TSGN: Transaction Subgraph Networks for Identifying Ethereum Phishing Accounts

Jinhuan Wang[1,2], Pengtao Chen[1,2], Shanqing Yu[1,2], and Qi Xuan[1,2,3(✉)]

[1] Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou 310023, China
xuanqi@zjut.edu.cn

[2] College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China

[3] PCL Research Center of Networks and Communications, Peng Cheng Laboratory, Shenzhen 518000, China

Abstract. Blockchain technology and, in particular, blockchain-based transaction offers us information that has never been seen before in the financial world. In contrast to fiat currencies, transactions through virtual currencies like Bitcoin are completely public. And these transactions of cryptocurrencies are permanently recorded on Blockchain and are available at any time. Therefore, this allows us to build transaction networks (TN) to analyze illegal phenomenons such as phishing scams in blockchain from a network perspective. In this paper, we propose a Transaction SubGraph Network (TSGN) based classification model to identify phishing accounts in Ethereum. Firstly we extract transaction subgraphs for each address and then expand these subgraphs into corresponding TSGNs based on the different mapping mechanisms. We find that TSGNs can provide more potential information to benefit the identification of phishing accounts. Moreover, Directed-TSGNs, by introducing direction attributes, can retain the transaction flow information that captures the significant topological pattern of phishing scams. By comparing with the TSGN, Directed-TSGN indeed has much lower time complexity, benefiting the graph representation learning. Experimental results demonstrate that, combined with network representation algorithms, the TSGN model can capture more features to enhance the classification algorithm and improve phishing nodes' identification accuracy in the Ethereum networks.

Keywords: Ethereum · Phishing identification · Subgraph network · Network representation · Graph classification

1 Introduction

Blockchain is a distributed public ledger that is secured by blockchain technology. All transactions take place between two different public addresses and are

H.-N. Dai et al. (Eds.): BlockSys 2021, CCIS 1490, pp. 187–200, 2021.
https://doi.org/10.1007/978-981-16-7993-3_15

permanently recorded on a specific blockchain built for Bitcoin. The process of securing these transactions is handled by Bitcoin miners, who use their computing power to solve complex encryption problems and validate blocks and transactions in the process [13]. There is no limit to the number of Bitcoin addresses that any individual or organization can create, and there is no need to verify the identity during the process of creating an address. With the above advantages, blockchain technology has been rapidly developed and naturally introduced into the financial field. In the digital currency scenarios, the most widely used application of blockchain is cryptocurrency technology [21], by which accounts can freely and conveniently conduct transactions with currency and information and do not have to rely on traditional third parties.

It's worth noting that the cryptocurrency market inevitably breeds many cybercrimes due to anonymity and unsupervised organization. Similarly, as the second-largest cryptocurrency platform next to Bitcoin, Ethereum has been affected by many entities/accounts engaging in illegal activities over the network, including smart Ponzi schemes, phishing, money laundering, fraud, and criminal-related activities. It is reported that phishing scams can break out periodically and are the most deceptive form of fraud [4]. Although the hash mechanism set up inside the blockchain can prevent transactions from being tampered with, so far, there are no available internal tools that can detect illegal accounts and suspicious transactions on the network. Thus it can be seen that cybercrimes, especially phishing scams, have become a critical issue on Ethereum and should be worthy of long-term attention and research to adopt effective countermeasures.

Generally, phishing is a social engineering attack that aims to exploit weaknesses caused by users in the system processes [9]. In traditional phishing attacks, the terminal consumers will receive emails or text messages containing a malicious website whose hostname is close to the legitimate domain from a trusted entity in disguise. Once the link is clicked, phishers will use the measures provided in the link to obtain the users sensitive information, such as usernames, passwords, and credit card details. Thus, existing researches on detecting phishing scams mainly focus on the suspected phishing website identification [5,16] and phishing text massages detection [1,7]. Compared with traditional phishing scenarios, blockchain's openness and transparency make the suspicious phishing addresses and fraudulent funds reportable and traceable. Therefore, traditional forms of phishing scams are difficult to implement on the Ethereum platform on a large scale, and the corresponding detection schemes are not suitable to migrate to the Ethereum phishing detection problem.

In order to identify phishing addresses on Ethereum, we construct transaction networks by transaction information recorded permanently on the Ethereum. Each account is accessible and their transaction history can be available freely. In the transaction networks, the nodes represent Ethereum addresses, while the edges indicate the transaction records with some attributes. Generally, each record between Ethereum accounts includes information such as transaction direction, transaction amount, and transaction timestamp. In this paper, we

propose the TSGN model to identify phishing accounts. We think of transaction direction and transaction amount as the essential attributes to build transaction networks. Based on the above, we preproccess the weighted directed transaction networks and then map these networks to subgraph network structural space. According to different pre-processing and mapping strategies, we can obtain the corresponding TSGN and Directed-TSGN for the subsequent feature extraction and detection task. Specifically, our contributions can be concluded as follow:

- We propose a new transaction network model, transaction subgraph networks (TSGNs). Compared with original transaction networks, our TSGN can increase the diversity of features benefiting the subsequent network algorithms.
- We introduce different network mapping strategies to fully capture the potential structural topological information which can not be obtained easily from transaction networks.
- We build the problem of Ethereum phishing account identification as a graph classification task. Our TSGN model can be utilized to enhance various graph classification algorithms such as manual attributes, Graph2Vec, and Diffpool.
- We apply the new model to three transaction network datasets, and our experimental results demonstrate the effectiveness of TSGNs. The fusion of TN and TSGNs generated by different mapping strategies can increase the performance of graph classification algorithms. Directed-TSGN achieves the best performance in 7 of 9 cases. Especially, the classification result Directed-TSGN increases to 93.90% (93.25% for TSGN) when only Diffpool is considered, greatly improving the phishing account identification performance. More remarkably, compared with TSGN, generating Directed-TSGN needs much less time, reduced by almost one order of magnitude.

The rest of the paper is structured as follows. In Sect. 2, we make a brief description of the phishing identification and graph representation methods. In Sect. 3, we mainly introduce the definitions and construction methods of transaction subgraph networks. In Sect. 4, we give several feature extraction methods, which together with TSGN and Directed-TSGN are applied to three Ethereum transaction network datasets. Finally, we conclude our paper in Sect. 5.

2 Background and Related Work

In this section, to supply some necessary background information, we give a brief overview of phishing detection and graph representation algorithms in graph mining.

2.1 Phishing Identification

Phishing scams have become a major threat to the security of Ethereum transactions. To create a good investment environment in the Ethereum ecosystem,

many researchers have paid lots of attention to study the effective detection methods for phishing scams. Different from the privacy of traditional financial transaction information, the transaction records of the blockchain are freely available and contain rich attributes. Therefore, many recent studies are mainly based on transaction records. Wu et al. [18] proposed an approach to detect phishing scams on Ethereum by mining its transaction records. By considering the transaction amount and timestamp, this work introduced a novel network embedding algorithm called trans2vec to extract the features of the addresses for subsequent phishing identification. Chen et al. [4] proposed a detecting method based on Graph Convolutional Network and autoencoder to precisely distinguish phishing accounts. One can see that these methods mentioned above mainly built phishing account detection as a node classification task, which can not capture more potential global structural features for phishing accounts. Yuan et al. [22] built phishing identification problem as the graph classification task, which used line graph to enhance the Graph2Vec method and achieved good performance. However, Yuan et al. only consider the structural features obtained from line graphs, ignoring the direction information, which plays a significant role in phishing scams' identification problem. As we know, in the process of phishing fraud, the phishing funds mostly flow from multiple accounts to a specific account. From the network's perspective, the phishing nodes' local topology may be more inclined to multiple inputs and a single output. Our method takes the direction information into consideration and builds the Directed-TSGN model, revealing the topological pattern of phishing scams.

2.2 Graph Representation

Network, as a general modeling approach, are frequently used to study various real world systems, such as social networks [19], traffic networks [15], protein interaction networks [3], literature citation networks [8], etc. Due to its unique structure characteristics, Blockchain ecosystem is naturally modeled as transaction networks to carry out related research. Simultaneously, many graph representation methods are applied to capture the dependency relationships between objects in the Blockchain network structure. Alarab et al. [2] adopted Graph Convolutional Networks (GCN) intertwined with linear layers to predict illicit transactions in the Bitcoin transaction graph and this method outperforms graph convolutional methods used in the original paper of the same data. Liu et al. [12] introduced an identify inference approach based on big graph analytics and learning, aiming to infer the identity of Blockchain addresses using the graph learning technique based on Graph Convolutional Networks. Zhang et al. [23] constructed a graph to represent both syntactic and semantic structures of an Ethereum smart contract function and introduced the graph neural network for smart contract vulnerability detection. According to the above works, one can find that graph representation methods can indeed be utilized to study blockchain networks and outperform in many different applications. In this work, we introduce three categories of graph representation methods such as handcrafted features [19], embedding method Graph2Vec [14], and deep learning method Diffpool [20],

to extract the features of TNs, TSGNs, and Directed-TSGNs, preparing for the subsequent phishing account identification.

3 Methodology

In this section, we first formulate the problem description and then present the construction detail of the transaction subgraph network model.

3.1 Problem Description

Generally, given a set of addresses on Ethereum, we can construct transaction network $G = (V, E, W)$, where the node set V indicates the set of addresses, the edge set E represents the transaction from a source address to a destination address with the transaction amounts as the weight value set W.

Here, we construct a set of transaction graphs for each target address $\mathbf{G} = \{G_{add.1}, G_{add.2}, \cdots, G_{add.n}\}$, where $G_{add.i} = (V_{add.i}, E_{add.i}, W_{add.i}, D_{add.i}, y_{add.i})$ is a transaction graph of target address i, $V_{add.i}$ represents address i and it's neighbor addresses, $E_{add.i}$ is the directed transaction set between the addresses of $V_{add.i}$ with direction set $D_{add.i}$ and weight set $W_{add.i}$, and $y_{add.i} \in Y^{|\mathbf{G}|\times|\phi|}$ is the label of address i and it's corresponding transaction subgraph, where ϕ is the label set of all target addresses. In this work. our purpose is to learn a mapping function $\mathscr{F} : \mathbf{G} \rightarrow Y$ which can predict the labels of graphs in $\mathbf{G}$. The label set Y includes phishing addresses and normal addresses in the scenario of Ethereum phishing account identification.

3.2 Transaction Subgraph Networks

In this section, we introduce the detail of our transaction subgraph network model. Firstly, we give the definitions of TSGN and Directed-TSGNs as shown in the Definition 1 and Definition 2, and then we elaborate the construction methods of transaction subgraph networks (TSGNs) and directed transaction subgraph networks (Directed-TSGNs), respectively.

Definition 1 (TSGN). *Given a transaction graph $G = (V, E, W)$, the TSGN, indicated by $T = \mathscr{L}(G)$, is a mapping from G to $T = (V', E', W')$, with the node and edge sets indicated by $V' = \{t_i | i = 0, 1, 2, \cdots\}$ and $E' \subseteq (V' \times V')$. There will generate an edge between the transaction subgraphs t_a and t_b if they share the same addresses or transactions in the original transaction graph G. The W' will be calculated by a weight mapping function $W' \leftarrow f(W)$.*

Definition 2 (Directed-TSGN). *Given a directed transaction graph $G = (V, E, W, D)$, the Directed-TSGN, denoted by $T_D = \mathscr{F}(G)$, is a mapping from G to $T_D = (V^*, E^*, W^*, D^*)$, with the node and edge sets denoted by $V^* = \{d_i | i = 0, 1, 2, \cdots\}$ and $E^* \subseteq (V^* \times V^*)$. A directed edge will be built between two directed transaction subgraphs d_a and d_b when they meet the following conditions: In the original directed transaction graph G, (i) they share the common*

addresses or transactions, (ii) and form a path with the same direction. The W^* will be calculated by a weight mapping function $W^* \leftarrow f'(W)$.

According to the above definitions, we can see that TSGN is a variant of SGN model [19] on Ethereum transaction networks. Different from SGN model, TSGN adds a network weight mapping mechanism, which can retain the transaction amount information in the original transaction network for downstream network analysis tasks. Based on TSGN model, Directed-TSGN introduces the direction information into the mapping mechanism which can capture the path of transaction behavior. Next, we will focus on demonstrating the specific construction methods.

3.3 Constructing TSGN

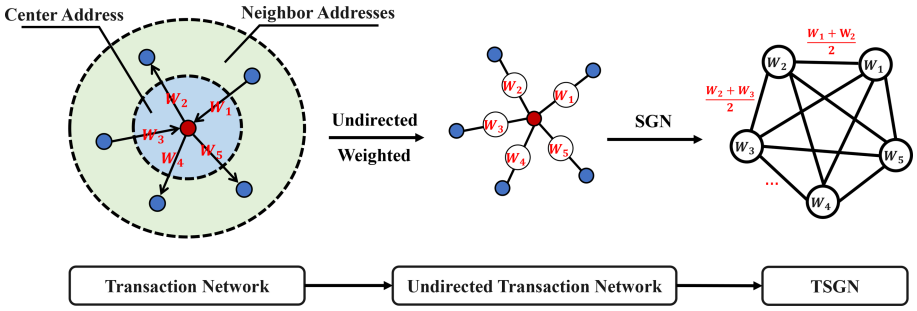

Fig. 1. A toy example of constructing TSGN.

Figure 1 shows the process of constructing TSGN. Given an original transaction network composed of a center address and it's neighbor addresses, we can firstly get a plain transaction network with weight values after undirected processing. And then, we map this network into TSGN structural space. Specifically, the edges in the undirected transaction network is mapping to the nodes W_1, W_2, W_3, W_4, W_5 of TSGN, and then new edges are built between nodes W_1, W_2, W_3, W_4, W_5 because the edges of undirected transaction network share the common (red) node. We choose the mean function $f(W_{ij}) = Mean(W_i, W_j)$ as weight mapping function in Definition 1, i.e., the weight of edge (W_1, W_2) can be calculated as $(W_1 + W_2)/2$. Of course, different weight mapping functions can be defined as required.

3.4 Constructing Directed-TSGN

According to the Sect. 3.3, we can find that the TSGN becomes more complex than the original transaction network, even a fully connected network, which may reduce graph mining algorithms' representation ability. Moreover, the mapping

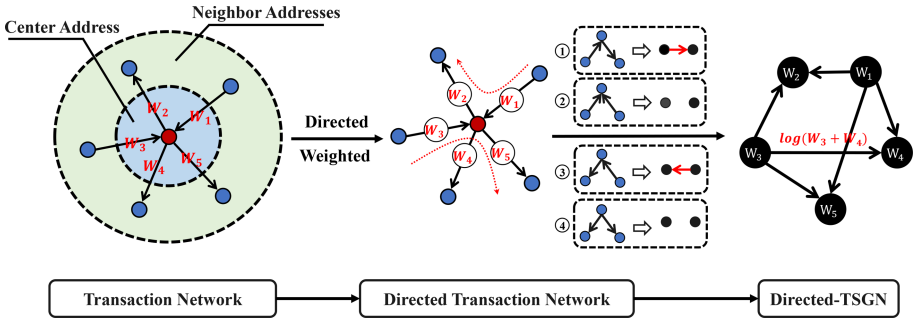

Fig. 2. A toy example of constructing Directed-TSGN.

mechanism of TSGN model can not retain the direction information, which may play an important role in the following tasks.

In response to the above problems, we propose the Directed-TSGN. As shown in Fig. 2, the directed transaction network remains the direction and weighted attributes of the original transaction network. Similarly, the edges are mapped into the black nodes W_1, W_2, W_3, W_4, W_5 of the Directed-TSGN. The two red directed dashed lines indicate that the transactions W_1 and W_2 and transactions W_3 and W_4 can be seen as two continuous transaction behaviors, respectively. In other words, the edges with weights W_1 and W_2 and the edges W_3 and W_4 can form two paths with the same direction, respectively. According to the four direction mapping strategies, we can build the new edges in the Directed-TSGN. Due to the fact that ② and ④ don't satisfy the requirements of constructing edges, the Directed-TSGN can limit the network size and get a relatively sparse transaction subgraph network. Here, $f'(W_{ij}) = log(W_i + W_j)$ is chosen as the weight mapping function in Definition 2.

4 Experimental Evaluations

4.1 Datasets

Ethereum, today's largest blockchain-based application, has fully open transaction data which can be easily accessed through the API of Etherscan(`etherscan.io`). Considering that the entire transaction network is enormous, we crawl some phishing addresses and normal addresses as the target nodes and only extract their first-order neighbor nodes from the Ethereum transaction records to construct transaction network datasets. After filtering and preprocessing the raw data, we finally got 1626 transaction networks centered on phishing nodes, and 1641 transaction networks centered on normal nodes. And then, these networks will be randomly divided so that we finally get three balanced datasets, each of which has 500 transaction networks of phishing addresses and 500 transaction networks of normal addresses. And the next experiments will be verified on these three datasets. The basic statistics of the three datasets are shown in Table 1.

Table 1. Statistics for the three datasets. N_G is the number of graphs, $\#C_{max}$ is the number of graphs belonging to the largest class, N_C is the number of classes, and #Nodes and #Edges are the average numbers of nodes and edges, respectively, of the graphs in the dataset.

Dataset	N_G	$\#C_{max}$	N_C	#Nodes	#Edges
EthereumG1	1000	500	2	26.003	25.031
EthereumG2	1000	500	2	31.650	30.673
EthereumG3	1000	500	2	26.338	25.369

4.2 Metrics

In order to accurately evaluate the quality of each classification model, in this paper, we will use *F1-Score* as a metric,

$$F_1 = \frac{2PR}{P+R}, \tag{1}$$

where P is precision and R is recall. *F1-Score* is the harmonic mean of precision and recall, so it can more comprehensively judge the pros and cons of the classification models.

4.3 Baselines and Experimental Setup

For the phishing account detection problem, we transform it into a graph classification task. In order to better verify the effect of the model proposed, we adopt three typical feature extraction methods to generate graph representation, namely handcrafted attributes, Graph2Vec, and Diffpool, which are introduced in the following.

Handcrafted Attributes. Many classic topological attributes can be utilized to analyze some important patterns in a complex network and can capture some basic information for graph classification [11,17,19], link prediction [6] and so on. In this paper, we aim to represent the networks by manually extracting the transaction network features, which are used in the downstream graph classification task. We mainly extracted 10 network features such as *the number of network nodes*, *the number of network edges* and *the average clustering coefficient*, etc. See the Appendix for details.

Graph2Vec. Graph2Vec [14] is the first embedding framework for the entire networks, which is relied on the embedding technique that has achieved impressive performances in NLP. Graph2Vec extracts some rooted subgraphs around different nodes and analogies them to Doc2Vec's context words and thus learns the graph representation [10]. Graph2Vec can learn the distributed representations of arbitrary sized graphs such that it can ignore the problem such as poor generalization.

Diffpool. This method [20] proposed a differentiable graph pool module, which can generate hierarchical representations of graphs and can be combined with various graph neural network architectures in an end-to-end manner. Diffpool learns the distinguishable soft cluster allocation of nodes on each layer of deep GNN and maps the nodes to a set of clusters, which then form the coarse input of the next GNN layer. This method mainly solves the problem that the traditional GNN methods are flat and can't learn the hierarchical representations of graphs.

Parameter Setting. The experimental part is mainly divided into two steps: the representation of the graph and the graph feature classification. In the graph feature representation part, we used the above three graph representation methods to extract features of TN, TSGN, and Directed-TSGN. For *Handcrafted Attributes*, there are no hyperparameters, just extract 10 features of each graph. For *Graph2Vec*, the parameter height of the WL kernel is set to 3. The embedding dimension is set to a commonly-used value of 1,024. For TN and Directed-TSGN, the parameters weight and direction are set to true. But for TSGN, we set the direction to false and the weight to true. We set the other parameters as: the learning rate is 0.025 and the epoch is 1000. For *Diffpool*, the parameter settings of the model are the same as in [20]. We also set corresponding initialization features for different networks. For TN, the node feature is a two-dimensional vector composed of in-degree and out-degree, while for TSGN and Directed-TSGN, the node feature is a one-dimensional vector composed of the weight of the corresponding link before graph mapping. In the graph feature classifier part, each dataset is randomly split into 9 folds for training and 1 fold for testing. To exclude the random effect of fold assignment, the experiment is repeated 500 times using the random forest classifier and then records the average F_1-*Score* and its standard deviation.

4.4 Results

According to the above setting, we conduct some experiments on the three Ethereum datasets, and the results of phishing account identification are shown in Table 2. We can find that, compared with the transaction networks (original), TSGN and Directed-TSGN models indeed has good performances in enhancing the phishing account identification. Interestingly, TSGN achieves the best classification performance 94.35% and 93.64%, in 2 of 9 cases based on the deep learning method Diffpool. Overall, Directed-TSGN increases the performance of the original classification results in 7 of 9 cases. Combined with the Handcrafted Attributes method, Directed-TSGN outperforms TN, leading to an increase of 1.11%. Directed-TSGN has an improvement over TN on all datasets, and it leads to an increase of 11.70% when considering the Graph2Vec method. We can see that TSGN and Directed-TSGN achieve the state-of-the-art results on the deep learning method Diffpool, which indicates that our TSGN model can further improve the representation capability of the deep learning method.

Table 2. The classification performance of different transaction subgraph network model.

Datasets	EthereumG1	EthereumG2	EthereumG3
Algorithm	Handcrafted		
TN(Original)	74.74±3.42	76.90±2.65	72.84±2.92
TSGN	75.25±1.63	76.94±2.29	73.04±2.43
Directed-TSGN	**75.35±3.88**	**77.50±2.25**	**73.95±2.31**
Algorithm	Graph2Vec		
TN(Original)	56.45±4.18	57.25±1.79	61.80±2.23
TSGN	56.95±2.42	57.85±2.72	62.05±3.02
Directed-TSGN	**68.15±2.26**	**68.10±1.28**	**64.15±2.48**
Algorithm	Diffpool		
TN(Original)	93.09±1.31	89.10±1.64	92.85±1.09
TSGN	**94.35±1.39**	**93.64±1.32**	93.25±1.49
Directed-TSGN	93.35±1.18	89.20±1.53	**93.90±2.43**

Furthermore, we record the computational times and compare the time consumption of constructing TSGN and Directed-TSGN on three datasets. The results are presented in Table 3, where one can see that, the computational time of Directed-TSGN is much less than that of TSGN on each dataset, decreasing from 3 hundred seconds to less than 70 s. Such results suggest that, Directed-TSGN can further enhance the performance of the algorithm for phishing account identification, while also greatly improve the efficiency of the algorithms.

Table 3. Time consumption (sec.) of constructing TSGNs and Directed-TSGNs.

Dataset	TSGN	Directed-TSGN
EthereumG1	1.355×10^2	7.3687
EthereumG2	3.650×10^2	56.9006
EthereumG3	1.264×10^2	65.3633

5 Conclusion

In this paper, we present a novel transaction subgraph network (TSGN) model for phishing account identification. By introducing different mapping mechanisms into the transaction networks, we built TSGN and Directed-TSGN models to enhance the classification algorithms. Compared with the TNs, our TSGN indeed provide more potential information to benefit the phishing account identification. Considering the direction attributes, the Directed-TSGNs can retain the transaction flow information that captures the significant topological pattern

of phishing scams. By comparing with the TSGN, Directed-TSGN is of a controllable scale and indeed have much lower time complexity, benefiting the network feature extraction methods to learn the network structure with higher efficiency. Experimental results demonstrate that, combined with network representation algorithms, the TSGN and Directed-TSGN models can capture more features to enhance the classification algorithm and improve phishing nodes identification accuracy in the Ethereum networks. In particular, when deep learning methods Diffpool is adopted to extract the features of these networks, we can achieve the state-of-the-art results on all datasets.

Acknowledgments. This work was partially supported by the National Key R&D Program of China under Grant No. 2020YFB1006104, by the National Natural Science Foundation of China under Grant No. 61973273, and by the Zhejiang Provincial Natural Science Foundation of China under Grant No. LR19F030001.

7 Appendix

- **Number of Graph Nodes** (N): The number of nodes in the graph.
- **Number of Graph Edges** (E): The number of edges in the graph.
- **Average Degree** (D_A): The mean number of edges connected to a node in the graph.
- **Percentage of leaf nodes** (P): A node is defined as a leaf node if it's degree is 1. If there are l leaf nodes in the graph, the percentage of leaf nodes can be calculated as $P = l/N$.
- **Average Clustering Coefficient** (C_{coef}): The clustering coefficient is a classic measure to quantify the edge density of the ego-network. Given a graph, there are m_i neighbors of node v_i and they are connected by e_i edges. Then, the average clustering coefficient of the graph can be defined as

$$C_{coef} = \frac{1}{N} \sum_{i=1}^{N} \frac{2e_i}{m_i(m_i - 1)} . \tag{2}$$

- **Largest Eigenvalue of the Adjacency Matrix** (λ): Given a graph G, it can be represented as an adjacency matrix $A^{N \times N}$. As the isomorphic invariant, we can adopt the largest one λ of eigenvalues of A as the graph feature.
- **Network Density** (D_N): Given a network, the numbers of nodes and edges are N and E, then the network density can be defined as $D = 2E/N(N-1)$
- **Average Betweenness Centrality** (C_{betw}): For each pair of nodes in a connected network, there exists at least one shortest path between the nodes such that the number of edges that construct this path is minimized. The betweenness centrality of a node is a measure of centrality based on the shortest paths. So, the betweenness centrality of a node is defined as

$$C_{betw}(i) = \sum_{m \neq i \neq n} \frac{e_{mn}(i)}{e_{mn}} . \tag{3}$$

$C_{betw}(i)$ can reflect the importance of node i as a bridge node. Where e_{mn} is the number of shortest paths between v_m and v_n, and $e_{mn}(i)$ is the number of shortest paths between v_m and v_n that pass through v_i.
Then, the average betweenness centrality of the network can be calculated as

$$C_{betw} = \frac{1}{N} \sum_{i=1}^{N} C_{betw}(i) \,. \tag{4}$$

– **Average Closeness Centrality (C_{close})**: The closeness centrality is also a measure of centrality based on the shortest paths, which requires taking into account the shortest paths from each node to the other nodes. Given a connected network, the closeness centrality of a node is represented as the reciprocal of the sum of shortest path length between this node and the others. The average closeness centrality of the network is defined as

$$C_{close} = \frac{1}{N} \sum_{i=1}^{N} \frac{k-1}{\sum_{j=1}^{k} e_{ij}} \,, \tag{5}$$

where e_{ij} is the shortest path length between nodes v_i and v_j.
– **Average Neighbor Degree ($D_{neighbor}$)**: The neighbor degree of a node is the average degree of all the neighbors of this node, which can capture the 2-hop information. We can calculate the neighbor degree of the node v_i as

$$D_{neighbor}(i) = \frac{1}{k_i} \sum_{v_j \in \mathcal{N}_i} k_j \,, \tag{6}$$

where $\mathcal{N}_i$ is the neighbor set of node v_i, and k_i and k_j are the degrees of node v_i and $v_j \in \Omega_i$. For a network, we can get the average neighbor degree

$$D_{neighbor} = \frac{1}{N} \sum_{i=1}^{N} D_{neighbor}(i) \tag{7}$$

References

1. Adebowale, M.A., Lwin, K.T., Sanchez, E., Hossain, M.A.: Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text. Expert Syst. Appl. **115**, 300–313 (2019)
2. Alarab, I., Prakoonwit, S., Nacer, M.I.: Competence of graph convolutional networks for anti-money laundering in bitcoin blockchain. In: Proceedings of the 2020 5th International Conference on Machine Learning Technologies, pp. 23–27 (2020)
3. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**, i47–i56 (2005)

4. Chen, L., Peng, J., Liu, Y., Li, J., Xie, F., Zheng, Z.: Phishing scams detection in Ethereum transaction network. ACM Trans. Internet Technol. (TOIT) **21**(1), 1–16 (2020)
5. Feng, F., Zhou, Q., Shen, Z., Yang, X., Han, L., Wang, J.Q.: The application of a novel neural network in the detection of phishing websites. J. Ambient Intell. Humanized Comput. 1–15 (2018). https://doi.org/10.1007/s12652-018-0786-3
6. Fu, C., et al.: Link weight prediction using supervised learning methods and its application to yelp layered network. IEEE Trans. Knowl. Data Eng. **30**(8), 1507–1518 (2018)
7. Gualberto, E.S., De Sousa, R.T., Vieira, T.P.D.B., Da Costa, J.P.C.L., Duque, C.G.: The answer is in the text: multi-stage methods for phishing detection based on feature engineering. IEEE Access **8**, 223529–223547 (2020)
8. Hosseini, M.R., Maghrebi, M., Akbarnezhad, A., Martek, I., Arashpour, M.: Analysis of citation networks in building information modeling research. J. Constr. Eng. Manage. **144**(8), 04018064 (2018)
9. Khonji, M., Iraqi, Y., Jones, A.: Phishing detection: a literature survey. IEEE Commun. Surv. Tutorials **15**(4), 2091–2121 (2013)
10. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
11. Li, G., Semerci, M., Yener, B., Zaki, M.J.: Graph classification via topological and label attributes. In: Proceedings of the 9th International Workshop on Mining and Learning with Graphs (MLG), vol. 2, San Diego, USA (2011)
12. Liu, X., Tang, Z., Li, P., Guo, S., Fan, X., Zhang, J.: A graph learning based approach for identity inference in dapp platform blockchain. IEEE Trans. Emerg. Top. Comput. (2020)
13. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Technical Report, Manubot (2019)
14. Narayanan, A., Chandramohan, M., Chen, L., Liu, Y., Saminathan, S.: subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs. In: International Workshop on Mining and Learning with Graphs (2016)
15. Ruan, Z., Song, C., Yang, X.H., Shen, G., Liu, Z.: Empirical analysis of urban road traffic network: a case study in Hangzhou city, china. Phys. Stat. Mech. Appl. **527**, 121287 (2019)
16. Sahingoz, O.K., Buber, E., Demir, O., Diri, B.: Machine learning based phishing detection from URLs. Expert Syst. Appl. **117**, 345–357 (2019)
17. Wang, J., et al.: Sampling subgraph network with application to graph classification. arXiv preprint arXiv:2102.05272 (2021)
18. Wu, J., et al.: Who are the phishers? phishing scam detection on Ethereum via network embedding. IEEE Trans. Syst. Man Cybern. Syst. (2020)
19. Xuan, Q., et al.: Subgraph networks with application to structural feature space expansion. IEEE Trans. Knowl. Data Eng. (2019). https://doi.org/10.1109/TKDE.2019.2957755
20. Ying, R., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 4805–4815 (2018)

21. Yuan, Y., Wang, F.Y.: Blockchain and cryptocurrencies: model, techniques, and applications. IEEE Trans. Syst. Man Cybern. Syst. **48**(9), 1421–1428 (2018)
22. Yuan, Z., Yuan, Q., Wu, J.: Phishing detection on Ethereum via learning representation of transaction subgraphs. In: Zheng, Z., Dai, H.-N., Fu, X., Chen, B. (eds.) BlockSys 2020. CCIS, vol. 1267, pp. 178–191. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-9213-3_14
23. Zhuang, Y., Liu, Z., Qian, P., Liu, Q., Wang, X., He, Q.: Smart contract vulnerability detection using graph neural networks. In: Proceedings of the 2020 29th International Joint Conference on Artificial Intelligence, pp. 3283–3290 (2020)