation learning

Introduction

Blockchain, firstly introduced by Satoshi Nakamoto in his seminal paper,[1] is a distributed ledger technology that can maintain a growing number of data records without a centralized and trusted third party. Its public, decentralized, and tamper-proof approach—first introduced to solve the double spending problem—has been extended to a wide range of other use cases, from supply chain tracking to secure voting.

✉ Yingmei Wei
weiyingmei@nudt.edu.cn

Beibei Han
hanbei969@163.com

Qingyong Wang
wang@cnu.ac.cn

Francesco Maria De Collibus
decollibus@ifi.uzh.ch

Claudio J. Tessone
claudio.tessone@uzh.ch

[1] College of Systems Engineering, National University of Defense Technology, Deya Road, Kaifu District, Changsha 410073, Hunan, People's Republic of China

[2] Blockchain and Distributed Ledger Technologies Group, Universität Zürich, Andreasstrasse 15, 8050 Zurich, Switzerland

[3] UZH Blockchain Center, Universität Zürich, Andreasstrasse 15, 8050 Zurich, Switzerland

[1] https://bitcoin.org/bitcoin.pdf.

Ethereum [1], launched in 2015, is the first platform to implement smart contracts, and nowadays one of the most popular blockchain-based platforms to transact and exchange value, with a throughput of 14.8 transactions per second and almost 700000 daily active address on it [2].

With the recent boom of cryptoassets, the topic of anomaly detection in Ethereum [1] has received considerable attention. For example, in Ethereum, the unexpected appearance of particular subgraphs may indicate newly emerging malware or price pump-and-pump trading [3]. Anomaly detection in blockchain transaction networks is an emerging area of research in the cryptocurrency community [4].

The existing methods about anomaly detection on Ethereum transaction networks, networks where nodes denote the Ethereum transaction addresses and edges denote transactions between addresses, can be roughly divided into two categories [4]:

- *Traditional machine learning with manual-extracted features*: these methods first manually extract the transaction features (such as, transaction value), then, they apply traditional machine learning classification methods to perform the classification tasks. However, these feature engineering methods need expert knowledge, and the classification results mainly rely on the manual-extracted features, which ignore the structural information of the transaction network resulting in suboptimal results. And these methods are unable to automatically extract the transaction network features.
- *Node representation learning*: these methods adopt the random walk node representation learning [5, 6] or graph neural network (GNN) [4] technique to automatically learn the deep features in transaction network to obtain each node's embedding for further classification task to identify the anomaly node and normal node. However, most of them ignore the timestamp information or the transaction flow direction information.

These methods overall achieve excellent results. However, there are still some problems yet to be solved, and room for further improvement:

1. *Multi-token transaction* An address (account) may involve multiple token transaction network. For example, money laundering tends to move funds across multiple cryptocurrency ledgers [7].
2. *Temporal transaction information* The existing methods either ignore the edge timestamp information, or only consider the last transaction record without fully model the information in transaction network.
3. *Growing of transaction data* An amount of nodes in the transaction network, e.g., the largest number of new nodes in 1 day is more than 10,000 in BNB (Binance Coin) token [8][2] transaction network. When 1-year transaction information and multi-token transactions are considered, the transaction network becomes so large that is difficult to analyze.
4. *Edge's feature* The original constructed transaction network we crawl by Ethereum Client with Ethereum-ETL is a weighted directed graph. There are transaction value and transaction flow direction information on each edge. The traditional graph neural network [9] technique is unsuitable for weighted directed networks. Because the adjacency matrix is symmetrical in the traditional graph neural network, and it is not easy to define a symmetric, real-valued laplacian for the directed graph.

To address the above challenges, we propose a Multi-layer Temporal Transaction Anomaly Detection (MT^2AD) model in Ethereum network with graph neural network technique, and transform it as a graph classification task. It mainly contains three modules:

1. Extraction of token transaction network initial feature. It includes the token subgraph sampling and modeling edge's feature to address the challenges 3 and 4.
2. Multi-layer token transaction network snapshot construction. It includes the token network snapshot extraction of a given token subgraph and multiple token snapshots combination at the same timestamp to address the challenge 1 and 2.
3. Anomaly detection with graph representation learning. It aims to transform the anomaly detection as a graph classification task.

We obtain the multiple token transaction information by Ethereum Client with Ethereum-ETL,[3] and model each token transaction information as a temporal weighted directed graph, where nodes represent the transaction address, edges represent the transaction between two address and its direction denotes the transaction flow from one address to another address, edge weight denotes the transaction value, and timestamp information is on each edge. Extraction of token transaction network initial feature module includes subgraph feature sampling and edge's feature modeling. Subgraph sampling component [10] is to sample nodes on each token transaction network. Because most daily token networks have <100 nodes [11], subgraph sampling can reduce the size of transaction network to be processed and has minimal impact on results. First, we extract fixed-size nodes according to the p most active edges in each transaction network to obtain a subgraph. Besides, as the subgraph is a weighted

[2] https://etherscan.io/token/0xB8c77482e45F1F44dE1745F52C74426C631bDD52.

[3] https://github.com/blockchain-etl/ethereum-etl.

directed graph, each edge has transaction value and direction information, which are important features in a transaction network. In order to reduce information loss when learning graph representation with graph neural network encoder, we transform the transaction value and direction information on each edge as node attributes. Second, the multi-layer token transaction network snapshot construction module is to extract time information and model the cross-cryptocurrency trading patterns. We first transform each token transaction subgraph network as a series of network snapshots according to edge's timestamp and time window Δt. Then, multiple token network snapshots with the same timestamp will be combined as a new combined graph for further capturing the cross-cryptocurrency trading patterns with graph neural network. Third, in anomaly detection with graph representation learning module, the graph neural network with attention mechanism [12] is utilized to encode each node information as a embedding on each combined graph. Not only the node's structural context information, and the impact from its different neighborhoods are considered, but also the common information among different token transaction networks are captured with the same graph encoder. Then, the mean pooling operation is adopted on nodes' embeddings of each combined graph to achieve entire graph-level representation (i.e., a vector), for further graph classification (normal or abnormal). Extensive experiments are carried out on three real-world transaction network datasets to demonstrate the effect and effectiveness of our proposed method.

In summary, the key contributions of this paper are:

- We introduce a Multi-layer Temporal Transaction Anomaly Detection (MT^2AD) model in Ethereum network with graph neural network technique. Compared with the existing anomaly detection methods for blockchain transaction network, our MT^2AD considers the cross-cryptocurrency trading patterns.
- The anomaly detection in multi-token transaction network is naturally transformed as a graph classification task to classify the combined graph (normal or abnormal) with the graph-level embedding.
- Extensive experiments on three real-world transaction datasets including two Ethereum multi-token transaction networks and Ripple multi-blockchains networks demonstrate the effectiveness of our proposed model in comparison with some competing approaches.

The remainder of this paper is organized as follows. In Sect. 2, we summarize some researches related to our work. "Problem definition" introduces the problem statements, and "Model" is our method. In "Experiments", we carry out our experiments. "Conclusion" concludes this paper.

Related works

Node representation learning

Node representation learning [13–18], called node embedding or graph embedding or graph representation learning, is designed to learn low-dimensional dense vector (or embedding) for all nodes in a graph. It maps each node in a graph into a low-dimensional vector space, which captures the nodes' similarity, network structure and/or other attributes. Inspired by the great success of deep neural networks in computer vision field, Kipf et al. [9] extended the convolutional operation from image data to graph structure data, and proposed Graph Convolutional Networks (GCNs). GCNs can learn both graph structural and node attributes information simultaneously through iteratively aggregating neighbor's information to current node, i.e., message passing mechanism. On the basis of GCNs, GAT [12] (Graph Attention Networks) utilizes the attention mechanism to learn node importance of different neighbors, and GraphSAGE [19] is an inductive representation learning on large graphs. GraphSAGE generates node embeddings by sampling fixed local node neighborhoods and aggregates these node's features. Since then, lots of graph neural network algorithms have been proposed. The representative methods are Graphormer [20], Auto-GNAS [21], CS-TGN [22] and DeepRank-GNN [23].

Anomaly detection in blockchain transaction network

Trans2vec [6] is a phishing scam detection on Ethereum transaction network based on network embedding technique. It first constructed the Ethereum transaction network as a weighted directed network. Then, on the basis of node2vec [13], Trans2vec proposed a random walk-based network embedding method taking the timestamp and edge transaction value into consideration to sample node sequences. Finally, it utilizes one-class support vector machine to perform node classification into normal or phishing account. LiangChen [5] proposed a phishing scam detection method in Ethereum transaction network based on graph autoencoder. It first utilized the random walk technique to sample a fixed-size subgraph for a random selected node. Then, it utilized GCN to extract the feature of each subgraph. Finally, the output of GCN and attributed matrix were concatenated as the final result for further phishing account classification with LightGBM.[4] Yijun [24] proposed an attributed ego-graph embedding framework to perform phishing detection on Ethereum transaction network through extracting ego-graphs for each labeled account, and then learned graph representa-

[4] https://lightgbm.readthedocs.io/en/latest/.

tions for each ego-graph through a Skipgram model [25]. This article adopted a decision tree to perform detection task on the obtained embeddings. TTAGN [4] is a temporal transaction aggregation graph network representation learning method for Ethereum phishing scams detection. It modeled the temporal relation of historical transaction records between nodes to construct the edge representation of Ethereum transaction network. Besides, the edge representations were fused with topological interactive relations through the graph neural network technique for further phishing address detection. TSGN [26] is the transaction subgraph networks for identifying Ethereum phishing accounts. It built the Ethereum phishing account identification as a graph classification task. TSGN first constructed a set of transaction subgraph for each phishing address and normal address. Each obtained transaction subgraph had a label (phishing or normal). Then, TSGN learned the graph embedding of each transaction subgraph. The last step used the obtained graph embeddings to train a classification to predict the labels of each transaction subgraph. However, these anomaly detection methods based on the graph representation learning technique are all only taking single transaction network into consideration. A growing number of cryptocurrency criminals utilize cross-cryptocurrency trades to hide their identity [7]. Ofori-Boateng [11] has demonstrated that considering multi-token transaction networks can improve the anomaly detection accuracy. In [11], a stacked persistence diagram (SPD) method were proposed to perform the topological anomaly detection on dynamic multi-layer blockchain networks. However, the edge's features were not taken into consideration, such as transaction amount between different accounts, node's degree, etc. These features are important to classify account into normal and anomalous one. Besides, it is unable to automatically learn the deep features in transaction networks taking advantage of the deep learning technique. Readers can refer to the survey [27] about graph anomaly detection with deep learning technique.

Graph classification using graph neural network

Given a set of graphs $\{g_1, g_2, \ldots, g_n\}$, and only partial graphs with labels, the graph classification is to predict the label of unseen graphs. Graph classification can be divided into graph kernel algorithms [28, 29] and neural networks technique [30]. Graph kernel methods [28, 29] is to define a distance to measure the similarity between two graphs, and classify them using this metric, including random walk kernel, shortest path kernel, Weisfeiler Lehman subtree kernel and deep graph kernel et al. However, graph kernel algorithms need handcrafted features (random walk, shortest path, etc.) and hence result in poor generalization. In this paper, we focus on using neural network technique to perform graph classification. The classical algorithm is Graph2vec [30]. It is the first neural embedding framework to learn data-driven distributed representations of arbitrary sized graphs, and is an unsupervised methods. Graph2vec introduced the Skipgram model [25] of word2vec to graph, and it considered a graph as a document, and the rooted subgraph around the nodes as words. On the basis of Graph2vec [30], GL2vec [31] utilized the line graphs (edge-to-vertex dual graphs) of input graphs to handle the edge labels and captured more structural information. Another graph classification algorithms using neural network technique are to perform multiple feature transformations on the input graphs by graph convolution operation, and then performs a pooling operation to reduce the scale of input graph. This process can be repeated many times, and finally the entire graph-level representation (i.e., a vector) is obtained for further graph classification. The representative algorithms are DiffPool [32] and SAGPool [33]. GAU-Nets [34] is a graph neural network model that has strong processing capabilities for graph structures. The model is proposed for image classification and has been tested on MS-COCO and other datasets. The experimental results have proved that GAU-Nets performed better than other traditional graph neural network models. Graph U-Nets [35] is a graph neural network model that uses an encoder–decoder architecture for graph data to the node classification and graph classification tasks. DGCNNII [18] is an end-to-end model called Deep Graph Convolutional Neural Network II, that can perform graph classification with up to 32 layers. It can extract multiscale features of nodes. Readers can refer to the survey [36] about the graph classification algorithms.

Problem definition

Definition 1 (*Dynamic graph*) A dynamic (or temporal) graph can be represented as $G_d = \{G_{t_1}, G_{t_2}, \ldots, G_{t_T}\}$, the $G(t_t) = (E(t_t), V(t_t))$ is the network snapshot at timestamp t, where $E(t_t)$ is the edge set and $|E(t_t)| \geq 1$, all vertices in the edge set $E(t_t)$ at timestamp t are nodes of $G(t_t)$ denoted by $V(t_t)$. The maximum timestamp is T. It should be noted that not all nodes in G will be known at timestamp t, as some new nodes may emerge at timestamp t' for any $t' > t$.

Definition 2 (*Multi-layer graph*) A multi-layer graph can be represented as $G_m = \{G^{l_1}, G^{l_2}, \ldots, G^{l_L}\}$ with $|L|$ layer graph, where $G^l = (V^l, E^l)$ is the l-th graph, $|V^l|$ or n^l and $|E^l|$ represent the number of nodes and edges respectively.

Definition 3 (*Attributed multi-layer graph*) When the nodes in the multi-layer graph contain attributes information, this multi-layer graph G_m is called attributed multi-layer graph. And the node's attributes matrix in G^{l_L} is denoted by $X^{l_L} \in \mathcal{R}^{n^{l_L} \times F}$, where F is the node attribute dimension and n^{l_L} is the number of nodes in G^{l_L}.

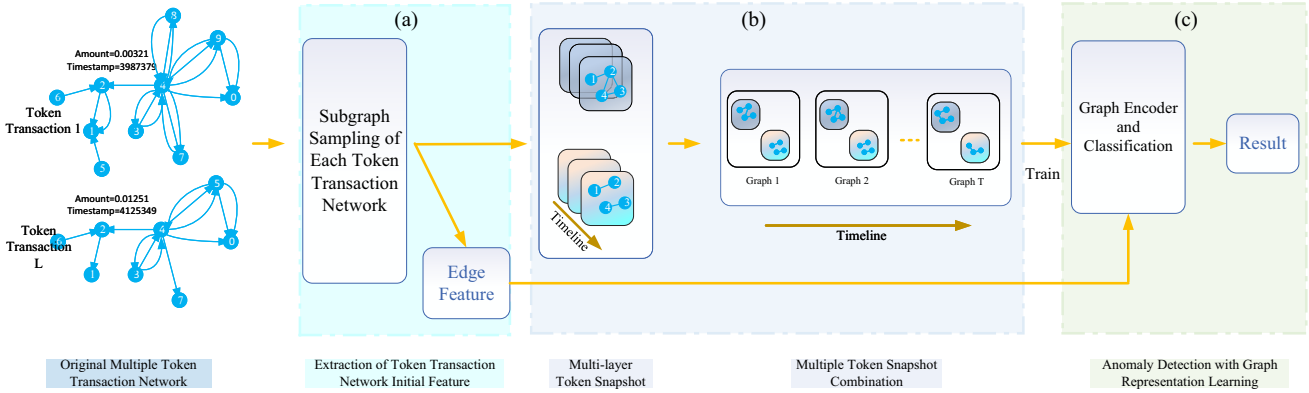

Fig. 1 The framework of MT^{2}AD. It includes three modules: **(a)** Extraction of token transaction network initial feature module is to sample fixed number of nodes according to the p most active edges in each token transaction network to achieve a subgraph feature. At the same time, the edge's feature, i.e., transaction value and transaction flow direction information, are transformed as node's attributes. **(b)** Multi-layer token transaction network snapshot construction module is first to transform each token subgraph as a series of snapshot networks according to the timestamp of the edges and time window Δt, and then to combine multiple token network snapshots with the same timestamp t into a new combined graph $G^m_{t_t} = \{G^{l_1}_{t_t}, G^{l_2}_{t_t}, \ldots, G^{l_L}_{t_t}\}$. (c) Anomaly detection with graph representation learning module is to encode each $G^m_{t_t}, t \in \{1, 2, \ldots, T\}$ as a vector for further graph classification

Definition 4 (*Dynamic attributed multi-layer graph*) A dynamic attributed multi-layer graph with T timestamps and L tokens transaction networks can be denoted as $G_{dm} = \{\{G^{l_1}_{t_1}, G^{l_2}_{t_1}, \ldots, G^{l_L}_{t_1}\}, \ldots, \{G^{l_1}_{t_t}, G^{l_2}_{t_t}, \ldots, G^{l_L}_{t_t}\}, \ldots, \{G^{l_1}_{t_T}, G^{l_2}_{t_T}, \ldots, G^{l_L}_{t_T}\}\}$. Where $G^m_{t_t} = \{G^{l_1}_{t_t}, G^{l_2}_{t_t}, \ldots, G^{l_L}_{t_t}\}$ is the multi-layer graph at timestamp t. Besides, all nodes in G_{dm} have node attributes with same dimension F.

Definition 5 (*Graph classification with representation learning*) The purpose of graph classification is to predict the class of a graph. In this paper, we consider the anomaly detection in dynamic attributed multi-layer token transaction network as a graph classification task.

Learning a map function f to encode n nodes in an attributed multi-layer graph $G^m_{t_t}$ at timestamp t into d-dimensional dense vector space, which can be represented as a matrix $H \in \mathcal{R}^{n \times d} (0 < d \leq n)$, $H = \{h_1, h_2, \ldots, h_n\}$. Then, the pooling operation (mean pooling or max pooling etc.)—readout function [36]—is adopted on matrix H to generate a vector (mean pooling example: $\mathcal{R}(H) = \sigma(\frac{1}{n}\sum_{i=1}^{n} h_i)$, σ is the Sigmoid function). $\mathcal{R}(H)$ is the graph-level representation of $G^m_{t_t}$ and contains the entire graph information. There are T multi-layer token graph snapshots, which means we can achieve T graph-level representation vectors. The classifier is trained with those T graph-level embeddings to classify its corresponding combined graph as normal or abnormal.

Model

Overall framework

The proposed MT^{2}AD, a Multi-layer Temporal Transaction Anomaly Detection method in Ethereum networks with GNN, includes three modules, which are extraction of token transaction network initial feature module, multi-layer token transaction network snapshot construction module, and anomaly detection with graph representation learning module. The over framework is shown in Fig. 1.

Extraction of token transaction network initial feature

We obtain the ERC-20 token transaction records by Ethereum Client with Ethereum-ETL. According to the specification, ERC-20 is the Ethereum standard for fungible tokens [8], and is the interface that a smart contract can implement to exchange this kind of tokens. In this paper, the Binance Coin (BNB), Tether (USDT)[5] and Chainlink (LNK)[6] ERC-20 Ethereum-based token transaction records are extracted by Ethereum Client with Ethereum-ETL. We construct transaction records of each token as a transaction network, where nodes represent the addresses and edges represent the transaction between a pair of addresses. It should be noted that the

[5] USDT: 0xdAC17F958D2ee523a2206206994597C13D831ec7.

[6] Chainlink (LNK): 0x514910771AF9Ca656af840dff83E8264EcF986CA.

token transaction network is a temporal weighted directed network, where weight represents the transaction amount from the source address to the destination address. The timestamp information (an integer number here) on each edge represent the time that transaction is executed. Given the large size of the original token transaction network, in extraction of its initial feature module, we will first extract its subgraph feature (i.e., subgraph sampling), and then model edge's feature as node attributes for further graph convolution operation [9].

Token subgraph sampling

The scale of original transaction network is so large, it is not possible to address such a large dataset using the common GPU with deep learning technique. Taking BNB token transaction network as an example, the number of new daily nodes reaches up to around 10,000 in just 1 day, as shown in Fig. 2. If the BNB transaction information of 1 year are taken into consideration, the transaction network will be very large. Therefore, to maintain reasonable computation time, the first step is to sample the original token transaction network to achieve a subgraph that retains the prominent feature. The common and simple sample method is the random walk technique, which has been utilized in some related works [26]. However, because of the inherent randomness of random walk, therefore, we adopt the maximum weight subgraph sampling method [10] on original token transaction network to achieve a small size transaction subgraph. We restrict the size of transaction subgraph through extracting the p most active edges. The obtained transaction subgraph has minimal impact on results. The reason is that even for the most traded tokens (Tronix and Bat), only top 150 nodes in daily networks form 75% and 80% of all edges [11]. Subgraph sampling is not only able to reduce the computation time and GPU requirement, but also retain the important feature as much as possible [11].

Modeling edge's feature

There are transaction values and direction information on each edge in each transaction network, and these information are basic and important feature in a transaction network. However, traditional graph convolutional network technique [9] is unsuitable for weighted directed graph. If the transaction values and direction information are transformed as node attributes, to some extent, it can reduce the information loss when learning graph representations with graph convolution operation. In this article, we utilize node attribute matrix X to model edge's transaction values and direction information. They are described in the following way:

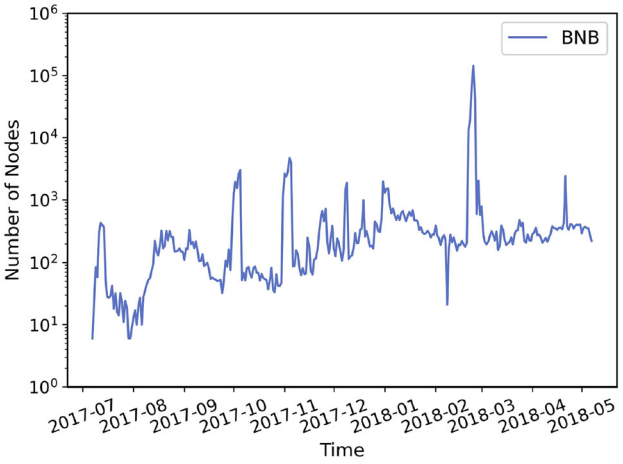

Fig. 2 The number of new nodes every day for the BNB token transaction network

- *Out-degree* For a given node, the out-degree means the out-going edges [37]. In transaction network, it represents the number of transactions that the current node sends to other nodes.
- *In-degree* For a given node, the in-degree means the in-going edges [37]. In transaction network, it represents the number of transactions that other nodes send to the current node.
- *Degree* For a given node, the degree means the one-hop neighborhoods [37]. In transaction network, it represents the total number of edges (or transaction) about the current node. The degree can reflect the activity or importance of a node in a graph.
- *Normalized transaction amount* The transaction amount represents the weight between a pair of addresses. For simplify the calculation and better comparison, we normalized the weight (transaction amount) in [0,1] through dividing by the maximum of all node transactions.

Therefore, edge's feature is transformed as node attributes $X \in \mathcal{R}^{n\times 4}$, which contains four dimensions and n is the number of nodes.

Multi-layer token transaction network snapshot construction

In "Modeling edge's feature", the transaction values and transaction flow between linked nodes have been transformed as node attributes. In this section, we will construct the token network snapshot according to edge's timestamp of a given token transaction network. It mainly includes two steps. First step is to extract token network snapshot according to edge's timestamp of each token transaction network. Second, at given timestamp t, the multiple token network snapshot $G_{t_t}^{l_1}, G_{t_t}^{l_2}, \ldots, G_{t_t}^{l_L}$ is combined as a new combined

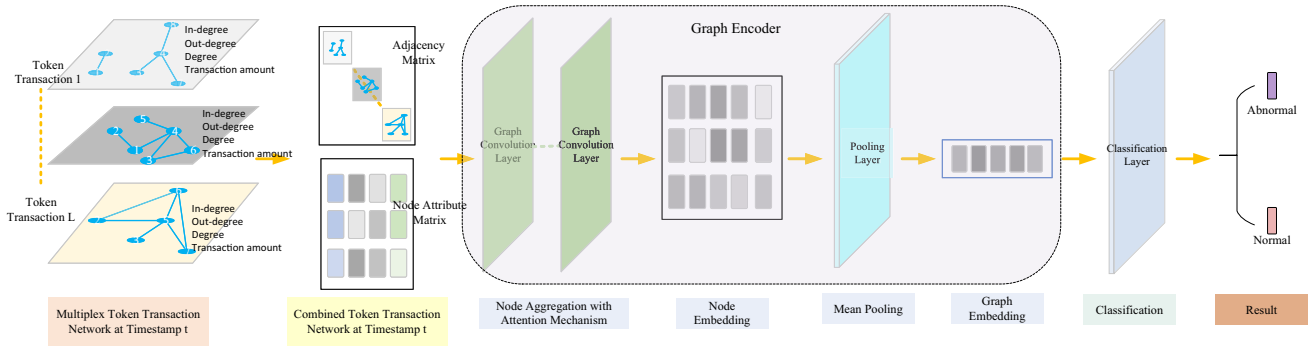

Fig. 3 Anomaly detection with graph representation learning

graph (i.e., $G_{t_t}^m = \{G_{t_t}^{l_1}, G_{t_t}^{l_2}, \ldots, G_{t_t}^{l_L}\}$) for downstream graph representation learning. It is shown in Fig. 1b.

Token network snapshot extraction

Given the l-th dynamic token transaction network G_d, we transform it as a series of network snapshots according to edge's timestamp. The obtain network snapshots can be represented as $G_d^{l_l} = \{G_{t_1}^{l_l}, G_{t_2}^{l_l}, \ldots, G_{t_T}^{l_l}\}$, where $t' = t + \Delta t$ and Δt is the time window. It means $t_2 = t_1 + \Delta t$. Δt can be set as different time value according to the requirements of different analysis tasks. In this paper, Δt is set as 1 day (24 h). For different token transaction networks (i.e., different layers $l_l, l_2, \ldots, l_L$), we can obtain its corresponding network snapshots according to timestamp, $G_d^{l_1} = \{G_{t_1}^{l_1}, G_{t_2}^{l_1}, \ldots, G_{t_T}^{l_1}\}, G_d^{l_2} = \{G_{t_1}^{l_2}, G_{t_2}^{l_2}, \ldots, G_{t_T}^{l_2}\}, \ldots, G_d^{l_L} = \{G_{t_1}^{l_L}, G_{t_2}^{l_L}, \ldots, G_{t_T}^{l_L}\}$

Multiple token snapshot combination

It is critical to analyze cross-cryptocurrency transactions [7] for identifying illicit activities on blockchain. The existing research [7] has shown that the patterns of behavior is largely based on the role they play in tracking money as it moves across the ledgers of different cryptocurrencies [7]: therefore, there are common patterns of behavior across different token transaction networks. However, in "Token network snapshot extraction", we only obtain a series of network snapshots of each dynamic token transaction network respectively. To take advantage of different token transaction networks information and capture common feature among different token transaction networks with graph representation learning, we combine multiple token transaction networks information at a given timestamp t_t as a graph $G_{t_t}^m = \{G_{t_t}^{l_1}, G_{t_t}^{l_2}, \ldots, G_{t_t}^{l_L}\}$. $G_{t_t}^m$ is the combined transaction graph from $l_1, l_2, \ldots, l_L$ token transaction networks at timestamp t_t, i.e., network snapshots $\{G_{t_t}^{l_1}, G_{t_t}^{l_2}, \ldots, G_{t_t}^{l_L}\}$. In this way, we can obtain T combined transaction graph $G_{t_1}^m, G_{t_2}^m, \ldots, G_{t_T}^m$ for further graph classification.

Anomaly detection with graph representation learning

In this section, we will introduce our graph encoder to learn node embedding of each combined transaction graph (i.e., $G_{t_t}^m, t_t \in \{t_1, t_2, \ldots, t_T\}$) for graph pooling operation to achieve entire graph-level embedding, and further graph classification anomaly detection to predict the label of each combined transaction graph. The labels of each combined transaction graph are generated according to the Blockchain daily events from Wikipedia.[7] And these labels are also used in the article [11]. Therefore, we construct the labels of each combined transaction graph according to the Blockchain daily events from Wikipedia. The labels of each combined transaction graph is anomaly and normal. Therefore, in the paper, we transform the anomaly detection as a graph classification task. We learn the graph-level embedding of each combined transaction graph. Then, we use the graph embeddings to train the graph classifier to predict the labels of each combined transaction graph. The multiple token transaction networks anomaly detection with graph representation learning is shown in Fig. 3, and it includes Graph Convolution Layer, Pooling Layer and Graph Classification Layer.

Graph information encode

Graph convolution layer To perform the anomaly detection on combined graph with representation learning technique, we first have to obtain each node embedding in $G_{t_t}^m$. The pooling operation is then utilized on all node embeddings in $G_{t_t}^m$ to achieve the overall graph-level embedding, which is a vector retaining overall graph feature information used for further graph classification. However, in token transaction networks, the node usually conducts a transaction with multiple nodes at the same time and the interactive node pairs usually have similarity characteristic. For example, if a node is a black node in black market transaction, nodes

[7] https://en.wikipedia.org/wiki/History_of_bitcoin.

with direct transactions with this black node may also be black. Besides, the nodes itself in original transaction network only have structure feature, and the transformed node attributes from edge's features in "Modeling edge's feature" are sparse. If only the structure feature and node attributes of node itself are taken into consideration when performing the graph encoder operation, it will obtain an unsatisfied node embedding. To solve this problem, we take advantage of the neighborhood information by aggregation operations which integrates the information from its neighbors to the current node. However, different neighbor nodes have different effects to the current node. Therefore, we utilize the multi-head attention mechanism [12] to capture the difference.

For each node in a given transaction network, we first learn the effect weights between the node itself and its neighbor nodes. The weighted aggregation operation will then be performed according to the obtained weights. For a given node n_i, its one-hop neighborhoods are $N(n_i)$, and $n_j \in N(n_i)$. The effect weights can be learned given the feature of n_i and its neighborhoods $N(n_i)$ with the following method:

$$e_{ij} = \sigma(a_w^T([W \cdot h_i] \| [W \cdot h_j])) \tag{1}$$

where h_i and h_j are the node features of n_i and n_j, respectively. W is the learnable parameter matrix. a_w^T is the attention parameter matrix. $\cdot^T$ denotes the transpose and $\|$ is the concatenate operation. σ is the activation function.

Then, the softmax function is adopted to transform the effect weights coefficients to [0, 1] in Eq. 2.

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{n_j \in N(n_i)} \exp(e_{ij})} \tag{2}$$

Finally, the node embedding of n_i can be generated through the aggregation operation with multi-head attention [12] in Eq. 3. The schematic diagram of multi-head attention is shown in Fig. 4.

$$h_i' = \|_{k=1}^{K} \sigma\left(\sum_{n_j \in N(n_i)} a_{ij}^k W^k h_j\right) \tag{3}$$

where $\|$ represents the concatenation operation, a_{ij}^k are normalized attention coefficients through the k-th attention (a^k). W^k is the linear transformation matrix. h_j is the node embedding of neighborhood of node n_i.

Pooling layer Through the aggregation operation with attention mechanism, we can obtain each node embedding $(H_{t_1}^m, H_{t_2}^m, \ldots, H_{t_T}^m$, where $H_{t_t}^m = \{h_1, h_2, \ldots, h_n\}_{t_t}^m)$ in combined graph at each timestamp i.e., $G_{t_1}^m, G_{t_2}^m, \ldots, G_{t_T}^m$. Then, we adopt readout function (mean pooling operation [38], Eq. 4) on all nodes of a given graph $G_{t_t}^m$ to achieve the

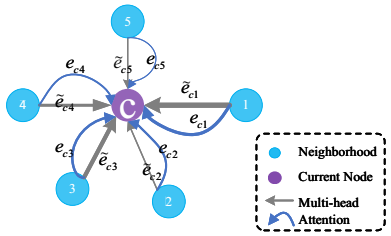

Fig. 4 Aggregating node information with multi-head attention mechanism

overall graph-level representation, which means to reduce a whole graph to a single vector for further graph classification anomaly detection.

$$\mathcal{R}(H)_{t_t}^m = \sigma\left(\frac{1}{n}\sum_{i=1}^{n} h_i\right)_{t_t}^m \tag{4}$$

where σ is the Sigmoid function, $\mathcal{R}$ means the readout function.

Graph classification loss function

In this paper, we utilize the graph classification task to perform anomaly detection, and each $G_{t_t}^m$ corresponds to a label (normal or abnormal). We minimize the cross-entropy loss to train the proposed model, which is as follows:

$$\min_{\theta} L_{GC} = \min_{\theta} \sum_{v_i \in S} \sum_{i=1}^{C} [-y_i \log \hat{y}_i] \tag{5}$$

where S represents the training samples, y_i is the label of graph g_i, $\hat{y}_i$ is the predicted label of graph g_i. C represents the number of graph classes. θ is the learnable parameters, and we summarize it as $\theta = \{W, e\}$.

The loss function is updated by the gradient descent method, and the parameters θ are learned via back-propagation algorithm. While the loss function L_{GC} is not converged, the partial derivative $\frac{\partial L_{GC}}{\partial \theta}$ is calculated and the parameters θ are optimized via the back-propagation algorithm. The MT^{2}AD algorithm is summarized in Algorithm 1.

Time complexity

The MT^{2}AD include three modules: extraction of token transaction network initial feature, multi-layer token transaction network snapshot construction, and anomaly detection with graph representation learning. In the module about the extraction of token transaction network initial feature, the token subgraph sampling and modeling edge's feature can be calculated in advance. And in the module about multi-layer token transaction network snapshot construction, the token network snapshot extraction and multiple token snapshot

Algorithm 1 The MT^{2}AD algorithm

Input: T combined transaction graph $G^m_{t_1}, G^m_{t_2}, \ldots, G^m_{t_T}$, the dimension of node embedding d, node attribute matrix X.
Output: Graph classification results.
1: Initialize the parameters $\theta = \{W, e\}$.
2: **while** the loss function Eq.(5) is not converged **do**
3: learn the weights from different neighbors in Eq.(1) and Eq.(2);
4: aggregate information from different neighbors in Eq.(3) to achieve each node embedding $h_1, h_2, \ldots, h_n$ for $G^m_{t_t}$;
5: graph pooling operation with Eq.(4);
6: calculate the $\frac{\partial L_{GC}}{\partial \theta}$ and optimize the parameters θ by the back-propagation algorithm.
7: **end while**

combination can also be calculated in advance. Therefore, the time complexity of the MT^{2}AD is mainly about the anomaly detection with graph representation learning. We use the graph convolutional network with multi-head attention mechanism to extract the features of the T combined transaction graph, and the average pooling is used to obtain the graph-level representation vector. The time complexity depends on the type of convolution and the number of layers. The convolution operation on graphs involves multiplying the node features by a weight matrix and then aggregating the features from the neighboring nodes. The weight matrix multiplication has a time complexity of $O(|V|FF')$, where F is the dimension of node features and F' is the number of output feature. The aggregation step has a time complexity of $O(|E|F')$, where $|E|$ is the number of edges in the graph. Therefore, the overall time complexity of one convolution layer is $O(|V|FF'+|E|F')$. The attention mechanism involves computing a pair-wise similarity score between each node and its neighbors, and then applying a softmax function to normalize the scores. The time complexity of the attention mechanism is $O(F|V|^2)$. In this paper, we use the multi-head attention, each head's calculations are independent, and can be paralleled. As for the average pooling, the time complexity is $O(|V|)$, and $|V| \ll |V|^2$. Therefore, the overall time complexity of MT^{2}AD can be simplified as $O(|V|FF' + |E|F' + F|V|^2)$.

Experiments

In this section, experimental results of the proposed MT^{2}AD for anomaly detection method are presented. We aim to answer the following Research Questions (RQ):

- *RQ1* How the proposed method benefits the anomaly detection of multiple token transaction network?
- *RQ2* How is the feasibility of the model? i.e., loss value change with epoch (it means the number of complete run of training set data in the algorithm.) and single-layer token v.s. multi-layer token transaction network.
- *RQ3* How different parameters (different dimension sizes and different multi-heads attention sizes) affect the model?

Implementation details

Baselines

To illustrate the effectiveness of our methods, we compare our proposed MT^{2}AD with some competing methods:

- MLP [39]: Multi-layer Perceptron (MLP) is a fully connected classification of feedforward artificial neural network (ANN). In this experiment, we use MLP to learn the deep feature in transaction network instead of graph convolutional network. This means that only the node attributes are considered in this setup, without the graph transaction structure subgraph.
- GCN [9]: GCN is a graph convolutional network to perform the aggregation operations between the current node and its neighbors. In this setup, the attention mechanism with multiple heads are not considered. It means that different neighbors have equal influence on the current node.
- FEATHER [29]: FEATHER is an algorithm for the representation learning of node-level and graph-level. It considers the neighborhood feature distributions. It is to calculate a specific variant of the characteristic functions defined on graph vertices to describe the distribution of vertex features at multiple scales. These characteristic functions are effectively to create node embeddings. The features extracted by this procedure are useful for machine learning tasks.
- Graph2vec [30]: Graph2vec is a first neural embedding method to learn representation of entire graph in an unsupervised manner. Graph2vec is similar to doc2vec method [40], which needs a corpus of graphs to learn representations and follows the doc2vec Skipgram [25] training process.
- GL2vec [31]: On the basis of Graph2vec, GL2vec takes the edge labels into consideration, and it utilizes the line graphs (edge-to-vertex dual graphs) of input graphs. The results of graph embedding is the concatenation of embedding of the input graph and its corresponding line graph.

Datasets

We first obtained the BNB, LNK, USDT token transaction records. As the labels of blockchain transaction network have been verified in [11]. These labels are generated according to the Blockchain events from Wikipedia. Therefore, the time span of BNB, LNK and USDT are same as the label's. The

Table 1 The statistics of evaluation datasets

Dataset	Nodes	Total edges	Attributes	Number of tokens
D1	17,914	78,610	4	3
D2	100	74,000	4	6
D3	90	415,011	4	5

end time of BNB, LNK and USDT are all May 2018. The start time of BNB, LNK and USDT are July 2017, September 2017 and November 2017, respectively. We use D1 to represent our own dataset. Besides, we also use Ethereum Token Networks (bytom, cybermiles, decentraland, tierion, vechain and zrx) and Ripple Currency Networks (JPY, USD, EUR, CCK and CNY) from [11][8] to evaluate the performance of different methods. The D2 and D3 are utilized to represent these two datasets, respectively. Because these two original datasets D2 and D3 only have transaction structure feature. Therefore, we also transform edge's transaction value and transaction flow direction information as nodes' attributes. The statistics of these three datasets are shown in Table 1.

Evaluation metrics

The same as article [5], we use metrics Precision, Recall, F-value to evaluate the performance of different methods on anomaly detection. Because F-beta score can adjust relative weight between Precision and Recall with β coefficient to achieve better results (Eq. 6).

$$\text{FbetaScore} = (1+\beta^2)\cdot\frac{\text{Precision}\cdot\text{Recall}}{\beta^2\cdot\text{Precision}+\text{Recall}} \tag{6}$$

Therefore, in this paper, we use metrics Fbeta–Macro and Fbeta-Weighted instead of $F1$-value to evaluate the performance of different methods on anomaly detection. The β value is 5, 7 and 7 for D1, D2 and D3 respectively.

Parameter settings

The graph embedding dimension d of all methods is set to 8. For attention mechanism, the size of multi-head attention is set to 2. In our model, we utilize one convolution layer to aggregate the information from its neighbors. Therefore, the convolution size is $F \times 4$. The learning rate and weight decay are set to 0.0005 and 1e−5, respectively. For GCN, the parameters were set from their official implementations. FEATHER, Graph2vec and GL2vec code implemented in karateclub.[9]

[8] https://github.com/tdagraphs/tdagraphs.

[9] https://github.com/benedekrozemberczki/karateclub.

Performance results (RQ1)

To answer RQ1, we compare the performance of all methods in the task of anomaly detection on multiple token transaction network. The results of anomaly detection are shown in Table 2. We have the following observations:

- Compared with the MLP method, the GCN has greatly improved the performance. The reason is that MLP only considered the node attributes extracted manually, which is not enough to describe the information of multiple token transaction network. GCN takes the node attributes and structure of transaction network simultaneously, and it performs the aggregation operations to integrate the information from neighbors. This demonstrates the advantage of graph convolutional network and the importance of considering the network structure. In transaction network, transaction addresses are anonymous, and the nodes themselves have no attributes. So the difference between nodes is mainly on the transaction structure and edge's attributes. In this paper, we transform edge's attributes (timestamp and transaction value) into node attributes for further taking advantage of the graph convolutional network to extract depth feature.
- In most cases, the performance of FEATHER is better than Graph2vec and GL2vec. The reason is that FEATHER considers the neighborhood feature when creating the node embeddings. The idea of taking the neighborhood feature information into consideration is consistent with ours. Compared with FEATHER and our method, our method can achieve better results. It demonstrates the advantage of our method with graph neural network technique.
- From the results of GCN method and our proposed approach, it shows that our method achieve excellent results. The reason is that we use multi-head attention mechanism to aggregate information from neighbors. It can capture the difference of influence among different neighbors.
- Compared with Graph2vec and GL2vec method, our method and GCN achieve better performance. This demonstrates the advantage of graph neural network comparing with skipgram when processing graph network data. Graph neural network is able to capture the node pairs' similarity from nodes' attributes and structure simultaneously.

Table 2 The results of anomaly detection

Token	D1				D2				D3			
	Precision	Recall	FbMacro	FbWeighted	Precision	Recall	FbMacro	FbWeighted	Precision	Recall	FbMacro	FbWeighted
MLP	0.6602	0.8125	0.4956	0.8054	0.7656	0.8750	0.4993	0.8738	0.8643	0.9297	0.4996	0.9290
GCN	0.7119	0.8438	0.4965	0.8378	0.8560	0.9219	0.7495	0.9211	0.8794	0.9375	0.4999	0.9374
FEATHER	0.8485	0.4590	0.4982	0.8455	0.8621	0.4630	0.4984	0.8593	**0.9664**	0.4915	0.4997	0.9658
Graph2vec	0.8407	0.2852	**0.5008**	0.2773	0.8189	0.5895	0.5647	0.5899	0.9069	0.5633	0.4611	0.5635
GL2vec	0.7013	0.1311	0.4235	0.1176	0.8950	0.1228	0.4379	0.1076	0.9007	0.5761	0.4196	0.5763
MT^{2}AD	**0.8789**	**0.9375**	0.4987	**0.9351**	**0.9395**	**0.9688**	**0.7497**	**0.9681**	0.9539	**0.9766**	**0.6250**	**0.9765**

Best values are in bold
Best values are underlined

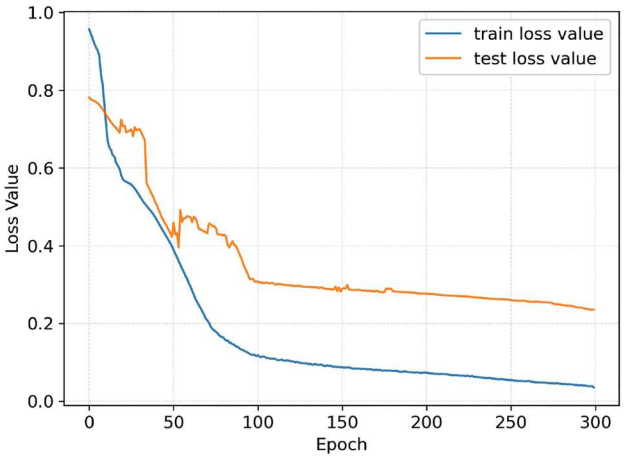

Fig. 5 Training and test loss value on D1 dataset

Table 3 Anomaly detection on single token v.s. multiple tokens

Token	D1			
	Precision	Recall	FbMacro	FbWeighted
BNB	0.7769	0.8797	0.4973	0.8751
LNK	0.7656	0.8750	0.4973	0.8702
USDT	0.6602	0.8125	0.4977	0.8088
BNB LNK USDT	0.8789	0.9375	0.4987	0.9351

Study of the proposed model (RQ2)

Loss change

To answer RQ2, the D1 dataset is taken as an example to explain the change of loss value on the training and test dataset, and the result is shown in Fig. 5. The x-axis represents the number of epoch and the y-axis represents the changes of loss value. It shows that the training loss value decreases constantly, and it gradually reaches a steady state. The test loss value also decreases during training the model. This demonstrates the effectiveness of our proposed method.

Multiple tokens

In this section, we take D1 dataset as an example to evaluate the anomaly detection results on single-layer and multi-layer transaction network, respectively. The results are shown in Table 3. We can observe that the performance on single token transaction network is unsatisfactory when comparing with overall multi-layer token transaction network (three tokens in here). The reason is that only single token is unable to model comprehensive information of transaction network, as there are cross-cryptocurrency trading patterns. MT^{2}AD can describe the cross-cryptocurrency trading patterns, and hence it improves the performance. Besides, we use the

same graph encoder on all token transaction networks to learn node embedding, this can not only capture each token respective characteristics through the convolution operation according to each token transaction network structure and node attributes, but also capture the common feature on all these multiple token transaction networks through the shared graph encoder.

Ablation experiment

In this section, we do the ablation experiment to test the transformed edge's feature as node attribute matrix. It means that we only consider the transaction structure information. That is to say, the input of the graph encoder is the adjacency matrix and the identity node attribute matrix. We use the identity node attribute matrix to replace the transformed node attribute matrix. In this case, the edge's feature (out-degree, in-degree, degree and normalized transaction amount) is not taken into consideration. We take the D1 dataset as an example to test the ablation experiment. The results are shown in Table 4. It shows that when the edge's feature is ignored, the performance is degraded. Because the identity node attribute matrix do not have any transaction flow and transaction amount information. This demonstrates that the transformed edge's feature as node attribute matrix can make the graph embedding retain transaction edge's information when performing the graph convolutional operation.

Sensitivity analysis (RQ3)

To answer RQ3, we further evaluate the performance of our proposed method in terms of different dimension size and multi-head attention size to see how these two hyperparameters affect the performance.

Effect of attention size

As for the attention size when aggregating the information from different neighbors, the performance changes of our model are shown in Fig. 6. We can observe that the performance of our method on these three datasets achieves different trends under different attention size. On D1 dataset, the FbMacro remains nearly constant, while on D2 and D3 dataset, the FbMacro achieves different values on different dimension sizes. The other three evaluation metrics on these three datasets achieve similar results. It shows that our method can still achieve good performance when the attention size is small (attention size is 2). This demonstrates the importance taking the neighborhoods information and its different influences into consideration.

Table 4 Ablation experiment about the edge's feature

Token	D1			
	Precision	Recall	FbMacro	FbWeighted
MT^2AD-identity	0.8213	0.9062	0.4980	0.9027
MT^2AD	0.8789	0.9375	0.4987	0.9351

Effect of embedding size

Embedding size is another hyper parameter that will affect the performance of our method. We vary the embedding size from 4 to 64 to investigate how it influences the graph classification about the anomaly detection on transaction network, and the results are shown in Fig. 7. It can be seen that high embedding size tends to overfitting, and hence the performance decreases. Besides, high embedding size will increase the computational cost. On the contrary, a low embedding size might be unable to capture the transaction network information. Hence, the performance is not good: for example on the results of D1 and D2 datasets, the performance about Precision metrics on a small embedding size ($d = 4$) is relatively worse than a large embedding size ($d = 8$). Therefore, we take the embedding size as 8 to balance the computational cost and performance.

Conclusion

In this paper, we proposed an anomaly detection model in multi-layer Ethereum transaction networks with graph representation learning technique and demonstrated its effectiveness through three research questions. We first showed the performance of anomaly detection in comparison with some baselines, which demonstrated the competitiveness of our proposed method. Then, we analyzed the loss value change with epoch and single-layer token versus multi-layer token transaction network to show the feasibility of our method. Finally, we analyzed the sensitivity to parameters on different embedding dimension sizes and different multi-head attention sizes. The key of the proposed model is that our method transforms the anomaly detection into a graph classification task, and model the edge's features as node's attributes, which can take advantage of the graph convolution network with attention mechanism integrating the transaction structure and node attributes naturally. Besides, our method can capture cross-cryptocurrency trading patterns on multiple token transaction networks through multi-layer token transaction network snapshot construction module and graph encoder module. These strategies improve the model's overall performance of anomaly detection.

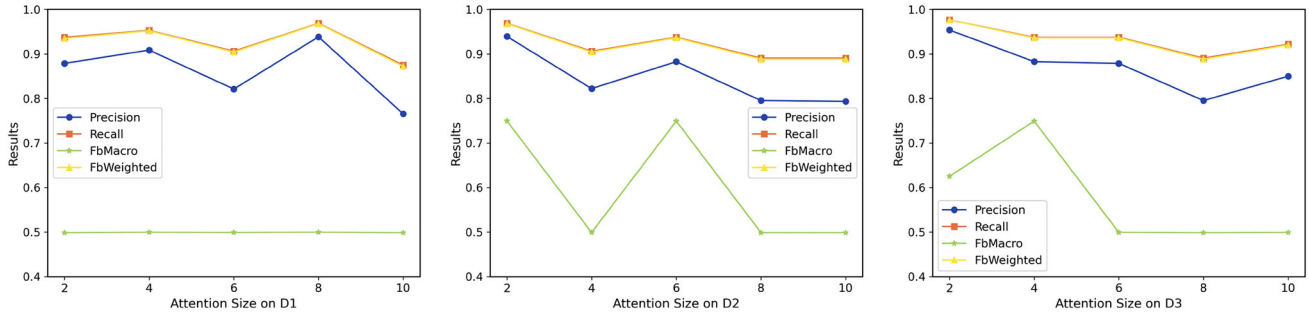

Fig. 6 Attention sizes

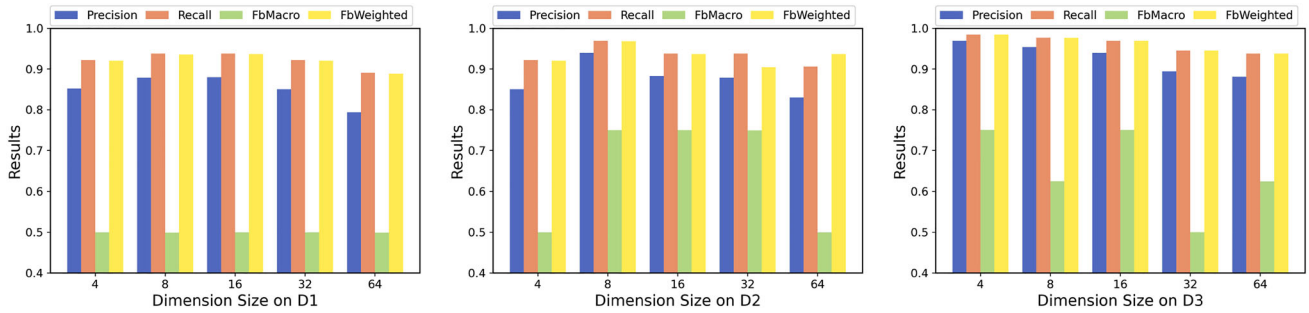

Fig. 7 Embedding sizes

As a next step, in the future, we will explore the unsupervised graph neural network method to analysis the Bitcoin transaction network. This is specifically important as labelling Bitcoin addresses is a fundamental problem to develop further the growing field of blockchain analytics.

Acknowledgements This paper is support by the National Natural Science Foundation of China (NSFC) under grant number 61873274.

Declaration

Conflict of interest No conflict of financial interests or personal relationships exit in the submission of this manuscript, and it is approved by all authors for publication.

References

1. Wang Z, Jin H, Dai W, Choo K-KR, Zou D (2021) Ethereum smart contract security research: survey and future research opportunities. Front Comp Sci 15:1–18
2. Chen H, Pendleton M, Njilla L, Xu S (2020) A survey on ethereum systems security: vulnerabilities, attacks, and defenses. ACM Comput Surv 53(3):1–43
3. Xu J, Livshits B (2019) The anatomy of a cryptocurrency pump-and-dump scheme. In: Proceedings of the 28th USENIX conference on security symposium. SEC'19. USENIX Association, USA, pp 1609–1625
4. Li S, Gou G, Liu C, Hou C, Li Z, Xiong G (2022) Ttagn: temporal transaction aggregation graph network for ethereum phishing scams detection. In: Proceedings of the ACM Web conference 2022. WWW '22. Association for Computing Machinery, New York, NY, USA, pp. 661–669
5. Chen L, Peng J, Liu Y, Li J, Xie F, Zheng Z (2020) Phishing scams detection in ethereum transaction network. ACM Trans Internet Technol 21(1):1–16
6. Wu J, Yuan Q, Lin D, You W, Chen W, Chen C, Zheng Z (2022) Who are the phishers? phishing scam detection on ethereum via network embedding. IEEE Trans Syst Man Cybernet Syst 52(2):1156–1166
7. Yousaf H, Kappos G, Meiklejohn S (2019) Tracing transactions across cryptocurrency ledgers. In: Proceedings of the 28th USENIX conference on security symposium. SEC'19. USENIX Association, USA, pp 837–850
8. De Collibus FM, Partida A, Piškorec M, Tessone CJ (2021) Heterogeneous preferential attachment in key ethereum-based cryptoassets. Front Phys 9:568
9. Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: 5th International conference on learning representations, ICLR 2017, Toulon, France, April 24–26, 2017, conference track proceedings

10. Vassilevska V, Williams R, Yuster R (2006) Finding the smallest h-subgraph in real weighted graphs and related problems. In: Bugliesi M, Preneel B, Sassone V, Wegener I (eds) Automata, languages and programming. Springer, Berlin, pp 262–273
11. Ofori-Boateng D, Dominguez IS, Akcora C, Kantarcioglu M, Gel YR (2021) Topological anomaly detection in dynamic multilayer blockchain networks. In: Oliver N, Pérez-Cruz F, Kramer S, Read J, Lozano JA (eds) Machine learning and knowledge discovery in databases research track. Springer, Cham, pp 788–804
12. Velickovic P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: 6th International conference on learning representations, ICLR
13. Grover A, Leskovec J (2016) Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '16. Association for Computing Machinery, New York, NY, USA, pp 855–864
14. Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: online learning of social representations. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp 701–710
15. Wei L, He Z, Zhao H, Yao Q (2023) Search to capture long-range dependency with stacking gnns for graph classification. In: Proceedings of the ACM Web conference 2023. WWW '23. Association for Computing Machinery, New York, NY, USA, pp 588–598
16. Zhang B, Luo S, Wang L, He D (2023) Rethinking the expressive power of GNNs via graph biconnectivity. In: The Eleventh international conference on learning representations
17. Chamberlain BP, Shirobokov S, Rossi E, Frasca F, Markovich T, Hammerla NY, Bronstein MM, Hansmire M (2023) Graph neural networks for link prediction with subgraph sketching. In: The eleventh international conference on learning representations
18. Zhou Y, Huo H, Hou Z, Bu F (2023) A deep graph convolutional neural network architecture for graph classification. PLoS One 18(3):e0279604
19. Hamilton WL, Ying R, Leskovec J (2017) Inductive representation learning on large graphs. In: Proceedings of the 31st international conference on neural information processing systems. NIPS'17. Curran Associates Inc., Red Hook, NY, USA, pp 1025–1035
20. Ying C, Cai T, Luo S, Zheng S, Ke G, He D, Shen Y, Liu T-Y (2021) Do transformers really perform badly for graph representation? In: Thirty-fifth conference on neural information processing systems
21. Chen J, Gao J, Chen Y, Oloulade BM, Lyu T, Li Z (2022) Autognas: a parallel graph neural architecture search framework. IEEE Trans Parall Distrib Syst 33:1–1
22. Hashemi F, Behrouz A, Hajidehi MR (2023) Cs-tgn: community search via temporal graph neural networks. arXiv preprint arXiv:2303.08964
23. Réau M, Renaud N, Xue LC, Bonvin AM (2023) Deeprank-gnn: a graph neural network framework to learn patterns in protein-protein interfaces. Bioinformatics 39(1):759
24. Xia Y, Liu J, Wu J (2022) Phishing detection on ethereum via attributed ego-graph embedding. IEEE Trans Circuits Syst II Express Briefs 69(5):2538–2542
25. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th international conference on neural information processing systems—volume 2. NIPS'13, Red Hook, NY, USA, pp 3111–3119
26. Wang J, Chen P, Yu S, Xuan Q (2021) Tsgn: transaction subgraph networks for identifying ethereum phishing accounts. In: International conference on blockchain and trustworthy systems. Springer, pp 187–200
27. Ma X, Wu J, Xue S, Yang J, Zhou C, Sheng Q, Xiong H, Akoglu L (2021) A comprehensive survey on graph anomaly detection with deep learning. IEEE Trans Knowl Data Eng:1–1
28. Nikolentzos G, Siglidis G, Vazirgiannis M (2022) Graph kernels: a survey. J Artif Intell Res 72:943–1027
29. Rozemberczki B, Sarkar R (2020) Characteristic functions on graphs: birds of a feather, from statistical descriptors to parametric models. In: Proceedings of the 29th ACM international conference on information & knowledge management
30. Narayanan A, Chandramohan M, Venkatesan R, Chen L, Liu Y, Jaiswal S (2017) Graph2vec: learning distributed representations of graphs. In: 13th International workshop on mining and learning with graphs (MLGWorkshop 2017)
31. Chen H, Koga H (2019) Gl2vec: graph embedding enriched by line graphs with edge features. In: Gedeon T, Wong KW, Lee M (eds) Neural information processing. Springer, Cham, pp 3–14
32. Ying R, You J, Morris C, Ren X, Hamilton WL, Leskovec J (2018) Hierarchical graph representation learning with differentiable pooling. In: Proceedings of the 32nd international conference on neural information processing systems. NIPS'18. Curran Associates Inc., Red Hook, NY, USA, pp 4805–4815
33. Lee J, Lee I, Kang J (2019) Self-attention graph pooling. In: Proceedings of the 36th international conference on machine learning, pp 3734–3743
34. Zhao H, Zhang C (2021) Gau-nets: graph attention u-nets for image classification. J Phys Conf Ser 1861:012045
35. Gao H, Ji S (2022) Graph u-nets. IEEE Trans Pattern Anal Mach Intell 44(9):4948–4960
36. Zhaohui W, Huawei S, Qi Cao XC (2022) Survey on graph classification. J Softw 33:171–192
37. Boccaletti S, Latora V, Moreno Y, Chavez M, Hwang D-U (2006) Complex networks: structure and dynamics. Phys Rep 424(4):175–308
38. Mesquita D, Souza A, Kaski S (2020) Rethinking pooling in graph neural networks. Adv Neural Inf Process Syst 33:2220–2231
39. von der Malsburg C (1986) Frank rosenblatt: Principles of neurodynamics: perceptrons and the theory of brain mechanisms. Brain Theory:245–248
40. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: Proceedings of the 31st international conference on international conference on machine learning—volume 32. ICML'14