Journal of the Royal Statistical Society Series A: Statistics in Society, 2025, **00**, 1–20
https://doi.org/10.1093/jrsssa/qnaf135

ROYAL STATISTICAL SOCIETY
DATA | EVIDENCE | DECISIONS
Journal of the Statistics Society
Series A
Statistics in Society

Original Article

Multilayer topology-aware graph contrastive learning for fraud detection in the Ethereum transaction network

Yuzhou Chen[1], Yuanyuan Zhang[2], Stephen Chan[3], Jeffrey Chu[4] and Nicholas Lord[2]

[1]Department of Statistics, University of California, 900 University Ave., Riverside, CA 92521, USA
[2]Center for Digital Trust and Society, Department of Criminology, University of Manchester, Oxford Road, Manchester, M13 9PL, UK
[3]Department of Mathematics and Statistics, American University of Sharjah, PO Box 26666, Sharjah, UAE
[4]Center for Applied Statistics, School of Statistics, Renmin University of China, No. 59 Zhongguancun Street, Haidian, Beijing 100872, China

Address for correspondence: Jeffrey Chu, Center for Applied Statistics, School of Statistics, Renmin University of China, No. 59 Zhongguancun Street, Haidian, Beijing 100872, China. Email: jeffrey.jchu@ruc.edu.cn

Abstract

Fraud detection in blockchain networks presents unique challenges due to the decentralized and pseudonymous nature of transactions. This study introduces a novel Multilayer Topology-Aware Graph Contrastive Learning (MTGCL) framework to detect fraudulent activity within the Ethereum transaction network. The proposed approach leverages node-level and topology-level representations, integrating persistent homology to capture high-order structural patterns and enhance anomaly detection. By employing adaptive graph augmentation and self-supervised contrastive learning, MTGCL effectively improves fraud detection performance. Empirical evaluations demonstrate that MTGCL outperforms existing graph contrastive learning models in classification accuracy across multiple time periods while maintaining competitive computational efficiency. The framework also exhibits scalability for large-scale blockchain analysis, achieving lower computational costs compared with other baselines methods. These findings highlight MTGCL's potential for real-world applications, offering valuable insights for financial institutions, cryptocurrency exchanges, regulatory bodies, and blockchain analytics firms in combating fraudulent activities and enhancing anti-money laundering compliance.

Keywords: blockchain, contrastive learning, Ethereum, graph learning, topological data analysis

1 Introduction

Over the past 15 years, digital assets and transactions have experienced severe disruption from the development of emerging technologies such as blockchain and digital cryptocurrencies. In particular, domestic and global finance is becoming increasingly intertwined with blockchain-based systems, with such technology playing a key role in financial markets (Zheng et al., 2017).

The use of blockchain brings benefits, such as increased efficiency, transparency, and immutability to transactions, however, these systems can still suffer from a number of key issues. Examples include fraudulent or unauthorized transactions that become permanent, data corruption leading to errors and inconsistencies in blockchain records, and network attacks leading to the collapse of blockchain networks, to name but a few. Although blockchain-based cryptocurrencies are not generally considered to be a mainstream payment method, they are becoming an extremely common method for engaging in fraudulent financial activity, with over $1 billion USD reported

Received: February 28, 2025. **Revised:** July 8, 2025. **Accepted:** August 14, 2025

to have been lost between 2021 and 2022 to related scams (Nguyen & Fletcher, 2022). In particular, phishing presents a significant threat in this fast-moving environment, with the Ethereum blockchain platform being a prime target due to its prominence and unique features such as smart contract and decentralized application (dApps) support.

Moreover, whilst traditional financial systems have centralized controls and monitoring systems to detect and prevent fraud, the open and pseudonymous characteristics of blockchain technology complicate these efforts. The lack of a central governing body that flags and stops fraud before it happens, and the immutability of cryptocurrency transactions mean that once the money is gone it is unlikely to be recovered (AAG, 2025). Combined with the social media presence and buzz around cryptocurrencies, and the result is a 'combustible combination' (Nguyen & Fletcher, 2022) for fraud, with almost half of all cryptocurrency scam victims acknowledging that social media was the source (Nguyen & Fletcher, 2022). Given the potential and popularity of these systems, it is increasingly critical to investigate anomalies and fraud resulting from malicious behaviour, in order to reduce the severity of their impacts.

Although blockchain-based systems possess complex structures and contain a huge volume of data, there are numerous paths for modelling and analysis. One convenient method that is frequently used is to represent and model blockchains in a simple way as large graphs or networks. The simplest model of a blockchain can be considered an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which is defined in terms of a set of nodes or vertices V, and edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ connecting pairs of nodes (Boginski et al., 2005), where nodes often represent users or accounts, and an edge may represent a transaction of data or monetary value. In general, the two key building blocks of blockchains are addresses and transactions, with the most common network representations being: (i) transaction graphs—where nodes are blocks of transactions and weighted edges indicate the transfer of an asset; (ii) address graphs—where nodes are user addresses and weighted edges indicate a transaction; (iii) entity graphs—where nodes are a cluster of addresses belonging to a single entity and weighted edges indicate a transaction, with weights reflecting data, information, value, etc.

1.1 Related works

1.1.1 Network models for blockchains

Indeed, the majority of the related network modelling literature is centred around the relationships between addresses and transactions, respectively, in the Bitcoin and Ethereum blockchains, using undirected graphs (Y. Chen et al., 2021; Y. Chen, Gel et al., 2022; Y. Chen, Segovia Dominguez et al., 2022; Lee et al., 2020), weighted graphs (W. Chen et al., 2020; Y. Li et al., 2020), subgraphs (Akcora et al., 2018; Y. Li et al., 2020), and bipartite graphs (Jourdan et al., 2018). More recent studies have investigated the common properties of blockchain-based systems, which have been found to exhibit similarities with real-world networks, for example disassortativity, small-world phenomena, and power-law degree distributions (Guidi et al., 2020; Lee et al., 2020; Y. Li et al., 2020; Nerurkar et al., 2021; Sharma et al., 2022; Tao et al., 2022). However, a challenge in modelling blockchain-based networks is that the number of nodes and edges can fluctuate significantly over time (Y. Li et al., 2020), highlighting the importance of temporal analyses (Nerurkar et al., 2021; Zhao et al., 2021).

1.1.2 Fraud detection in blockchain networks

From an anomaly detection perspective, network representations of blockchain-based networks are complex due to their unique properties. This can be attributed to blockchain networks generating significantly greater volumes of data compared with other network types, as data may relate to the network structure and features, but may also be specific to the type of transaction. As a result, many assumptions cannot be applied directly from studies on traditional (i.e. financial) networks, as the structures and behaviours can be very different.

At a global level, the detection problem relates to searching for any type of anomaly and identifying abnormal users and transactions in the address and transaction graphs. An indirect method has been proposed for reconstructing user behaviour through identifying the type of entity (de-anonymization). Traditional supervised machine learning methods such as k-nearest neighbours and random forests, and deep learning methods such as recurrent neural networks (RNNs) have been applied extensively for classifying and predicting Bitcoin entity types based

on address and transaction networks (Shao et al., 2018; Sun Yin et al., 2019; Sun Yin & Vatrapu, 2017). Gradient boosting has time and again been shown to give the best performance with an accuracy of up to 90% when using additional features beyond just address features (Jourdan et al., 2018). De-anonymization has also been extended by studies such as Juhász et al. (2018) to link transactions to IP addresses, determine geographical locations, and track the distribution and flow of Bitcoin. However, de-anonymization alone cannot fully confirm whether addresses are truly anomalous.

More direct methods involve seeking out anomalous or fraudulent users, addresses, and transactions from raw features. Early studies applied relatively simple distance-based measures, such as the Mahalanobis distance, to determine the distance between features of the address and transaction graphs in the Bitcoin network; the power law model to model the distribution of network features; local outliers based on rankings of distance from local clusters of data. While these studies were relatively simple, they confirmed that anomalies in blockchain-based networks are generally associated with extreme values.

K-means and its variations have also received consistent attention in anomaly detection in the Bitcoin network, being applied to address, transaction, and network features (Monamo et al., 2016; Morishima & Matsutani, 2018; Shayegan et al., 2022). Performance has been shown to be satisfactory, however, *k*-means may not be as powerful in detecting anomalies as other methods (Monamo et al., 2016). As a result, it is often used as a baseline measure. Recently, more complex methods have been implemented including support vector machines (SVM), singular value decomposition (SVD) (W. Chen, Wu et al., 2019), random forests (Baek et al., 2019), and deep learning. Though these studies have attempted to detect anomalous transactions in both the Bitcoin and Ethereum networks, and found that abnormal transactions may result from illegal activities, and policy changes or industry events, many common methods do not appear to show any overlap, suggesting that there is no single best-performing method for anomaly detection.

In contrast, more targeted methods focus on detecting particular types of anomalies in specific contexts, and relate mainly to phishing scams, Ponzi schemes, and pump-and-dump schemes. The one-class support vector machine (OCSVM) has been shown to perform better than common classification methods such as the naïve Bayes and logistic regression classifiers in detecting phishing transactions and addresses in the Ethereum network (Wu et al., 2021). Moreover, the inclusion of temporal features further improves performance, supporting the case for modelling blockchain-based networks dynamically. Supervised classification methods have been used in detecting Ponzi schemes in the Bitcoin and Ethereum networks (Bartoletti et al., 2018; W. Chen, Xu et al., 2019; W. Chen, Zheng et al., 2019; Toyoda et al., 2019). Random forests and gradient boosting have been shown to give the best performance with a false positive rate of around 5% (Toyoda et al., 2019). In terms of pump-and-dump schemes, studies have similarly applied supervised classification methods (random forests, generalized linear models, Apriori algorithm) and found that digital currencies can be predicted based on market information, and users involved in such schemes are linked to abnormal transactional behaviour (W. Chen, Zheng et al., 2019; Xu & Livshits, 2018).

1.1.3 Graph representation learning and graph contrastive learning

Recently, inspired by the success of convolutional neural networks (CNN) on image-based tasks, graph neural networks (GNNs) have emerged as a powerful tool for graph representation learning. Based on the spectral graph theory, Bruna et al. (2014) introduces a graph-based convolution in the Fourier domain. ChebNet (Defferrard et al., 2016) approximates the graph convolutions using Chebyshev polynomials. Graph convolutional networks (GCNs) of Kipf and Welling (2017) simplify the graph convolution with a localized first-order approximation. ARMA (Bianchi et al., 2021) aims to learn the rational graph convolutions through the Auto-Regressive Moving Average filters family. More recently, there have been proposed various approaches based on accumulation of the graph information from a wider neighbourhood, using diffusion aggregation and random walks. Such higher-order methods include approximate personalized propagation of neural predictions (APPNP) (Klicpera et al., 2019), and higher-order graph convolutional architectures (MixHop) (Abu-El-Haija et al., 2019), and graph neural networks with Levy flights mechanism (Y. Chen et al., 2020). Despite the recent developments, the

above-mentioned GNN-based models mainly focus on supervised and semi-supervised settings and differ from our unsupervised representation learning scheme. Graph contrastive learning is a self-supervised learning approach to learn an encoder for extracting meaningful representations from unlabelled data. Existing methods mainly focus on local–local CL (Bielak et al., 2022; Zhu et al., 2021), global–local CL (Sun et al., 2019; Velickovic et al., 2018), and global–global CL (S. Li et al., 2022; You et al., 2020). For instance, graph contrastive coding (GCC) (Qiu et al., 2020) proposes a pre-training framework based on local–local CL which constructs multiple graph views by sampling subgraphs based on random walks. For global–local CL, the works Velickovic et al. (2018), Hassani and Khasahmadi (2020), and Asano et al. (2020) follow the InfoMax principle (Linsker, 1988) to maximize the mutual information (MI) between the representation of local features and global features. Moreover, another graph contrastive learning mode, i.e. global–global CL (Fang et al., 2022; You et al., 2020) studies the relationships between the global context representations of different samples, which performs better on graph-level tasks. However, all these methods fail to capture the local and global topology information. To overcome the difficulties, we propose a novel graph contrastive learning framework which is first mixup-based framework collaboratively leveraging both node-level and topology-level information from graphs, and pioneers a novel Multilayer Topology-Aware Graph Contrastive Learning (MTGCL) architecture to model cross-level representation learning.

In this study, we focus on the Ethereum network blockchain as it provides an interesting case study to investigate the problem of anomaly and fraud detection. While Bitcoin retains the crown as the original and largest blockchain-based cryptocurrency, its main use case arguably remains limited to the transfer of monetary value. In contrast, Ether plays the role of currency in the Ethereum blockchain, but the platform itself extends beyond simply supporting monetary transfers, to dApps, smart contracts, and more (T. Chen et al., 2020). It is therefore not unreasonable to draw a link between the wide-ranging utility of the Ethereum blockchain and its position as the number one decentralized platform target for scams and frauds (PhishStats, 2024).

The remainder of this article is organized as follows. In Section 2, we provide the notations and preliminaries on persistent homology necessary for the methodology. In Section 3, we describe the Ethereum blockchain dataset used in the experimental work, and provide a discussion of its structure and properties. Section 4 introduces the proposed graph contrastive learning methodology. In Section 5, we illustrate and evaluate the performance of the proposed methodology in comparison to state-of-the-art baselines. Section 6 discusses the broader implications of our empirical findings for society, as well as their relevance to other types of fraud and blockchain applications. Finally, in Section 7, we provide some concluding remarks and suggest a number of potential extensions to the analysis undertaken in this article.

2 Notations and preliminaries

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ be an attributed graph, where $\mathcal{V}$ is a set of nodes ($|\mathcal{V}| = N$), $\mathcal{E}$ is a set of edges, and $\boldsymbol{X} \in \mathbb{R}^{N \times F}$ is a node feature matrix (here F is the dimension of node features). Let $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ be a symmetric adjacency matrix such that a_{uv} if nodes u and v are connected and 0, otherwise. Furthermore, $\boldsymbol{D}$ represents the degree matrix with $d_{uu} = \sum_v a_{uv}$, corresponding to $\boldsymbol{A}$.

2.1 Preliminaries on persistent homology

PH is a subfield in computational topology whose main goal is to detect, track, and encode the evolution of shape patterns in the observed object along various user-selected geometric dimensions (Carlsson & Vejdemo-Johansson, 2021; Edelsbrunner et al., 2000; Zomorodian & Carlsson, 2005). These shape patterns represent topological properties such as connected components, loops, and, in general, n-dimensional 'holes', that is, the characteristics of the graph $\mathcal{G}$ that remain preserved at different resolutions under continuous transformations. By employing such a multi-resolution approach, PH addresses the intrinsic limitations of classical homology and allows for retrieving the latent shape properties of $\mathcal{G}$ which may play the essential role in a given learning task. The key approach here is to select some suitable scale parameters v and then to study changes in shape of $\mathcal{G}$ that occur as $\mathcal{G}$ evolves with respect to v. That is, we no longer study $\mathcal{G}$ as a single object but as a *filtration* $\mathcal{G}_{v_1} \subseteq \ldots \subseteq \mathcal{G}_{v_n} = \mathcal{G}$, induced by monotonic changes of v. To ensure that the process of pattern selection and counting is objective and efficient, we build an abstract

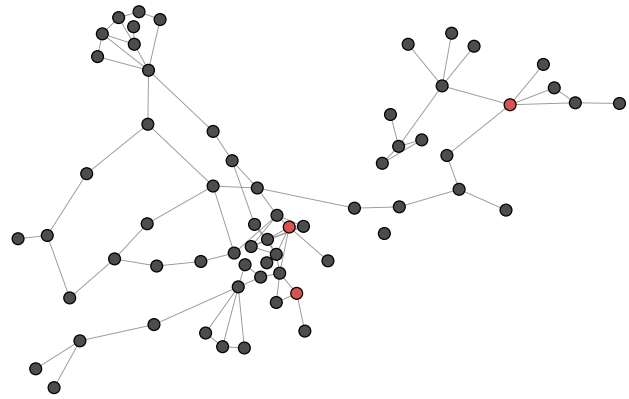

Figure 1. An illustration of a sample transaction network, where light coloured (dark coloured) dots represent labelled phishing (non-phishing) nodes.

simplicial complex $\mathcal{K}(\mathcal{G}_{v_j})$ on each $\mathcal{G}_{v_j}$, which results in a filtration of complexes $K(\mathcal{G}_{v_1}) \subseteq \ldots \subseteq K(\mathcal{G}_{v_n})$. For instance, for an edge-weighted graph $(\mathcal{V}, \mathcal{E}, w)$, with the edge-weight function $a : \mathcal{E} \to \mathbb{R}$, we can set $\mathcal{G}_{\leq v_j} = (\mathcal{V}, \mathcal{E}, a^{-1}(-\infty, v_j])$ for each v_j, $j = 1, \ldots, n$, yielding the induced sublevel edge-weighted filtration. Similarly, we can consider a function on a node set $\mathcal{V}$, for example, node degree, which results in a sequence of induced subgraphs of $\mathcal{G}$ with a maximal degree of v_j for each $j = 1, \ldots, n$ and the associated degree sublevel set filtration. We can then record scales b_i (birth) and d_i (death) at which each topological feature first and last appear in the sublevel filtration $\mathcal{G}_{v_1} \subseteq \mathcal{G}_{v_2} \subseteq \mathcal{G}_{v_3} \ldots$. However, in such sublevel filtration, some topological features may never disappear (i.e. persist forever), resulting in a loss of the important information on the underlying latent topological properties of $\mathcal{G}$ and, hence, making it more difficult to use the extracted topological information for shape matching among objects.

3 Data

The data analysed in this study is derived from Ethereum phishing transaction data obtained from XBlock[1], a publicly accessible blockchain database. The original data, provided in Pickle (PKL) format, originates from Etherscan—a leading Ethereum block explorer and analytics platform. The data analysed spans the period from 1st January 2018 to 19th January 2019 and contains 2,058,528 nodes and 8,541,503 edges. Among these, 1,165 nodes are labelled as phishing addresses with the remaining 2,057,363 nodes labelled as non-phishing. Transactions are considered to be phishing transactions if either party involved is a phishing address, and phishing transactions occur between 1st January 2018 and 19th January 2019. A sample transaction network, with phishing labels indicated, is presented in Figure 1.

The data contain six attributes that capture key information regarding blockchain transactions, including details of the sender and receiver addresses, their classification as phishing or non-phishing entities, the timestamp corresponding to the transaction mining process, and the value transacted. The attributes 'From' and 'To' are particularly important, as they signify whether the sender or receiver is associated with phishing activity. Specifically, 'From' takes a value of one if the sender is identified as a phishing entity and zero otherwise, and 'To' follows the same convention for the receiver. These binary indicators yield four possible transaction types[2] as shown in Table 1. To complement the numbers of transaction types, a summary of the overall address counts is also provided in Table 2.

As can be seen in Table 1, the highest transaction count is observed for transactions between non-phishing and non-phishing addresses, and the lowest for the phishing to phishing addresses. Additionally, as noted by Ghosh et al. (2023), the comparably smaller transaction count for $(1-1)$

[1] https://xblock.pro.

[2] $(0-0)$, $(0-1)$, $(1-0)$, and $(1-1)$.

Table 1. Summary of phishing/non-phishing transaction types

Transaction type	Number of transactions
(0 – 0)-Txns	15,000,000
(0 – 1)-Txns	36,194
(1 – 0)-Txns	23,544
(1 – 1)-Txns	261

Table 2. Summary of phishing/non-phishing addresses

Total addresses	Phishing	Non-phishing
2,058,528	1,165	2,057,363

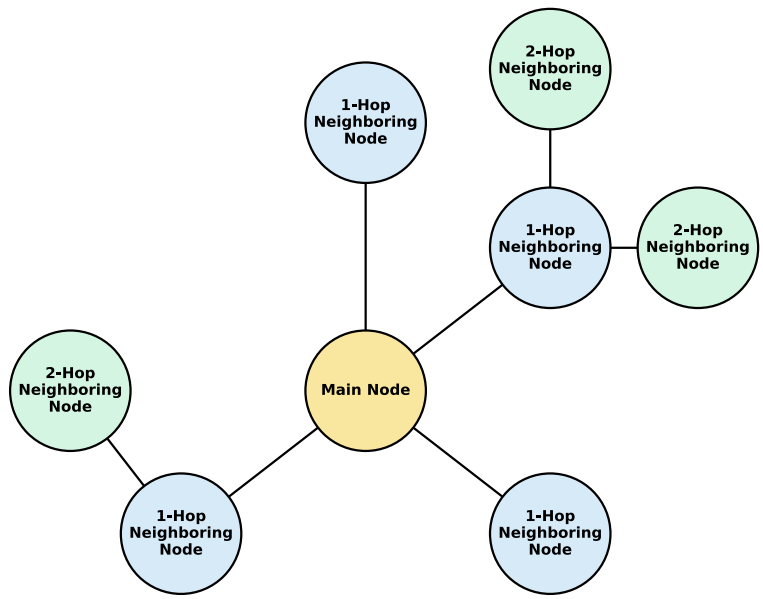

Figure 2. A simplified scheme of a transaction network node (centre, labelled 'Main Node') being connected to other 1-hop (labelled '1-Hop Neighboring Node') or 2-hop neighbours (connected to 1-hop neighbour, labelled '2-Hop Neighboring Node').

transactions than those for (0 – 1) and (1 – 0) transactions provides evidence for the 1-hop neighbourhood (see Figure 2 for a schematic representation of the concept of the 1-hop or 2-hop neighbourhood described in Tong et al., 2020) of the phishing nodes being mainly populated by non-phishing nodes.

A summary of common network properties computed for the monthly graphs from January 2018 to January 2019 is presented in Table 3. A clear declining trend in the number of nodes and edges is observed, dropping from 438,319 nodes and 1,297,388 edges in January 2018 to just 19,111 nodes and 19,589 edges by January 2019. Consequently, the average degree fluctuates throughout the year, peaking at 7.26 in March 2018, indicating greater connectivity during this period, before declining to 4.76 in January 2019. The standard deviation of degrees mirrors this trend, reflecting variability in the connectivity of nodes throughout the network. Centrality measures reveal notable shifts; for example, degree centrality and betweenness centrality peak in late 2018 at 0.28 and 0.26, respectively, suggesting increased influence of key nodes during

Table 3. Network properties of monthly graphs for January 2018 to January 2019

Network properties	January 2018	February 2018	March 2018	April 2018	May 2018	June 2018	July 2018	August 2018	September 2018	October 2018	November 2018	December 2018	January 2019
Number of nodes	438,319	427,585	313,798	263,111	381,018	360,888	271,140	170,471	90,869	80,218	58,746	31,886	19,111
Number of edges	1,299,738	1,305,524	1,118,113	908,709	1,120,561	1,005,791	615,876	436,677	225,332	216,848	141,776	102,760	43,798
Density	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Average degree	6.02	6.24	7.26	6.98	5.84	5.69	4.56	5.28	5.03	5.41	4.83	6.45	4.76
Std. dev. of degree	144.52	150.48	156.51	144.62	129.57	128.82	106.72	107.26	74.47	80.39	68.76	89.54	36.65
Degree centrality	0.06	0.07	0.03	0.04	0.03	0.04	0.04	0.06	0.12	0.11	0.15	0.28	0.10
Betweenness centrality	0.08	0.04	0.06	0.08	0.05	0.05	0.07	0.10	0.22	0.12	0.26	0.12	0.13
Closeness centrality	1.55	NaN	1.57	NaN	NaN	1.56	1.57	NaN	1.59	NaN	1.59	NaN	NaN
Eigenvector centrality	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Avg. geodesic path length	4.52	4.56	4.73	4.79	4.59	4.67	4.76	5.05	5.27	5.36	5.57	6.06	6.62
Diameter	14	14	16	15	16	14	14	18	14	15	19	20	18
Avg. clustering coefficient	0.04	0.04	0.05	0.05	0.05	0.09	0.06	0.07	0.09	0.04	0.05	0.06	0.06
Global clustering coefficient	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Adhesion	1	2	1	2	1	12	3	8	1	1	1	1	3
Cohesion	1	2	1	1	1	6	2	5	1	1	1	1	2
Degree correlation coefficient	−0.20	−0.16	0.24	0.27	−0.21	0.13	−0.18	0.34	0.24	0.53	0.06	0.92	0.42
Number of connected components	814	699	743	708	661	730	668	613	623	576	585	567	547
Number of triangles	46,362	79,278	45,210	53,343	55,929	65,493	21,720	20,016	13,197	10,152	3,225	1,956	981

this period. Interestingly, we note that for some monthly graphs the closeness centrality cannot be fully computed. We attribute this to the underlying structures of the graphs, which may only be weakly connected impacting shortest path calculations, as supported by low measures of connectedness such as density, degree centrality, clustering coefficient, etc. Despite fluctuations, eigenvector centrality remains consistently at 1.00, implying that the most central nodes retain their significance in the network hierarchy. Further examination of structural characteristics shows an increase in the average geodesic path length, increasing from 4.52 in January 2018 to 6.62 in January 2019, indicating a reduction in network cohesion over time. The diameter of the network also increases, reaching its maximum of 20 in December 2018, which aligns with the increased path length and reduced node connectivity. The clustering coefficients show slight variations, with the average clustering coefficient peaks at 0.09 in June and September 2018, suggesting periods of greater local connectivity. However, the global clustering coefficient remains at 0.00 throughout the year, highlighting the absence of global clustering tendencies. Notably, degree correlation coefficients become positive in the second half of 2018, peaking at 0.53 in September, indicating a tendency for nodes with similar degrees to connect. The number of connected components decreases overall but shows a slight increase towards the end of the year, possibly reflecting network fragmentation. Finally, the number of triangles, indicative of closed triads and local clustering, dramatically declines from 46,362 in January 2018 to just 981 in January 2019, showing the diminishing local cohesion within the network.

Tables 1 and 2 also highlight the significantly smaller volume of fraudulent addresses and the corresponding fraudulent transactions associated with these addresses. The limited number of labelled phishing accounts stems from the fact that labelling is mainly derived from public reporting. While the number of fraudulent accounts being flagged is increasing over time, the process is somewhat hindered by two main factors: (i) the decentralized nature of blockchains (and lack of standardized auditing procedures); (ii) the significant manpower and resources required for labelling (L. Chen et al., 2020). Therefore, in line with previous studies employing this dataset, we assume that all labelled phishing accounts are indeed fraudulent.

However, the rarity of fraudulent cases compared to non-fraudulent ones results in an imbalanced dataset, and the imbalance problem can deteriorate the performance of graph ML models in fraud detection tasks. More specifically, the problem of graph imbalance, in this study, is that the numbers of normal transactions and fraud transactions in the graph are unbalanced or even severely biased. For example, the monthly data in January 2018 consists of approximately 0.35% fraud transactions (i.e. 3,726 out of 1,048,576 fraud transactions). To address the above challenge, we apply the undersampling approach to reduce majority class samples (i.e. normal transactions), that is, we retain all minority class samples (i.e. fraud transactions) and randomly sample ρ% edges from the normal transaction edge set (where $\rho \in (0, 50]$). We then combine the existing minority class samples and sampled majority class samples. In this article, for each set of monthly data, we conduct undersampling 10 times which results in 10 different monthly subgraphs.

4 Methodology

In this section, we first present the proposed dynamic graph augmentation, and then bring forward the details of topological information extraction and multilayer topology-aware graph contrastive learning. The overview of the MTGCL framework is shown in Figure 3.

4.1 Dynamic graph augmentation

We employ a graph attention network with the multi-head self-attention mechanism to generate L dynamic graph augmentations with L adaptive graph structures, i.e. $\{\tilde{A}^{\ell}\}_{\ell=1}^{L}$. Compared to current graph augmentation methods, e.g. node/edge dropping, subgraph sampling, and feature masking, without requiring prior domain knowledge, our proposed dynamic graph augmentation enables us to recognize fundamental structures of the graph by leveraging both graph structural information and node features. Specifically, the ℓth dynamic augmentation view, denoted as $\mathcal{T}_{\alpha^{\ell}}(\mathcal{G})$, is modelled as a single-layer feedforward neural network and the attention coefficient between

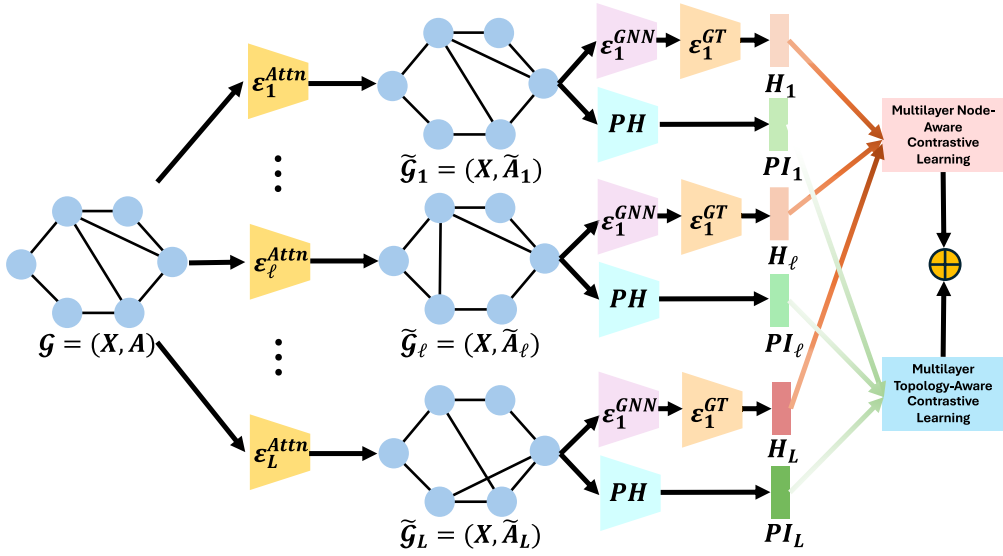

Figure 3. The overview of the Multilayer Topology-Aware Graph Contrastive Learning framework which is capable of performing unsupervised fraudulent node detection (where $\mathcal{E}_\ell^{\text{Attn}}$, $\mathcal{E}_\ell^{\text{GNN}}$, and $\mathcal{E}_\ell^{\text{GT}}$ denote the self-attention mechanism, pre-trained GNN, and graph Transformer of the ℓth layer.

two connected nodes u and v can be denoted as

$$\tilde{a}_{uv}^{\ell} = \frac{e^{\sigma(\boldsymbol{\alpha}^{\ell}[\boldsymbol{W}^{\ell}\boldsymbol{x}_u, \boldsymbol{W}^{\ell}\boldsymbol{x}_v])}}{\sum_{k \in \mathcal{N}_u} e^{\sigma(\boldsymbol{\alpha}^{\ell}[\boldsymbol{W}^{\ell}\boldsymbol{x}_u, \boldsymbol{W}^{\ell}\boldsymbol{x}_k])}}, \tag{1}$$

where $\tilde{a}_{uv}^{\ell}$ is set to 0 if pre-defined adjacency matrix $a_{uv} = 0$, $\boldsymbol{W}^{\ell}$ is a trainable weight matrix of the ℓth augmented view that projects node feature matrix to a lower-dimensional embedding space, $\boldsymbol{\alpha}^{\ell}$ is a trainable vector of the ℓth head, $[\cdot, \cdot]$ is the concatenation operation, and $\sigma(\cdot)$ represents the non-linear activation function. Compared to existing graph contrastive learning methods, our dynamic graph augmentation can automatically provide diverse contexts for nodes in different views. Additionally, traditional handcrafted graph augmentations often rely on random node or edge dropout to generate different views. However, our approach can be highly destructive, as removing key structural elements may severely distort the original graph topology—potentially compromising critical information that is essential for downstream tasks, ultimately leading to suboptimal embedding quality. In contrast, the dynamic graph augmentation method takes a fundamentally different approach by learning adaptive edge coefficients, rather than arbitrarily modifying the graph structure. This ensures that the augmentation process remains topology-aware and enables diverse augmented views while preserving the graph's intrinsic structure. As a result, it achieves safer and more effective graph augmentation, fostering diversity across views without introducing harmful distortions to the original topology.

4.2 Topological information extraction

To extract intrinsic high-order connectivity patterns within the graph (i.e. topological features), we propose a three-step topological information extraction: (i) define a q-hop subgraph for each node (where $q \geq 1$), (ii) compute the persistence diagrams of each subgraph via the PH, and (iii) determine the topological connection pattern for any pair of nodes based on the similarity between their corresponding persistence diagrams.

For each node, we aim to explore the topological features derived from their local connections (i.e. subgraphs). This approach offers twofold benefits. First, it focuses on local topological information, capturing the most relevant context of each node. Second, it facilitates the identification of

recurring patterns within similar substructures over the entire graph, which is beneficial for self-attention mechanisms. More specifically, we consider q-hop subgraphs centred at each node v, denoted by $\mathcal{G}_v^q = (\mathcal{V}_v^q, \mathcal{E}_v^q) \subseteq \mathcal{G}$ (here q is set to be 2). These subgraphs include the neighbours of the node v within a maximum distance of q-hops along the shortest paths. Once we identify the neighbouring nodes $\mathcal{V}_v^q$ connected to the node v, we can assign weights to the edges $\mathcal{E}_v^q$ based on the features exhibited by the neighbouring nodes. Let $\tilde{\mathcal{E}}_v^q$ be the weighted edge matrix, where the weight of the edge connecting two nodes u_i and u_j is computed as $\|X_{u_i} - X_{u_j}\|/f_0, \forall u_i, u_j \in \mathcal{V}_v^q$. Finally, we denote the weighted q-hop subgraph of the node v as $\bar{\mathcal{G}}_v^q = (\mathcal{V}_v^q, \bar{\mathcal{E}}_v^q)$.

Next, we propose the topology-induced connectivity learning module to learn connectivity information from the topology-based perspective. We adapt PH to capture local topological features for each node. Given a filtration function, we calculate a persistence diagram from the weighted q-hop subgraph of the node v, i.e. $\mathrm{Dg}_{\bar{\mathcal{G}}_v^q} = \mathrm{PH}(\bar{\mathcal{G}}_v^q)$. In this article, we leverage the Vietoris–Rips (VR)-complex to capture higher-order information for graph learning. By analysing $\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}$, we can understand the underlying topological features within the subgraph $\bar{\mathcal{G}}_v^q$ centred around node v. We discuss how to construct the topology-induced connectivity of the entire $\mathcal{G}$ based on $\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}$ as follows. The persistence diagram $\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}$ allows us to capture topological features of $\bar{\mathcal{G}}_v^q$ for the node v, and nodes which have similar persistence diagrams likely share similar underlying topological features.

Let $\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}$ and $\mathrm{Dg}_{\bar{\mathcal{G}}_u^q}$ be the persistence diagrams for subgraphs $\bar{\mathcal{G}}_v^q$ of the node v and $\bar{\mathcal{G}}_u^q$ of the node u, respectively. The distance between these persistence diagrams, denoted by $\mathcal{W}_p(\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}, \mathrm{Dg}_{\bar{\mathcal{G}}_u^q})$, is calculated as follows: $\mathcal{W}_p(\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}, \mathrm{Dg}_{\bar{\mathcal{G}}_u^q}) = \inf_{\gamma \in \Gamma} \left(\sum_{(x,y)\sim\gamma} \|x - y\|_\infty^p \right)^{1/p}$, where $1 \le p \le \infty$, and Γ refers to the set of all couplings in the two input persistence diagrams. In our work, we consider nodes to be similar based on their topological features if the Wasserstein distance $\mathcal{W}_p(\cdot, \cdot)$ between their persistence diagrams falls below a threshold value r (where $r \in [0, 1]$). Note that, since Wasserstein distance can be sensitive to noise and unstable when persistence diagrams differ significantly in cardinality, we consider (i) limiting the number of points in each persistence diagram by retaining only the most significant topological features, i.e. those with the largest persistence and (ii) using the sliced Wasserstein kernel for persistence diagrams (Carriere et al., 2017) which can reduce the dimensionality of the comparison using sliced projections and be more robust to cardinality differences between diagrams. Based on these similarities, we then can construct the topology-induced connectivity matrix $\boldsymbol{A}^{\mathrm{topo}} \in \mathbb{R}^{N \times N}$, which is formulated as follows

$$a_{vu}^{\mathrm{topo}} = \begin{cases} 1, & \text{if } 0 \le \mathcal{W}_p(\mathrm{Dg}_{\bar{\mathcal{G}}_v^q}, \mathrm{Dg}_{\bar{\mathcal{G}}_u^q}) < r, \\ 0, & \text{otherwise,} \end{cases} \quad \forall\, v, u \in \mathcal{V}. \tag{2}$$

4.3 Multilayer topology-aware graph contrastive learning

GCL aims to maximize the mutual information (MI) between different augmented views of a graph by contrasting positive and negative node pairs. Commonly used self-supervised contrastive losses, e.g. InfoNCE (Gutmann & Hyvärinen, 2010) and NT-Xent (T. Chen et al., 2020) have been widely adopted in node-level GCL methods (Veličković et al., 2018; You et al., 2020). However, these approaches typically consider only a single positive pair per anchor node and treat all other embeddings as negative pairs. Most GNN-based contrastive learning frameworks operate under the assumption of homophily, where connected nodes are expected to have similar representations; thus, traditional contrastive learning methods often push away the embeddings of neighbouring nodes which contradicts this assumption. To address this fundamental limitation, we introduce MTGCL which explicitly incorporates graph structural information, local topological information, and node features into the contrastive objective by redefining how positives and negatives are selected. That is, instead of considering only the same node of the anchor across views as a positive pair which focuses on node-level representation, our MTGCL expands the definition of positives as follows: (i) *Same node across views*: the most intuitive positive pair is the embedding of the same node in different augmented views which ensures consistency across perspectives; (ii) *Intra-view neighbours at the node-level and topology-level*: nodes connected within the same view

(i.e. 1-hop neighbourhood of the target node) should be treated as positives which reinforces structural consistency in the graph; and (iii) *Inter-view neighbours at the node-level and topology-level*: nodes that are neighbours in one view should also be considered positives across different views which maintains graph connectivity across augmentations. Note that, we consider two types of structural connectivity between a node and its intra-view and inter-view neighbours, i.e. (a) pre-defined graph structure $\boldsymbol{A}$ and (b) topology-aware graph structure $\boldsymbol{A}^{\text{topo}}$.

For the node-level contrastive learning, we first use a graph Transformer to encode the augmented graphs. The structure of the graph Transformer layer can be formalized as

$$\begin{aligned}\bar{\boldsymbol{H}}_G^{\ell} &= \text{Attn}^{\ell}(\text{LN}(\mathcal{E}_{\phi}^{\ell}(\boldsymbol{X}))) + \mathcal{E}_{\phi}^{\ell}(\boldsymbol{X}),\\ \boldsymbol{H}_G^{\ell} &= \text{FNN}^{\ell}(\text{LN}(\bar{\boldsymbol{H}}_G^{\ell})) + \bar{\boldsymbol{H}}_G^{\ell},\end{aligned} \tag{3}$$

where $\text{FFN}^{\ell}(\cdot)$ represents a position-wise feed-forward network with a single hidden layer and GeLU activation function for the ℓth layer, $\text{LN}(\cdot)$ denotes for layer normalization, $\mathcal{E}_{\phi}^{\ell}(\cdot)$ is a pre-trained graph encoder for the ℓth layer, and the self-attention module is implemented by

$$\text{Attn}^{\ell}(\text{LN}(\mathcal{E}_{\phi}^{\ell}(\boldsymbol{X}))) = \text{Softmax}\left(\frac{\delta_a(\boldsymbol{Q}^{\ell}(\boldsymbol{K}^{\ell})^{\top})}{\sqrt{d_h}}\right)\boldsymbol{V}^{\ell}, \tag{4}$$

where $\boldsymbol{Q}^{\ell} = \boldsymbol{W}_q^{\ell}(\text{LN}(\mathcal{E}_{\phi}^{\ell}(\boldsymbol{X})))$, $\boldsymbol{K}^{\ell} = \boldsymbol{W}_k^{\ell}(\text{LN}(\mathcal{E}_{\phi}^{\ell}(\boldsymbol{X})))$, $\boldsymbol{V}^{\ell} = \boldsymbol{W}_v^{\ell}(\text{LN}(\mathcal{E}_{\phi}^{\ell}(\boldsymbol{X})))$, $\boldsymbol{W}_q^{\ell}$, $\boldsymbol{W}_v^{\ell}$, and $\boldsymbol{W}_k^{\ell}$ are learnable weight parameters of size $\mathbb{R}^{d\times d_h}$, and δ_a is a dropout operation. The multilayer node-level contrastive loss between the layer ℓ and the layer ℓ' associated with the anchor $\boldsymbol{H}_u^{\ell}$ can be denoted as

$$\begin{aligned}\mathcal{L}(\boldsymbol{H}_u^{\ell}) = -\log\Bigg(&\frac{e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_u^{\ell'})/\tau_G}/(L|\mathcal{N}_u|+1)}{e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_u^{\ell'})/\tau_G} + \sum_{v_i\in\mathcal{V}}(e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G})}\\ &+\frac{\sum_{v_i\in\mathcal{N}_u}(e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G})/(L|\mathcal{N}_u|+1)}{e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_u^{\ell'})/\tau_G} + \sum_{v_i\in\mathcal{V}}(e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{H}_u^{\ell},\boldsymbol{H}_{v_i}^{\ell'})/\tau_G})}\Bigg),\end{aligned} \tag{5}$$

where $\boldsymbol{\theta}(\cdot)$ is a similarity function and τ_G denotes the temperature of the multilayer node-level contrastive learning (see Figure 4a). Similarly, the multilayer topology-level contrastive loss between the layer ℓ and the layer ℓ' associated with the anchor $\boldsymbol{Z}_u^{\ell}$ can be denoted as

$$\begin{aligned}\mathcal{L}(\boldsymbol{Z}_u^{\ell}) = -\log\Bigg(&\frac{e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_u^{\ell'})/\tau_T}/(L|\mathcal{N}_u|+1)}{e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_u^{\ell'})/\tau_T} + \sum_{v_i\in\mathcal{V}}(e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T})}\\ &+\frac{\sum_{v_i\in\mathcal{N}_u}(e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T})/(L|\mathcal{N}_u|+1)}{e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_u^{\ell'})/\tau_T} + \sum_{v_i\in\mathcal{V}}(e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T} + \sum_{\ell'\neq\ell}^{L} e^{\boldsymbol{\theta}(\boldsymbol{Z}_u^{\ell},\boldsymbol{Z}_{v_i}^{\ell'})/\tau_T})}\Bigg),\end{aligned} \tag{6}$$

where $\boldsymbol{Z}_u^{\ell} = \text{DeepSets}(\text{Dg}_{\tilde{\mathcal{G}}_u^q})$ is the topological embedding Zaheer et al. (2017), and τ_T denotes the temperature of the multilayer topology-level contrastive learning (see Figure 4b). Finally, the overall training objective of our framework can be denoted as

$$\mathcal{L} = \frac{1}{L}\sum_{\ell=1}^{L}\sum_{u\in\mathcal{V}}(\mathcal{L}(\boldsymbol{H}_u^{\ell}) + \mathcal{L}(\boldsymbol{Z}_u^{\ell})). \tag{7}$$

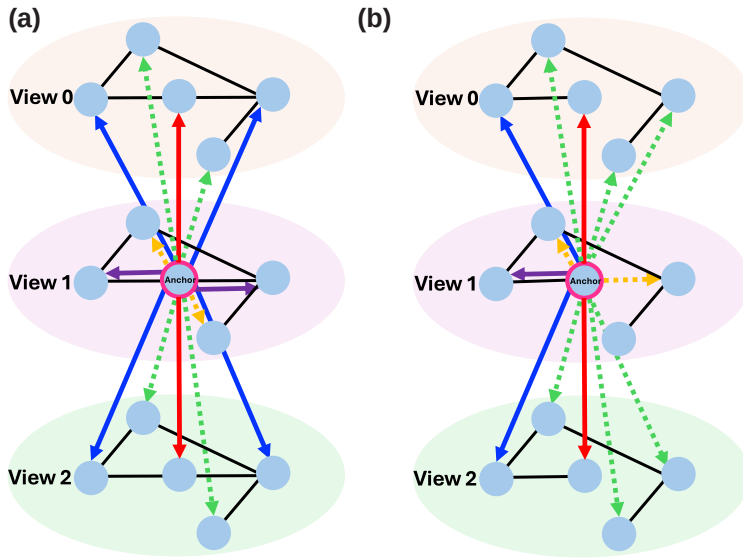

Figure 4. The overview of positive and negative pairs defined in the graph contrastive loss (a) and topology contrastive loss (b), where solid lines and dashed lines represent positive and negative pairs based on pre-defined graph structure and topology-aware graph structure, respectively.

5 Experiments

5.1 Baselines

For all graphs, we use 10-fold cross validation accuracy as the classification performance (based on a non-linear SVM model, i.e. LIB-SVM (Chang & Lin, 2011)) and repeat the experiments 10 times to report the mean and standard deviation. The best results are given in bold while the best performances achieved by the runner-ups are underlined. We evaluate the performances of our MTGCL on 13 graph datasets versus 6 state-of-the-art baselines including: (i) Deep Graph Infomax (DGI) (Veličković et al., 2018), (ii) Multi-View Graph Representation Learning (MVGRL) (Hassani & Khasahmadi, 2020), (iii) GRAph Contrastive Epresentation learning (GRACE) (Zhu et al., 2021), (iv) Graph Barlow Twins (GBT) (Bielak et al., 2022), (v) Bootstrapped Graph Latents (BGRL) (Thakoor et al., 2021), and (vi) Topological Graph Contrastive Learning (TopoGCL) (Y. Chen et al., 2024).

5.2 Experimental settings

We conduct our experiments on four NVIDIA RTX A5000 GPUs, each with 24 GB of memory. For each dataset, we use the Adam optimizer and perform a grid search to tune the learning rate from the set $\{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}, 1e^{-2}\}$, and weight decay factor from $\{0, 1e^{-6}, 1e^{-5}, 1e^{-4}\}$. We train the MTGCL model and all baselines for 200 epochs. The resolution of PI is fixed at 20×20, and take the number of layers L, the hidden dimension d, the number of layers of the multi-head self-attention mechanism #Layers, the dropout rate δ_a as hyperparameters, i.e. we search among $L \in \{3, 4, 5, 6\}$, $d \in \{16, 32, 64, 128\}$, #Layers $\in \{1, 2, 3\}$, and $\delta_a \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. We evaluate the fraud detection, i.e. node classification by using accuracy (in %).

5.3 Performance evaluation

The empirical performance is summarized in Table 4. Overall, from the table, we observe that our proposed MTGCL model shows competitive performance across all 13 datasets. Specifically, MTGCL consistently outperforms DGI, MVGRL, GBT, BFRL, and TopoGCL models. For instance, MTGCL has 3.76%, 3.42%, and 3.97% average relative improvement compared with DGI, MVGRL, and BGRL, respectively. Compared to TopoGCL, MTGCL also achieves

Table 4. Performance comparison on data from January 2018 to January 2019

Model	January 2018	February 2018	March 2018	April 2018	May 2018	June 2018	July 2018	August 2018	September 2018	October 2018	November 2018	December 2018	January 2019
DGI Velickovic et al. (2018)	95.82 ± 0.72	93.49 ± 1.07	95.31 ± 0.63	93.39 ± 0.67	89.61 ± 0.76	88.74 ± 1.24	91.11 ± 1.28	93.35 ± 0.77	89.52 ± 4.26	83.44 ± 4.21	87.51 ± 7.15	78.51 ± 9.06	78.61 ± 6.52
MVGRL Hassani and Khasahmadi (2020)	95.40 ± 1.24	94.39 ± 1.07	95.41 ± 0.25	93.49 ± 0.79	89.17 ± 0.68	87.23 ± 1.38	92.14 ± 0.95	93.41 ± 0.62	88.51 ± 6.10	77.53 ± 1.44	88.19 ± 5.40	83.92 ± 4.79	83.25 ± 6.57
GRACE Zhu et al. (2021)	95.50 ± 1.28	**94.44 ± 1.17**	94.99 ± 0.62	94.55 ± 0.48	90.60 ± 0.71	88.89 ± 1.08	88.71 ± 1.37	91.46 ± 1.38	85.94 ± 3.73	84.50 ± 2.59	87.57 ± 4.42	77.21 ± 6.66	87.87 ± 6.51
GBT Bielak et al. (2022)	95.16 ± 0.85	92.80 ± 0.84	94.97 ± 0.82	93.53 ± 0.54	89.69 ± 2.26	87.03 ± 1.00	91.17 ± 1.19	91.03 ± 1.05	88.49 ± 3.29	83.95 ± 1.72	87.55 ± 3.47	82.79 ± 6.92	86.01 ± 8.52
BGRL Thakoor et al. (2021)	94.91 ± 0.65	93.23 ± 0.71	95.40 ± 0.43	93.62 ± 0.61	89.34 ± 0.75	87.96 ± 1.31	89.68 ± 1.78	92.83 ± 2.01	87.73 ± 2.11	84.43 ± 9.13	86.91 ± 4.21	82.92 ± 5.03	76.88 ± 4.84
TopoGCL Y. Chen et al. (2024)	94.85 ± 0.72	93.81 ± 1.18	95.55 ± 0.59	92.37 ± 0.42	91.16 ± 1.42	85.81 ± 1.63	90.89 ± 0.96	91.39 ± 0.51	86.49 ± 3.23	82.02 ± 7.72	90.13 ± 5.29	80.61 ± 5.81	85.15 ± 6.68
MTGCL (Ours)	**96.43 ± 0.90**	93.91 ± 1.12	**95.94 ± 0.56**	**94.36 ± 0.61**	**93.29 ± 2.17**	**90.22 ± 0.80**	**93.98 ± 1.51**	**95.40 ± 0.43**	**91.31 ± 2.24**	**88.97 ± 4.65**	**92.54 ± 5.80**	**84.88 ± 5.52**	**88.47 ± 5.96**

Table 5. Performance comparison (in precision) on January 2018, June 2018, and December 2018 data

Model	January 2018	June 2018	December 2018
		Precision	
DGI Velickovic et al. (2018)	0.6071	0.4545	0.3810
MVGRL (Hassani & Khasahmadi, 2020)	0.4856	0.4474	0.4048
GRACE (Zhu et al., 2021)	0.4500	0.4524	0.3684
MTGCL (Ours)	**0.7250**	**0.4773**	**0.4286**

Table 6. Performance comparison (in recall) on January 2018, June 2018, and December 2018 data

Model	January 2018	June 2018	December 2018
		Recall	
DGI Velickovic et al. (2018)	0.5694	0.5000	0.3750
MVGRL (Hassani & Khasahmadi, 2020)	0.5000	0.4524	0.3950
GRACE (Zhu et al., 2021)	0.4762	0.4722	0.4412
MTGCL (Ours)	**0.7250**	**0.6667**	**0.4737**

significant performance gains in relative improvement. The strong performance verifies the superiority of the proposed contrastive learning framework by leveraging both node-level and topology-level information.

Several recent works (L. Chen et al., 2020) compare the performance of a GCN-based model to baselines of feature-only, DeepWalk, Node2vec, and LINE, using the subgraphs sampled from the same Ethereum dataset as in this study, and find that their proposed method consistently performs better than the baselines in terms of recall (up to 0.1735), and in smaller subgraphs, however, accuracy only reaches 58%. Additionally, W. Chen et al. (2021) compares SVM, decision trees, LightGBM, and dual sampling ensemble (DE) versions of the three algorithms, on Ethereum transaction data from 2016 to 2019. The DE LightGBM method is shown to give the best performance with precision and recall of approximately 0.82, and accuracy of approximately 82%. Researchers also compare SVM, KNN, and Adaboost on Ethereum transaction data and show that Adaboost is able to generate the best performance, with precision and recall reaching 0.83 and 0.66, respectively, and an accuracy peaking at 92% (Wen et al., 2021).

We have also reported performance comparison in both precision and recall. As shown in Tables 5 and 6, we observe that our MTGCL maintains consistently superior performance compared to baselines.

In comparison to these existing studies, we note that regardless of the Ethereum transaction dataset used (and its total sample size), in all cases the number of fraudulent accounts is similar to or less than that in our study. Furthermore, it appears that most other studies simply perform a 'static' analysis—i.e. obtaining one or several subgraphs or samples from the original transaction graph and then comparing the performances of various models. Therefore, our proposed MTGCL model and experiments not only show the performance advantage over the baselines but also show that our model generally matches and exceeds the accuracy of the best models from a number of recent studies. Furthermore, our experiments in essence include a temporal component by considering monthly subgraphs, which highlights the consistently high performance of the MTGCL model as the transaction data evolves over time.

5.4 Robustness analysis

In this article, we also explore the robustness of MTGCL with noise on January 2018, June 2018, and December 2018 data. More specifically, we add Gaussian noise into 10% of each data where

Table 7. Robustness analysis with the additive Gaussian noise $\mathcal{N}(0, 1)$

Model	January 2018	June 2018 $\mathcal{N}(0, 1)$	December 2018
DGI Velickovic et al. (2018)	93.79 ± 1.07	83.17 ± 1.50	72.61 ± 7.00
MVGRL (Hassani & Khasahmadi, 2020)	92.07 ± 0.99	84.31 ± 0.60	80.07 ± 4.43
GRACE (Zhu et al., 2021)	93.38 ± 0.85	85.04 ± 0.96	72.85 ± 6.96
MTGCL (Ours)	$\mathbf{95.33 \pm 0.79}$	$\mathbf{88.82 \pm 0.43}$	$\mathbf{83.68 \pm 5.31}$

Table 8. Robustness analysis with the additive Gaussian noise $\mathcal{N}(0, 2)$

Model	January 2018	June 2018 $\mathcal{N}(0, 2)$	December 2018
DGI Velickovic et al. (2018)	93.06 ± 1.31	82.52 ± 1.67	72.05 ± 6.79
MVGRL (Hassani & Khasahmadi, 2020)	91.35 ± 0.77	83.65 ± 0.56	79.44 ± 4.43
GRACE (Zhu et al., 2021)	92.65 ± 1.03	84.39 ± 1.26	72.28 ± 6.78
MTGCL (ours)	$\mathbf{94.59 \pm 0.69}$	$\mathbf{88.14 \pm 0.72}$	$\mathbf{83.03 \pm 3.29}$

Table 9. Sensitivity analysis with different threshold r

Threshold	January 2018	June 2018	December 2018
$r = 0.2$	91.72 ± 0.96	86.00 ± 0.74	81.09 ± 5.44
$r = 0.5$	$\mathbf{96.43 \pm 0.90}$	$\mathbf{90.22 \pm 0.80}$	$\mathbf{84.88 \pm 5.52}$
$r = 0.8$	94.61 ± 0.92	88.71 ± 0.87	83.64 ± 5.50

noise follows zero-mean i.i.d. Gaussian density with fixed variance σ^2, i.e. $\mathcal{N}(0, \sigma^2)$ where $\sigma^2 = \{1, 2\}$. Tables 7 and 8 depict the results for node classification with additive Gaussian noise $\mathcal{N}(0, 1)$ and $\mathcal{N}(0, 2)$, respectively. From both tables, we observe that (i) our MTGCL consistently outperforms all three baselines on all data and (ii) the relative performance decay of MTGCL is lower than baselines, e.g. for December 2018 (with $\mathcal{N}(0, \sigma^2)$), relative performance decay of MTGCL, DGI, MVGRL, and GRACE are 1.41%, 7.51%, 4.59%, and 5.65%, respectively. Thus, the minimal degradation of our MTGCL under noise highlights its stability and reliability for node classification tasks in noisy or uncertain environments.

5.5 Sensitivity analysis

The optimal choice of the threshold r can be obtained via cross-validation. Table 9 shows the results sensitivity analysis on January 2018, June 2018, and December 2018, respectively. The results show the threshold $r = 0.5$ always achieves the best node classification performance. Additionally, the performance under $r = 0.2$ and $r = 0.8$ remains relatively stable which shows that the method is not overly sensitive to moderate changes in the threshold. Despite slight variations, the high performance across all three threshold values suggests that our MTGCL generalizes well and does not require extensive tuning of r, which is advantageous in real-world deployments.

5.6 Time complexity analysis

The computational complexity of 0-dimensional PDs is $\mathcal{O}(|\mathcal{E}|\varphi(|\mathcal{E}|))$ where $\varphi(\cdot)$ denotes the inverse Ackermann function (Cormen et al., 2022). The computational complexity of multilayer node-level

and topology-level contrastive learning is $\mathcal{O}(N^2dL)$ and $\mathcal{O}(N^2L)$, respectively (where d is the embedding dimension of the pre-trained graph encoder), and the computational complexity of the graph Transformer is $\mathcal{O}(N^2)$. We also compare the running time (training time per epoch (in seconds)) between our MTGCL and two baselines (GRACE and TopoGCL). For instance, on January 2019 data, MTGCL: 10.31 s (88.47%), GRACE: 12.72 (87.87%), and 15.83 s (85.15%). That is, compared with baselines, MTGCL always achieves competitive fraud detection and computational cost.

6 Broader implications: societal impact, fraud diversity, and blockchain applications

Our empirical evaluation demonstrates that the proposed MTGCL framework substantially advances fraud detection accuracy within the Ethereum network, outperforming leading baseline methods across multiple temporal snapshots. This improvement is not only technical but also deeply societal in impact. Given the global movement towards digital economies and cashless societies, blockchain technology presents itself as a viable successor to the technology that underpins payments systems. However, compared with traditional centralized payments and transactions systems, for example credit cards, the framework for fraud detection in blockchain transactions is significantly less mature, robust, and standardized. Therefore, there is scope for the development of enhanced detection capabilities, which directly translates to better protection for individual users against financial loss, strengthening of anti-money laundering (AML) and regulatory compliance for institutions, and contribution to the restoration of public trust in blockchain technologies. By effectively identifying evolving fraudulent behaviours at scale and speed, the MTGCL framework provides actionable tools for exchanges, regulators, and law enforcement to safeguard digital assets and maintain integrity in rapidly growing decentralized financial ecosystems. The broader implication is a safer, more resilient digital economy that is better equipped to support innovation and protect society from emerging financial crime risks.

Although our analysis focuses on Ethereum phishing, the approach is not confined to just phishing fraud. Existing empirical studies, such as those by Bartoletti et al. (2018), Xu and Livshits (2018), and Luo et al. (2024), have demonstrated that different types of blockchain frauds, including Ponzi schemes, pump-and-dump activities, honey pot schemes, and rugpull scams, each exhibit distinctive structural patterns within their respective transaction networks. For example, with illegal pump-and-dumps, fraudulent addresses (nodes) commonly exhibit similar features such as pairs of transactions (a purchase and a sale) that occur very close together in time, where the value of the sale transaction becomes significantly greater than the value of the purchase transaction. Our proposed MTGCL method, particularly through the use of persistent homology, can flexibly adapt to identify these unique patterns by learning and incorporating relevant node-level and transaction-level structures, and by adjusting the methodology to various types of frauds. In other words, the focus is on normal versus fraudulent transactions (and addresses) and their structures, rather than the specific type of fraud. However, more sophisticated or one off attacks such as single instance exploits, insider attacks, or targeted thefts may not exhibit such pronounced or repetitive topological features, and instead may closely resemble legitimate user behaviour at the local network level, making them more difficult to detect. Moreover, transaction data that combines and includes multiple types of fraud may also increase detection complexity. While the current availability of such combined and labelled datasets is somewhat limited, this does offer a potential direction for future work to develop.

Finally, the Ethereum platform is arguably simultaneously different and similar to other major blockchain-based platforms, for example the Bitcoin blockchain. On the one hand, whereas traditional platforms such as Bitcoin were originally developed simply for transfers of monetary value, Ethereum possesses unique features such as the support for smart contracts and decentralized applications, etc. On the other hand, however, the fundamental structure of blockchain transactions, defined by address level interactions and transaction graphs, is a characteristic shared by most digital cryptocurrencies, including Bitcoin, Ripple, and numerous decentralized finance (DeFi) ecosystems (Liao et al., 2024; Zhang et al., 2024). Therefore, the core methodology underlying the MTGCL framework is broadly applicable to a variety of digital assets. Specifically, the MTGCL framework can be effectively applied to other blockchain-based systems, provided

that sufficient transaction data are available and the network topology is well defined. Moreover, the scalability and computational efficiency of the MTGCL framework, as demonstrated in our experiments, indicate its potential for large scale analysis in both established and emerging digital asset environments.

7 Conclusion

This study introduces a MTGCL framework for fraud detection within the Ethereum transaction network. By leveraging both node-level and topology-level representations, the proposed method effectively captures the complex structures and interactions within blockchain networks, enhancing fraud detection performance. The integration of persistent homology enables the extraction of topological features that contribute to improved representation learning, addressing key challenges in anomaly detection within dynamic and complex networks.

The empirical results demonstrate that MTGCL consistently outperforms existing graph contrastive learning models, exhibiting superior classification accuracy across multiple time periods. The inclusion of adaptive graph augmentation ensures that the learned representations are robust to variations in transaction patterns and network dynamics. Moreover, the framework effectively mitigates the impact of class imbalance in fraud detection tasks by refining the selection of positive and negative pairs in contrastive learning. The study also provides insights into the evolving structure of fraudulent transactions within the Ethereum network. The findings indicate that fraudulent activity is increasingly characterized by complex, adaptive behaviours that require multi-scale and topology-aware modelling approaches. The results further reinforce the importance of dynamic and self-supervised learning techniques in addressing the limitations of traditional supervised methods in blockchain fraud detection. In addition, MTGCL achieves high accuracy with competitive computational efficiency, as demonstrated in the time complexity analysis. Comparisons with baseline models show that MTGCL delivers comparable or superior fraud detection performance while maintaining lower computational costs. These findings confirm MTGCL's scalability and effectiveness for large-scale blockchain transaction analysis.

The results of this study provide significant practical value for financial institutions, cryptocurrency exchanges, regulatory bodies, law enforcement agencies, and blockchain analytics firms in detecting, preventing, and mitigating financial fraud in blockchain networks. The MTGCL framework strengthens fraud detection by capturing complex transaction patterns and network anomalies across multiple scales, supporting AML compliance and aiding regulators in identifying illicit activities. By integrating adaptive graph augmentation and topology-aware learning, the proposed approach offers a robust and scalable tool for securing digital assets and ensuring trust in blockchain ecosystems.

In future work, we propose to explore extensions of the MTGCL framework to other blockchain networks, including Bitcoin and DeFi ecosystems, to assess its generalizability. Additionally, the incorporation of temporal graph learning, reinforcement learning strategies, and incorporating network characteristics may further improve the adaptability of fraud detection models to rapidly evolving transaction patterns.

Acknowledgments

We are grateful to the editor and the reviewers for their constructive feedback and comments which greatly improved the article.

Conflicts of interest: None declared.

Funding

J.C. is supported by the National Natural Science Foundation of China (No. W2433186) and supported by the Beijing Natural Science Foundation (No. IS23126). S.C. is supported by the Faculty Research Grant (FRG23-C) from the American University of Sharjah. Y.Z. is supported by the University of Manchester Internationalisation: Global Scholars Fund (2024–2025), and the Centre for Digital Trust and Society Seed Corn Grant (2024–2025). Y.C. is not supported by any funds from China.

Data availability

The data and codes corresponding to the experiments can be found in the GitHub repository: https://github.com/yuzhouguangc/MTGCL.git.

References

AAG (2025). The latest phishing statistics (updated January 2025). https://aag-it.com/the-latest-phishing-statistics/.

Abu-El-Haija S., Perozzi B., Kapoor A., Alipourfard N., Lerman K., Harutyunyan H., Steeg G. V., & Galstyan A. (2019). Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *Proceedings of the International Conference on Machine Learning*. PMLR.

Akcora C. G., Dixon M. F., Gel Y. R., & Kantarcioglu M. (2018). Bitcoin risk modeling with blockchain graphs. *Economics Letters*, *173*(17), 138–142. https://doi.org/10.1016/j.econlet.2018.07.039

Asano Y., Rupprecht C., & Vedaldi A. (2020). Self-labelling via simultaneous clustering and representation learning. In *Proceedings of the International Conference on Learning Representations*. Curran Associates, Inc.

Baek H., Oh J., Kim C. Y., & Lee K. (2019). A model for detecting cryptocurrency transactions with discernible purpose. In *2019 Eleventh International Conference on Ubiquitous and Future Networks* (pp. 713–717). IEEE.

Bartoletti M., Pes B., & Serusi S. (2018). Data mining for detecting bitcoin Ponzi schemes. In *Crypto Valley Conference on Blockchain Technology* (pp. 75–84). IEEE.

Bianchi F. M., Grattarola D., Livi L., & Alippi C. (2021). Graph neural networks with convolutional ARMA filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3496–3507. https://doi.org/10.1109/TPAMI.2021.3054830

Bielak P., Kajdanowicz T., & Chawla N. V. (2022). Graph Barlow twins: A self-supervised representation learning framework for graphs. *Knowledge-based Systems*, *256*(4), 109631. https://doi.org/10.1016/j.knosys.2022.109631

Boginski V., Butenko S., & Pardalos P. M. (2005). Statistical analysis of financial networks. *Computational Statistics & Data Analysis*, *48*(2), 431–443. https://doi.org/10.1016/j.csda.2004.02.004

Bruna J., Zaremba W., Szlam A., & LeCun Y. (2014). Spectral networks and locally connected networks on graphs. In *Proceedings of the International conference on learning representations*. Curran Associates, Inc.

Carlsson G., & Vejdemo-Johansson M. (2021). *Topological data analysis with applications*. Cambridge University Press.

Carriere M., Cuturi M., & Oudot S. (2017). Sliced Wasserstein kernel for persistence diagrams. In *International conference on machine learning* (pp. 664–673). PMLR.

Chang C.-C., & Lin C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–27. https://doi.org/10.1145/1961189.1961199

Chen L., Peng J., Liu Y., Li J., Xie F., & Zheng Z. (2020). Phishing scams detection in ethereum transaction network. *ACM Transactions on Internet Technology*, *21*(1), 1–16. https://doi.org/10.1145/3398071

Chen T., Kornblith S., Norouzi M., & Hinton G. (2020). A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning* (pp. 1597–1607). PMLR.

Chen W., Guo X., Chen Z., Zheng Z., & Lu Y. (2021). Phishing scam detection on ethereum: Towards financial security for blockchain ecosystem. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'20*. IJCAI.

Chen W., Wu J., Zheng Z., Chen C., & Zhou Y. (2019). Market manipulation of bitcoin: Evidence from mining the mt. gox transaction network. In *IEEE Conference on Computer Communications* (pp. 964–972). IEEE.

Chen W., Xu Y., Zheng Z., Zhou Y., Yang J. E., & Bian J. (2019). Detecting "pump & dump schemes" on cryptocurrency market using an improved apriori algorithm. In *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)* (pp. 293–2935). IEEE.

Chen W., Zhang T., Chen Z., Zheng Z., & Lu Y. (2020). Traveling the token world: A graph analysis of ethereum erc20 token ecosystem. In *Proceedings of the Web Conference 2020* (pp. 1411–1421). ACM.

Chen W., Zheng Z., Ngai E. C.-H., Zheng P., & Zhou Y. (2019). Exploiting blockchain data to detect smart ponzi schemes on ethereum. *IEEE Access: Practical Innovations, Open Solutions*, *7*, 37575–37586. https://doi.org/10.1109/Access.6287639

Chen Y., Frias J., & Gel Y. R. (2024). TopoGCL: Topological graph contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 38, pp. 11453–11461). AAAI Press.

Chen Y., Gel Y., & Poor H. V. (2022). Time-conditioned dances with simplicial complexes: Zigzag filtration curve based supra-Hodge convolution networks for time-series forecasting. In *NIPS'21: Proceedings of the 36th International Conference on Neural Information Processing Systems* 35 (Vol. 650, pp. 8940–8953). Curran Associates, Inc.

Chen Y., Gel Y. R., & Avrachenkov K. (2020). LFGCN: Levitating over graphs with levy flights. In *IEEE International Conference on Data Mining* (pp. 960–965). IEEE.

Chen Y., Segovia I., & Gel Y. R. (2021). Z-GCNETs: Time zigzags at graph convolutional networks for time series forecasting. In *Proceedings of the International Conference on Machine Learning* (pp. 1684–1694). PMLR.
Chen Y., Segovia Dominguez I., Coskunuzer B., & Gel Y. (2022). TAMP-S2GCNETs: When time-aware multi-persistence meets spatio-supra graph convolutional nets while forecasting time series. In *Proceedings of the International Conference on Learning Representations*. Curran Associates, Inc.
Cormen T. H., Leiserson C. E., Rivest R. L., & Stein C. (2022). *Introduction to algorithms*. MIT Press.
Defferrard M., Bresson X., & Vandergheynst P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems* (Vol. 29, pp. 3844–3852). Curran Associates, Inc.
Edelsbrunner H., Letscher D., & Zomorodian A. (2000). Topological persistence and simplification. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science* (pp. 454–463). IEEE.
Fang Y., Zhang Q., Yang H., Zhuang X., Deng S., Zhang W., Qin M., Chen Z., Fan X., & Chen H. (2022). Molecular contrastive learning with chemical element knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, pp. 3968–3976). AAAI Press.
Ghosh M., Ghosh D., Halder R., & Chandra J. (2023). Investigating the impact of structural and temporal behaviors in ethereum phishing users detection. *Blockchain: Research and Applications*, *4*(4), 100153. https://doi.org/10.1016/j.bcra.2023.100153
Guidi B., Michienzi A., & Ricci L. (2020). Steem blockchain: Mining the inner structure of the graph. *IEEE Access: Practical Innovations, Open Solutions*, *8*, 210251–210266. https://doi.org/10.1109/Access.6287639
Gutmann M., & Hyvärinen A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 297–304). JMLR Workshop and Conference Proceedings.
Hassani K., & Khasahmadi A. H. (2020). Contrastive multi-view representation learning on graphs. In *Proceedings of the International Conference on Machine Learning* (pp. 4116–4126). PMLR.
Jourdan M., Blandin S., Wynter L., & Deshpande P. (2018). Characterizing entities in the bitcoin blockchain. In *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)* (pp. 55–62). IEEE.
Juhász P. L., Stéger J., Kondor D., & Vattay G. (2018). A Bayesian approach to identify bitcoin users. *PLoS One*, *13*(12), 1–21. https://doi.org/10.1371/journal.pone.0207000
Kipf T. N., & Welling M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*. Curran Associates, Inc.
Klicpera J., Bojchevski A., & Günnemann S. (2019). Predict then propagate: Graph neural networks meet personalized pagerank. In *Proceedings of the International Conference on Learning Representations*. Curran Associates, Inc.
Lee X. T., Khan A., Sen Gupta S., Ong Y. H., & Liu X. (2020). Measurements, analyses, and insights on the entire ethereum blockchain network. In *Proceedings of the Web Conference 2020* (pp. 155–166). ACM.
Li S., Wang X., Zhang A., Wu Y., He X., & Chua T.-S. (2022). Let invariant rationale discovery inspire graph contrastive learning. In *Proceedings of the International Conference on Machine Learning* (pp. 13052–13065). PMLR.
Li Y., Islambekov U., Akcora C., Smirnova E., Gel Y. R., & Kantarcioglu M. (2020). Dissecting ethereum blockchain analytics: What we learn from topology and geometry of ethereum graph. In *Proceedings of the 2020 SIAM International Conference on Data Mining* (pp. 523–531). SIAM.
Liao X., Li Q., Chan S., Chu J., & Zhang Y. (2024). Interconnections and contagion among cryptocurrencies, DeFi, NFT and traditional financial assets: Some new evidence from tail risk driven network. *Physica A: Statistical Mechanics and its Applications*, *647*(4), 129892. https://doi.org/10.1016/j.physa.2024.129892
Linsker R. (1988). Self-organization in a perceptual network. *Computer*, *21*(3), 105–117. https://doi.org/10.1109/2.36
Luo B., Zhang Z., Wang Q., Ke A., Lu S., & He B. (2024). AI-powered fraud detection in decentralized finance: A project life cycle perspective. *ACM Computing Surveys*, *57*(4), 1–38. https://doi.org/10.1145/3705296
Monamo P., Marivate V., & Twala B. (2016). Unsupervised learning for robust bitcoin fraud detection. In *Proceedings of 2016 Information Security for South Africa (ISSA)* (pp. 129–134). IEEE.
Morishima S., & Matsutani H. (2018). Acceleration of anomaly detection in blockchain using in-gpu cache. In *IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications* (pp. 244–251). IEEE.
Nerurkar P., Patel D., Busnel Y., Ludinard R., Kumari S., & Khan M. K. (2021). Dissecting bitcoin blockchain: Empirical analysis of bitcoin network (2009–2020). *Journal of Network and Computer Applications (Online)*, *177*(1), 102940. https://doi.org/10.1016/j.jnca.2020.102940
Nguyen S. T., & Fletcher E. (2022). Reports show scammers cashing in on crypto craze. *Federal Trade Commission*. https://www.ftc.gov/news-events/data-visualizations/data-spotlight/2022/06/reports-show-scammers-cashing-crypto-craze.

PhishStats (2024). Phishstats. https://phishstats.info/.
Qiu J., Chen Q., Dong Y., Zhang J., Yang H., Ding M., Wang K., & Tang J. (2020). GCC: Graph contrastive coding for graph neural network pre-training. In *KDD 20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1150–1160). ACM.
Shao W., Li H., Chen M., Jia C., Liu C., & Wang Z. (2018). Identifying bitcoin users using deep neural network. In *Algorithms and Architectures for Parallel Processing* (pp. 178–192). Springer.
Sharma A., Agrawal A., Bhatia A., & Tiwari K. (2022). Bitcoin's blockchain data analytics: A graph theoretic perspective. In *Advanced Information Networking and Applications AINA 2022* (Vol. 449, pp. 459–470). Springer.
Shayegan M. J., Sabor H. R., Uddin M., & Chen C. L. (2022). A collective anomaly detection technique to detect crypto wallet frauds on bitcoin network. *Symmetry*, *14*(2), 328. https://doi.org/10.3390/sym14020328
Sun F.-Y., Hoffman J., Verma V., & Tang J. (2019). Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *Proceedings of the International Conference on Learning Representations*. Curran Associates, Inc.
Sun Yin H., Langenheldt K., Harlev M., Mukkamala R. R., & Vatrapu R. (2019). Regulating cryptocurrencies: A supervised machine learning approach to de-anonymizing the bitcoin blockchain. *Journal of Management Information Systems*, *36*(1), 37–73. https://doi.org/10.1080/07421222.2018.1550550
Sun Yin H., & Vatrapu R. (2017). A first estimation of the proportion of cybercriminal entities in the bitcoin ecosystem using supervised machine learning. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 3690–3699). IEEE.
Tao B., Dai H.-N., Wu J., Ho I. W.-H., Zheng Z., & Cheang C. F. (2022). Complex network analysis of the bitcoin transaction network. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *69*(3), 1009–1013. https://doi.org/10.1109/TCSII.2021.3127952
Thakoor S., Tallec C., Azar M. G., Munos R., Veličković P., & Valko M. (2021). Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*. Curran Associates, Inc.
Tong X., Li X., & Liu Y. (2020). Research on resource efficiency optimization model of TDMA-based distributed wireless ad hoc networks. *IEEE Access: Practical Innovations, Open Solutions*, *8*, 96249–96260. https://doi.org/10.1109/Access.6287639
Toyoda K., Takis Mathiopoulos P., & Ohtsuki T. (2019). A novel methodology for hyip operators' bitcoin addresses identification. *IEEE Access: Practical Innovations, Open Solutions*, *7*, 74835–74848. https://doi.org/10.1109/Access.6287639
Veličković P., Fedus W., Hamilton W. L., Liò P., Bengio Y., & Hjelm R. D. (2018). Deep graph infomax. In *International Conference on Learning Representations*. Curran Associates, Inc.
Wen H., Fang J., Wu J., & Zheng Z. (2021). Transaction-based hidden strategies against general phishing detection framework on ethereum. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)* (pp. 1–5). IEEE.
Wu J., Liu J., Zhao Y., & Zheng Z. (2021). Analysis of cryptocurrency transactions from a network perspective: An overview. *Journal of Network and Computer Applications (Online)*, *190*(1), 103139. https://doi.org/10.1016/j.jnca.2021.103139
Xu J., & Livshits B. (2018). The anatomy of a cryptocurrency pump-and-dump scheme. In *Proceedings of the 28th USENIX Security Symposium* (pp. 1609–1625). USENIX Association.
You Y., Chen T., Sui Y., Chen T., Wang Z., & Shen Y. (2020). Graph contrastive learning with augmentations. In *NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vol. 33, pp. 5812–5823). Curran Associates, Inc.
Zaheer M., Kottur S., Ravanbakhsh S., Poczos B., Salakhutdinov R. R., & Smola A. J. (2017). Deep sets. In *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*. (Vol. 30, pp. 3394–3404). Curran Associates, Inc.
Zhang Y., Chan S., Chu J., Liao X., & Helu M. (2024). Stylized facts of decentralized finance (defi). In *Artificial Intelligence and Beyond for Finance* (pp. 289–314). World Scientific.
Zhao L., Sen Gupta S., Khan A., & Luo R. (2021). Temporal analysis of the entire ethereum blockchain network. In *Proceedings of the Web Conference 2021* (pp. 2258–2269). ACM.
Zheng Z., Xie S., Dai H., Chen X., & Wang H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE International Congress on Big Data (BigData Congress)* (pp. 557–564). IEEE.
Zhu Y., Xu Y., Yu F., Liu Q., Wu S., & Wang L. (2021). Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021* (pp. 2069–2080). ACM.
Zomorodian A., & Carlsson G. (2005). Computing persistent homology. *Discrete & Computational Geometry*, *33*(2), 249–274. https://doi.org/10.1007/s00454-004-1146-y