TSGN: Transaction Subgraph Networks Assisting Phishing Detection in Ethereum

Jinhuan Wang, Pengtao Chen, Xinyao Xu, Jiajing Wu, Senior Member, IEEE, Meng Shen, Member, IEEE, Qi Xuan, Senior Member, IEEE, Xiaoniu Yang

Abstract—Due to the decentralized and public nature of the Blockchain ecosystem, the malicious activities on the Ethereum platform impose immeasurable losses for the users. Existing phishing scam detection methods mostly rely only on the analysis of original transaction networks, which is difficult to dig deeply into the transaction patterns hidden in the network structure of transaction interaction. In this paper, we propose a Transaction SubGraph Network (TSGN) based phishing accounts identification framework for Ethereum. We first extract transaction subgraphs for target accounts and then expand these subgraphs into corresponding TSGNs based on the different mapping mechanisms. In order to make our model incorporate more important information about real transactions, we encode the transaction attributes into the modeling process of TSGNs, yielding two variants of TSGN, i.e., Directed-TSGN and Temporal-TSGN, which can be applied to the different attributed networks. Especially, by introducing TSGN into multi-edge transaction networks, the Multiple-TSGN model proposed is able to preserve the temporal transaction flow information and capture the significant topological pattern of phishing scams, while reducing the time complexity of modeling large-scale networks. Extensive experimental results show that TSGN models can provide more potential information to improve the performance of phishing detection by incorporating graph representation learning.

Index Terms—Ethereum · Phishing identification · Subgraph network · Network representation · Graph classification.

I. INTRODUCTION

TROUGH consensus mechanisms and voluntary respect for the social contract, the internet can be utilized to make a decentralised value-transfer system that can be shared around the world and virtually free to use [1], Ethereum [2], a very professional system of a cryptographically secure, offers us information that has never been seen before in the financial world. In contrast to fiat currencies of financial world, the transactions through virtual currencies like Ether are completely transparent. Accounts can freely and conveniently conduct transactions with currency and information and do not have to rely on traditional third parties, and these transactions of cryptocurrencies are permanently recorded on Blockchain and are available at any time.

There is no doubt that Ethereum platform facilitates transactions between consenting individuals who are trust with each other. But, the cryptocurrency market provided by Ethereum is inevitably flooded with a variety of cybercrimes due to anonymity and unsupervised organization, including smart Ponzi schemes, phishing, money laundering, fraud, and criminal-related activities. The overall losses caused by DeFi exploits on Ethereum, the backbone of many DeFi applications, have totaled $12 billion so far in 2021 according to Elliptic [1] a firm which tracks movements of funds on the digital ledgers that underpin cryptocurrencies. Moreover, it is reported that phishing scams can break out periodically and are the most deceptive form of fraud [3]. In general, phishing is a kind of cybercrime aiming to exploit the weaknesses of users and obtain personal and confidential information [4]. Some scams lure investors to visit/click the fake projects/websites by offering extra Ether coins as the enticement, and to buy digital assets from scammers for the purpose of phishing scams. 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 maintain a more secure blockchain ecosystem.

Existing methods of phishing detection on Ethereum mainly fall into two categories, propose elaborate handcrafted features [5]–[7] and design automatic feature learning models [8]–[10]. These studies attempt to learn more useful information from the raw transaction data to reveal the transaction patterns, which are effective for phishing detection but have several important omissions. First, the transaction networks constructed by sampling the original Ethereum transaction data may result in missing structural information which can affect the performance of learning models. This forces researchers to design more precise and data-targeted representation methods to improve the effect, but at the expense of the generalization performance of models on datasets of different attributes and

[https://blockchaingroup.io/](https://blockchaingroup.io/)
scales. Second, the above models are well-designed only for learning the original networks, ignoring the potential transaction flow patterns provided by the interaction of transactions.

In this paper, inspired by SGN model \cite{11}, we develop a network model called TSGN to complement the features of original networks and further assist in identifying phishing accounts. According to the essential attributes contained in the transactions, we design the specific structural mapping strategies in order to encode the weight, direction and timestamp information into transaction subgraph networks. And then, the transaction network is mapped into the transaction subgraph structure space according to the different strategies to obtain the corresponding TSGN and its variants for the subsequent feature extraction and detection task. The algorithm framework is shown in Fig. 1. Specifically, our contributions can be concluded as follows:

- We propose a novel transaction network model, transaction subgraph networks (TSGNs). According to original transaction networks (TN) of various attributes and scales, our TSGNs can introduce different network mapping strategies to fully capture the potential structural topological information which can not be obtained easily from raw transaction networks, benefiting the subsequent phishing detection algorithms.
- We built the first variant of TSGN model, i.e., Directed-TSGN, to integrate direction attribute for effectively characterize transaction flow information from the original transaction data. And, we further introduce timestamp information into Directed-TSGN, producing the Temporal-TSGN which can exactly learn the time-aware transaction behaviors and is applied on multi-edge transaction networks.
- Experimental results demonstrate the effectiveness of the TSGN models by integrating various phishing detection models such as Handcrafted features, Graph2Vec, Deep-Kernel, Diffpool, and U2GNN. Especially, the classification result of Temporal-TSGN increases to 98.43% (97.00% for Directed-TSGN) when only Diffpool is considered, greatly improving the phishing detection performance. More remarkably, compared with TSGN, the variants of TSGN are of the controllable scale and have lower time cost.

The rest of the paper is structured as follows. In section II we make a brief description of the phishing identification and graph representation methods. In Section III we mainly introduce the definitions and construction methods of transaction subgraph networks. In Section IV we give several feature extraction methods, which together with TN, TSGN, Directed-TSGN, and Temporal-TSGN are applied to six Ethereum transaction network datasets. Finally, we conclude our paper in Section V. For further study, our source codes are available online.

II. BACKGROUND AND RELATED WORK

In this section, we provide some necessary background information on phishing detection methods and graph representation algorithms for blockchain data mining.

A. Phishing Identification

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. Wu et al. \cite{9} proposed an approach to detect phishing scams on Ethereum by mining its transaction records. By considering the transaction amount and timestamp, it introduced a novel network embedding algorithm called trans2vec to extract the features of the addresses for subsequent phishing identification. Chen et al. \cite{6} proposed a detecting method based on Graph Convolutional Network (GCN) and autoencoder to precisely distinguish phishing accounts. Li et al. \cite{12} introduced a self-supervised incremental deep graph learning model based on Graph Neural Network (GNN), for phishing scam detection problem on Ethereum. One can see that these methods mentioned above mainly built phishing account detection as a node classification task, which cannot capture more potential global structural features for phishing accounts. Yuan et al. \cite{13} 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 transaction temporal information which plays a significant role in phishing scam detection. As we know, the phishing funds mostly flow from multiple accounts to a specific account. Our method takes the direction and temporal information into consideration and builds the Directed-TSGN and Temporal-TSGN models, revealing the topological pattern of phishing scams.

B. Graph Representation on Blockchain

Graphs are a general language for describing and analyzing entities with relations or interactions. Due to its unique structure characteristics, the 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 structures. There are two main categories of existing representation methods, the former mostly employs machine learning models with dedicated feature engineering \cite{5, 7}, and the latter adopts some embedding techniques and deep learning models such as walk-based \cite{9} and the GNN methods. For example, Hu et al. \cite{8} presented

https://github.com/GalateaWang/TSGN-master
a novel deep learning scam detection framework assembling with the Gate Recurrent Unit (GRU) network and attention mechanism, which produced good results on both Ponzi and Honeypot scams. Alarab et al. [10] adopted GCNs 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. [14] introduced an identity inference approach based on big graph analytics and learning, aiming to infer the identity of Blockchain addresses using the graph learning technique based on GCNs. Zhang et al. [15] constructed a graph to represent both syntactic and semantic structures of an Ethereum smart contract function and introduced the GNN for smart contract vulnerability detection.

According to the above works, one can find that the graph representation methods, especially GNNs, can indeed be utilized to study the Blockchain data and outperform in many different applications. However, the existing studies have two potential problems. Firstly, these researches are mostly based on a large-scale transaction network where the scarcity of labels and the huge volume of transactions make it difficult and intricate to take advantage of GNN methods. Secondly, such methods largely rely on the original transaction network and thus may overlook the powerful potential of transaction interactions which contain important hidden transaction patterns. In this work, we collect the 1-hop neighborhood information to construct the subgraph for each target address and use the features of the entire subgraph as the representation of the target node for avoiding inadequate graph feature learning. We further obtain the TSGNs and their three variants by structural space mapping, capturing potential transaction patterns from various aspects.

III. METHODOLOGY

We first give the description of problem formulation of phishing identification and then illustrate the construction details of the transaction subgraph network models by introducing several definitions.

A. Problem Formulation

Generally, a cryptocurrency transaction network is modeled as an address interaction graph, illustrating the transferring amount and the flow direction of fund between different accounts in a certain time bucket. Given a set of addresses on Ethereum, we can construct transaction network as a directed graph $G = (V, E)$, where each node $v \in V$ indicates an address that can be an Externally Owned Account (EOA) or Contract Account (CA), the edge set is represented as a quad, $E = \{(v_{src}, v_{dst}, w, \ell)|v_{src}, v_{dst} \in V, w \in \mathbb{R}^+ \cup \{0\}, \ell \in L\}$, where $(v_{src}, v_{dst})$ indicates the direction of the transaction (e.g., assets transfer or smart contract invocation) from $v_{src}$ to $v_{dst}$, weight $w$ represents the amount of transferred cryptocurrencies or is 0 if the kind of transaction is smart contract invocation, and $l$ is the timestamp of the transaction.

Here, we construct a set of transaction graphs for the center addresses $G = \{(G_1, c_1), (G_2, c_2), \ldots, (G_n, c_n)\}$, where $c_i \in C^{|G| \times |c|}$ is the label of address $i$ and its corresponding transaction subgraph $G_i$, where $\phi$ is the label set of all target addresses. In this work, our purpose is to learn a mapping function $\mathcal{G} : G \rightarrow C$ which can predict the labels of graphs in $G$. The label set $C$ includes phishing labels and non-phishing labels in the scenario of Ethereum phishing account identification.

To keep mathematical formulas unified in a stricter and tidier manner, TABLE I shows the notations used in the progress of TSGNs modeling.

| Notation | Description |
|----------|-------------|
| $G$      | The original Ethereum transaction graph |
| $T$      | The TSGN graph |
| $T_D$    | The Directed-TSGN graph |
| $T_T$    | The Temporal-TSGN graph |
| $T_A$    | The generic notation of TSGN graph with an attribute |
| $v, u$   | The account nodes in graph $G$ |
| $e$      | A transaction edge in graph $G$ |
| $d$      | The directed transaction edge in graph $G$ |
| $t$      | The temporal directed transaction edge in graph $G$ |
| $\ell$   | The timestamp attached on the edge $t$ |
| $w, w', w^*, w_\infty$ | The edge weights set by the transaction amount |
| $V, V', V^*, V_\infty$ | The node sets |
| $E, E', E^*, E_\infty$ | The edge sets |
| $W, W'$ | The edge weight sets |
| $f(\cdot)$ | The weight mapping function |
| $\mathcal{L}(\cdot), \mathcal{X}(\cdot), \mathcal{H}(\cdot)$ | The structural mapping functions |
| $A(\cdot)$ | The function for getting the attributed object |
| $G$ | The set of graphs |
| $c, C$ | The label of the graph $G$ and the label set |

B. Transaction Subgraph Networks

In this section, we introduce the detail of our transaction subgraph network model. Firstly, we give the definitions of transaction subgraph network (TSGN), directed transaction subgraph network (Directed-TSGN), and temporal transaction network (Temporal-TSGN). And then, we elaborate the construction methods of the TSGN, Directed-TSGN, and Temporal-TSGN, respectively.

Definition 1 (TSGN). Given a transaction graph $G = (V, E)$ where $e = (v_{ik}, v_{kj}, w) \in E$ indicates a transaction edge with the weight $w$, the TSGN, denoted by $T = \mathcal{L}(G)$, is a mapping from $G$ to $T = (V', E')$, with the node and edge sets denoted by $V' = \{e_i \in E\}$ and $E' = \{(e_i, e_j, w')\}$, $i, j = 0, 1, 2, \ldots$. The transaction subgraphs $e_{ai}$ and $e_{bj}$ will be connected if they share the common addresses in the transaction graph $G$.

According to the Definition I, we can find that TSGN is the variant of SGN model for the background of Ethereum transaction ecosystem. Different from SGN model, TSGN adds a network weight mapping mechanism $W' \leftarrow f(W)$, which can retain the transaction amount information in the original transaction network for the downstream behavior analysis task. The edge weights of TSGN can be calculated specifically...
Fig. 2. The process of building TSGN from a given transaction network.

\[ w' = \begin{cases} \log\left(\frac{w_a + w_b}{2}\right), & \text{if } w_a \neq 0 \text{ or } w_b \neq 0 \\ 0, & \text{if } w_a = 0 \text{ and } w_b = 0 \end{cases} \quad (1) \]

where \( w' \in W' \) is the mapped weights, and \( w_a, w_b \in W \) are the original weights. Note that transaction amount is set to 0 when the account interaction relationship is the invocation of contract accounts. Here, TSGN is a classical model which can be applied for any simple Ethereum account interaction network. Considering the complex transaction direction information, we propose the variant named Directed-TSGN.

**Definition 2 (Directed-TSGN).** Given a transaction digraph \( G = (V, E) \), where \( d = (v_{\text{src}}, v_{\text{dst}}, w) \in E \) indicates a directed transaction edge, the Directed-TSGN, denoted by \( T_D = \mathcal{X}(G) \), is a mapping from \( G \) to \( T_D = (V^*, E^*) \), with the node and edge sets denoted by \( V^* = \{d_i \in E | i = 0, 1, 2, \cdots\} \). There will be a directed interaction edge \((d_a, d_b, w^*) \in E^* \) between transaction subgraphs \( d_a \) and \( d_b \) when they meet the following conditions:

- transaction subgraphs \( d_a \) and \( d_b \) share the common addresses in the weighted directed transaction graph \( G \);
- the source of \( d_a \) (or \( d_b \)) is the destination of \( d_b \) (or \( d_a \)), which can form a 2-top path with the same direction.

As shown in the Definition 2, Directed-TSGN based on TSGN model introduces the direction information into the mapping mechanism which can capture the path of transaction behavior. The weight \( w^* \) can be obtained by Eq. (1). Also, Ethereum data is dynamic and gets larger and more complicated over time, which brings the challenge to analyze transaction behaviors by adopting some graph mining algorithms. In addition, phishing behaviors generally have the strong time-aware property. These challenges require the development of a novel graph model mapping approach that is suitable for the analysis of time-varying and dense networks. Hence, we further propose Temporal-TSGN.

**Definition 3 (Temporal-TSGN).** Given a temporal transaction digraph \( G = (V, E) \), where \( t = (v_{\text{src}}, v_{\text{dst}}, w, \ell) \in E \) indicates a directed transaction edge with the timestamp \( \ell \), the Temporal-TSGN, denoted by \( T_T = \mathcal{X}(G) \), is a mapping from \( G \) to \( T_T = (V_x, E_x) \), with the node and edge sets denoted by \( V_x = \{t_i \in E | i = 0, 1, 2, \cdots\} \) and \( E_x \subseteq (V_x \times V_x) \). A directed edge \((t_a, t_b, w_x) \in E_x \) will be built between two temporal directed transaction subgraphs \( t_a \) and \( t_b \) when they meet the stricter conditions: In the original temporal transaction graph \( G \),

- \( t_a \) and \( t_b \) share the common addresses in the original directed transaction graph \( G \);
- the source of \( t_a \) (or \( t_b \)) is the destination of \( t_b \) (or \( t_a \)), which can form a 2-top path with the same direction.
- the timestamp \( \ell_a < \ell_b \) (or \( \ell_b < \ell_a \)) makes \((t_a, t_b) \) (or \((t_a, t_b)) \) present a sequential transaction flow.

Clearly, the Temporal-TSGN defined in the Definition 3 is the substructure which is obtained by filtering the nodes and edges of of Directed-TSGN according to the time information, i.e., \( T_T \subseteq T_D \). By attaching the time attribute on the directed transaction subgraphs (edges), Temporal-TSGN can capture neighbor transaction flow information for each target account address while reducing the scale of the network. The weight value \( w_x \) can also be calculated by Eq. (1). Next, we would focus on the specific construction methods for the TSGN, Directed-TSGN, and Temporal-TSGN.

**C. Construction of TSGN**

Fig. 2 shows the process of constructing TSGN. Given an original transaction network composed of a center address and its neighbor addresses, we can firstly get a plain transaction network with weight values by discarding the direction attribute and retaining the weight attribute. And then, this network is mapped into TSGN structural space according to the rule indicated in the Definition 1.

Specifically, the edges in the undirected transaction network is mapping to the nodes of TSGN, and then new edges are built between nodes \( e_1, e_2, e_3, e_4, e_5 \) because the edges of undirected transaction network share the common center node. There is no doubt that transaction subgraph mapping strategy of TSGN also works for the case that there are connections between neighbor addresses. The transactions \( e_6 \) and \( e_7 \) are mapped into the TSGN to acquire the neighbor trading pattern information for the center address. A pseudocode of constructing TSGN is given in Algorithm 1. The input of this algorithm is the original transaction network \( G(V, E) \) and the output is the constructed TSGN, denoted by \( T(V', E') \), where \( V' \) and \( E' \) represent the sets of nodes and edges in the TSGN, respectively. Here, we choose the weight mapping function introduced in Section III-B. Of course, different weight mapping functions can be defined as required.

**D. Construction of Directed-TSGN**

Transaction direction is one of the most noteworthy characteristics in the field of financial research. Most of studies model the transaction pattern of an account by Indegree and Outdegree in a transaction graph. In Ethereum or Bitcion platform, transaction direction combined with transaction amount information can be adopted to study the peculiar unusual patterns [15], detect abnormal economic interactions [3], predict virtual currency prices [17] and so on. According to Section III-C, we can find that the TSGN model can not retain the direction information which may play an important role in the following phishing identification tasks. Moreover, the TSGN is denser than the original transaction network and even becomes a fully connected network, which may reduce the
Temporal-TSGN will be executed when Algorithm 2 aiming to illustrate the Directed-TSGN mapping (Algorithm 1 and 2). The input of Algorithm 2 is two transaction constructing edges, the Directed-TSGN can limit the network build the new edges in the Directed-TSGN. Due to the fact according to the four direction mapping strategies, we can form two 2-hop paths with the same direction, respectively. In other words, the edges $d_1$ and $d_2$ and the edges $d_3$ and $d_5$ can form two 2-hop 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 patterns of (b) and (d) don’t satisfy the requirements of constructing edges, the Directed-TSGN can limit the network size and get a relatively sparse transaction subgraph network. The pseudocodes of constructing Directed-TSGN consist of Algorithm 1 and 2. The input of Algorithm 2 is two transaction subgraphs $d_a, d_b$ in the attribute attached graph $G$, and the output is the edge $(d_a, d_b)$ matching the mapping rules. Algorithm 2 aiming to illustrate the Directed-TSGN mapping will be executed when $G \leftarrow A_{directed}(G)$ is attached the transaction direction attribute.

### E. Construction of Temporal-TSGN

Temporal information, i.e., transaction timestamps, can clearly indicate when the transaction occurred. Hence, the transaction time in alliance with trading flow direction offers the possibility for the traceability of transferring funds on Ethereum [18]. In this section, we further introduce the temporal attribute of transactions into the TSGN mapping mechanism. On the basis of Directed-TSGN, Temporal-TSGN is proposed to fuse crucial temporal and direction attributes related to phishing detection for enhancing the ability to identify phishing scams. Temporal-TSGN improves the mapping strategies compared with Directed-TSGN, which can finely captures the temporal transaction flow structural information in Ethereum, further limiting the network scale.

Fig. 4 shows the progress of constructing Temporal-TSGN. Given an original transaction network, Directed-TSGN can be firstly obtained according to the strategy mentioned in Section III-D and then we can get the Temporal-TSGN from Directed-TSGN by filtering the edges that are not in chronological order. The timestamps on the edges determine the order in which transaction are generated, i.e., $(t_1, t_2, t_3, t_4, t_5, t_6, t_7)$. We define that, in the Temporal-TSGN, the transaction $t_i$ must be linked to transaction $t_j$ where $i < j$, which can form a 2-hop sequential transaction flow. Here, the edges $(t_3, t_2), (t_7, t_3), (t_6, t_1)$ are unsatisfied with the formation condition and are therefore ignored.

At this point, the Temporal-TSGN mapping strategy can be summarized as the single edges part of Fig. 5. We attach the timestamps onto the edges of different directions, getting the fund flow of the transactions within a certain time frame. It is important to note that, a 2-hop transaction chain is available for the construction of Temporal-TSGN only under the three conditions described in the Definition 3. Specifically, the transaction chain $(v_1 \xrightarrow{4s} v_2 \xrightarrow{7s} v_3)$ shown in Fig. 5 (a) can be successfully mapped into the TSGN structural space, becoming a 1-hop directed node pair, while the other three transaction chains $(v_1 \xrightarrow{4s} v_2 \xrightarrow{7s} v_3), (v_1 \xrightarrow{4s} v_2 \xrightarrow{7s} v_3)$ and $(v_1 \xrightarrow{4s} v_2 \xrightarrow{7s} v_3)$ shall be excluded since they do not satisfy the second condition or the third condition or both in the Definition 3 respectively.

### F. TSGN for Multi-edge Case

Ethereum transaction data poses following several challenges. First, the transaction network is generally sparse and dynamic, but it could be very dense when viewed over a long period of time. Secondly, the nodes in the transaction graph appear (i.e., new accounts are created) and disappear (i.e., no future transactions occur) daily. Thirdly, transaction throughput on a daily basis fluctuates greatly. So, the time attribute is a vitally important factor when constructing transaction networks, which can exhibit the real complexity of the Ethereum transaction network from the time dimension. Li [19] modeled...
the temporal relationship of historical transaction records between nodes to construct the edge representation of the Ethereum transaction network. In this section, how to deal with the mapping rules of multi-edge temporal networks is an urgent problem to be solved for the subsequent phishing behavior analysis.

It is worth to note that the temporal mapping mechanism can be extended to the case of multiple edges. As shown in Fig. 5(e)-(l), the case of multi-edge mapping does not alter the loop of a nodes pair (Fig. 5(e) and (f)), while the case of temporal edges can tackle it (Fig. 5(i) and (j)). The timestamps can break the cycle in the network structure and orient the information flow in the TSGN. Intuitively, TSGN under multi-edge case can yield to a clear path of funding transferring and capture the fund flow pattern in the complex temporal transaction network, which is conducive to the detection of illegal trading activities.

In particular, the construction progress of TSGN for multi-edge transaction network, denoted as Multi-TSGN, is shown in Fig. 6. Firstly, given a weighted directed multi-edge graph, it will turn to a MultiDiGraph where each transaction has its own fixed order of occurrence after being attached with the timestamp attribute. And then, according to the Algorithm 3 we can get the variant of Temporal-TSGN for multi-edge transaction network, i.e., Multi-TSGN.

To date, it is still the research focus to deeply understand and analyze the knowledge of multi-edge temporal transaction networks in Ethereum. In this section, Multiple-TSGN proposed provides a novel perspective to excavate the potential transaction patterns in Ethereum.
TABLE II

| Dataset   | Type    | $N_G$ | $C_{max}$ | $N_C$ | $N$ | max$N$ | Plain | Directed | Multiedge |
|-----------|---------|-------|-----------|-------|-----|--------|-------|----------|-----------|
| EtherG1   | Star    | 700   | 350       | 2     | 7   | 13     | 6     | 12       | 8         |
| EtherG2   | Star    | 700   | 350       | 2     | 14  | 33     | 13    | 32       | 20        |
| EtherG3   | Star    | 700   | 350       | 2     | 96  | 4972   | 95    | 4972     | 100       |
| EtherG4   | Net     | 700   | 350       | 2     | 7   | 13     | 7     | 20       | 7         |
| EtherG5   | Net     | 700   | 350       | 2     | 14  | 33     | 17    | 58       | 18        |
| EtherG6   | Net     | 700   | 350       | 2     | 96  | 4972   | 161   | 7534     | 174       |

Fig. 6. The progress of building TSGN for the multi-edge case and obtain Multiple-TSGN.

IV. EXPERIMENTS

In this section, we describe the experiments conducted to evaluate the effectiveness of our TSGN models on the task of phishing detection. We first construct the network datasets by exploiting transactions records from Ethereum, then outline several baselines followed by the corresponding experimental setup. Finally, we show the experimental results with discussion.

A. Data Collection and Preparation

As today’s largest blockchain-based application, Ethereum has fully open transaction data which are permanently recorded and easily accessed through the API of Etherscan or the academic blockchain data platform XBlock.

Ethereum has gained massive momentum in recent years, mainly due to the success of cryptocurrencies. Considering that the entire cryptocurrency trading network is enormous, we crawl some phishing addresses and non-phishing addresses as the target nodes and extract their first-order neighbor nodes from the first seven million blocks of Ethereum to construct transaction network datasets (all transactions occurred between July 30, 2015 and January 2, 2019). After filtering and preprocessing the raw data, we randomly select 1050 phishing addresses and 1050 unlabeled addresses according based on the account label information on the XBlock platform, and crawl their neighbor addresses and transaction records to construct transaction networks. And then, these networks will be randomly divided so that we finally get three balanced star-form network datasets EtherG1, EtherG2, and EtherG3, each of which has 350 transaction networks of phishing addresses and 350 transaction networks of non-phishing addresses. Besides, in order to increase the structural complexity of the transaction networks, we expand the scale of the networks by introducing the interactive edges between 1-hop neighbors, yielding to three net-form transaction datasets EtherG4, EtherG5, and EtherG6. The basic statistics of these datasets are presented in TABLE III.

As shown in TABLE III different variants of TSGN model have their own unique attribute dependencies. Hence, it is necessary to sketch the transaction network structure under different attributes. Based on Fig. 7 we further illustrate the details of edge statistics of TABLE II. Plain indicates no directed and temporal attributes are included in the network structures (Fig. 7 (a) and (d)), Directed means the edges of networks are directed (Fig. 7 (b) and (e)), and Multiedge signifies the edges are attached directed and temporal attributes simultaneously to distinguish the multiple same-directed edges between a node pair (Fig. 7(c) and (f)). Note that, the weight is set as the inherent attribute. Apparently, the richer the edge attributes, the more complex the network structures. And the next experiments will be conducted on these six datasets to verify the performance of the four TSGN models.

B. Baseline Methods

In order to better verify the effect of the proposed TSGN models, we adopt six Ethereum phishing scams detection models, namely handcrafted features, Graph2Vec, DeepKernel,
Graph2Vec which are introduced

Graph2Vec

Graph2Vec

Graph2Vec

TABLE III
Network attribute statistics required by different TSGN models.

| Model          | Weighted | Directed | Temporal | Multiedge |
|----------------|----------|----------|----------|-----------|
| TSGN           | ✓        |          |          |           |
| Directed-TSGN  | ✓        | ✓        |          |           |
| Temporal-TSGN  | ✓        | ✓        | ✓        |           |
| Multiple-TSGN  | ✓        | ✓        | ✓        | ✓         |

DeepKernel [24]: This method provides a unified framework, named DGK, that defines the similarity between graphs via leveraging the dependency information of sub-structures, while learning latent representations.

Diffpool [25]: This method proposed a differentiable graph pooling module, which can generate hierarchical representations of graphs and can be combined with various GNN architectures in an end-to-end manner. This method mainly solves the problem that the traditional GNN methods are flat and can’t learn the hierarchical representations of graphs.

U2GNN [26]: It presented a GNN model leveraging a transformer self-attention mechanism followed by a recurrent transition to induce a powerful aggregation function to learn graph representations, which achieved state-of-the-art performance on many classical benchmarks for graph classification.

Line_Graph2Vec [13]: This method collected a set of subgraphs, each of which contains a target address and its surrounding transaction network, and then converted them into the corresponding line graphs which can be input into the Graph2Vec model for acquiring the representation vectors for all the graphs.

C. Experimental Setup

To investigate the performance of our TSGN models, above mentioned representation methods are adopted to extract the feature vectors for TN, TSGN, Directed-TSGN, and Temporal-TSGN, preparing for training the classification model. Here, the parameters setting of representation models under different TSGNs are given as follows.

For Handcrafted Features, there are no hyperparameters, just extract 10 features manually for each graph as the feature vectors to train the phishing detection model.

For Graph2Vec and Line_Graph2Vec, the parameter height of the WL kernel is set to 3. The commonly-used value of 1,024 is adopted as the embedding dimension. The other parameters are set to defaults: the learning rate is set to 0.025 and the epoch is set to 1000.

For DeepKernel, the Weisfeiler-Lehman subtree kernel is used to build the corpus and its height is set to 2, the embedding dimension is set to 10, the window size is set to 5 and skip-gram is used for the word2vec model. Furthermore, the node degrees are set as the node labels for TN and all TSGNs. As the key point of this method, the sub-structure similarity matrix $\mathcal{M}$, is calculated by the matrix $\mathcal{V}$ with each column representing a sub-structure vector, $\mathcal{M} = \mathcal{V}^T$. Let $\mathcal{P}$ denote the matrix with each column representing a sub-structure frequency vector. According to the definition of kernel: $K = \mathcal{P} \mathcal{M} \mathcal{P}^T = \mathcal{P} \mathcal{V} \mathcal{V}^T \mathcal{P}^T = \mathcal{H} \mathcal{H}^T$, one can use the columns in the matrix $\mathcal{H} = \mathcal{P} \mathcal{V}$ as the inputs to the classifier.

For Diffpool, we use the in-degree and out-degree as the node features for the TN, and initialize the feature matrix by the edge weights of TN for different TSGNs. The other parameters of the model are set to defaults.

For U2GNN, we use the in-degree and out-degree as the node features for the TN, and initialize the feature matrix by the edge weights of TN for different TSGNs. The other parameters of the model are set to defaults.
TABLE IV
CLASSIFICATION PERFORMANCE OF OUR TSGN MODELS ON SIX ETHERUM DATASETS, COMBINING DIFFERENT GRAPH CLASSIFICATION METHODS.

| Method      | Dataset                  | Classification results (F1-Score, %) | Star | Net | Avg. |
|-------------|--------------------------|-------------------------------------|------|-----|------|
|             |                         | EtherG1    | EtherG2    | EtherG3    | EtherG4    | EtherG5    | EtherG6    |       |
| Feature-based | Handcrafted             | 74.29 ± 4.40 | 79.86 ± 3.01 | 83.86 ± 2.64 | 76.21 ± 2.39 | 81.35 ± 3.48 | 86.64 ± 2.17 | 80.36 |
|             | Embedding                | 69.07 ± 2.95 | 71.79 ± 3.31 | 64.43 ± 2.51 | 72.57 ± 3.29 | 73.00 ± 4.44 | 70.86 ± 2.59 | 70.29 |
| Baseline    | DeepKernel               | 72.71 ± 5.09 | 75.29 ± 3.99 | 73.57 ± 3.35 | 76.43 ± 5.08 | 73.71 ± 5.50 | 74.57 ± 4.03 | 74.38 |
| GNN-based   | Diffpool                 | 81.29 ± 3.47 | 87.00 ± 1.74 | 95.00 ± 2.49 | 93.36 ± 1.20 | 92.36 ± 2.30 | 93.79 ± 1.36 | 90.47 |
|             | U2GNN                    | 76.57 ± 3.45 | 76.57 ± 4.99 | 70.43 ± 5.57 | 75.85 ± 4.67 | 75.71 ± 5.31 | 73.00 ± 3.03 | 74.69 |
|             | Line_Graph2Vec           | 69.57 ± 3.32 | 72.43 ± 4.65 | 70.29 ± 6.11 | 73.29 ± 5.56 | 73.00 ± 3.47 | 72.57 ± 4.57 | 71.86 |
|             | Handcrafted w/ TSGN      | 75.57 ± 3.35 | 79.00 ± 3.44 | 85.43 ± 3.49 | 77.86 ± 5.47 | 81.86 ± 3.89 | 88.29 ± 3.71 | 81.34 |
|             | Handcrafted w/ Directed-TSGN | 77.50 ± 5.02 | 80.07 ± 3.42 | 85.00 ± 2.21 | 81.07 ± 2.46 | 83.35 ± 3.13 | 89.57 ± 1.87 | 82.76 |
|             | Handcrafted w/ Temporal-TSGN | 84.14 ± 2.51 | 85.71 ± 4.65 | 88.86 ± 4.18 | 81.43 ± 4.24 | 85.71 ± 3.94 | 89.43 ± 3.45 | 85.88 |
| %Increase   |                          | 13.26%  | 10.52%  | 3.54%  | 6.85%  | 5.36%  | 3.38%  | 6.87%  |
| Graph2Vec w/ TSGN |                        | 69.57 ± 3.32 | 72.43 ± 4.65 | 70.29 ± 6.11 | 73.29 ± 5.56 | 73.00 ± 3.47 | 72.57 ± 4.37 | 71.86 |
| Graph2Vec w/ Directed-TSGN |                       | 73.57 ± 4.05 | 73.00 ± 3.44 | 68.50 ± 3.30 | 73.57 ± 4.00 | 76.79 ± 2.81 | 71.50 ± 3.94 | 72.82 |
| Graph2Vec w/ Temporal-TSGN |                    | 76.43 ± 3.57 | 79.00 ± 5.15 | 73.43 ± 2.57 | 76.00 ± 5.60 | 79.29 ± 4.88 | 75.57 ± 5.55 | 76.62 |
| %Increase   |                          | 10.66%  | 10.04%  | 13.97%  | 4.73%  | 8.62%  | 6.65%  | 9.01%  |
| DeepKernel w/ TSGN |                       | 74.29 ± 5.61 | 77.57 ± 6.13 | 68.71 ± 5.73 | 75.29 ± 5.68 | 75.29 ± 2.65 | 75.86 ± 3.53 | 74.50 |
| DeepKernel w/ Directed-TSGN |                  | 73.86 ± 2.64 | 73.86 ± 4.19 | 69.71 ± 4.94 | 73.86 ± 3.56 | 77.14 ± 6.48 | 78.00 ± 4.00 | 74.41 |
| DeepKernel w/ Temporal-TSGN |               | 77.71 ± 3.65 | 78.57 ± 4.61 | 75.00 ± 4.39 | 76.14 ± 4.91 | 78.71 ± 4.58 | 77.57 ± 5.82 | 72.56 |
| %Increase   |                          | 6.88%  | 4.36%  | 1.94%  | 0.38%  | 6.78%  | 4.60%  | 3.87%  |
| Diffpool w/ TSGN |                         | 89.71 ± 3.05 | 96.29 ± 1.83 | 96.29 ± 2.65 | 91.43 ± 3.00 | 96.57 ± 1.59 | 95.57 ± 1.49 | 94.31 |
| Diffpool w/ Directed-TSGN |                        | 93.14 ± 1.78 | 92.29 ± 2.73 | 97.00 ± 1.35 | 93.43 ± 1.10 | 94.07 ± 1.36 | 94.29 ± 1.84 | 94.03 |
| Diffpool w/ Temporal-TSGN |                      | 89.29 ± 2.95 | 94.71 ± 2.12 | 96.71 ± 1.81 | 94.14 ± 3.16 | 96.14 ± 2.56 | 98.43 ± 1.35 | 94.90 |
| %Increase   |                          | 14.58%  | 10.68%  | 2.10%  | 0.84%  | 4.56%  | 4.95%  | 4.90%  |
| U2GNN w/ TSGN |                           | 75.29 ± 4.09 | 76.00 ± 4.77 | 71.29 ± 4.58 | 75.29 ± 5.11 | 76.00 ± 3.92 | 77.14 ± 2.47 | 75.17 |
| U2GNN w/ Directed-TSGN |                        | 76.00 ± 4.03 | 77.71 ± 3.84 | 72.86 ± 3.78 | 76.43 ± 3.63 | 78.57 ± 3.50 | 78.00 ± 3.63 | 76.60 |
| U2GNN w/ Temporal-TSGN |                      | 80.71 ± 3.27 | 84.71 ± 3.56 | 73.29 ± 4.47 | 79.14 ± 4.66 | 82.00 ± 2.80 | 78.00 ± 3.79 | 79.64 |
| %Increase   |                          | 5.41%  | 10.63%  | 4.06%  | 4.34%  | 8.31%  | 6.85%  | 6.65%  |

Next, for each feature representation method, the feature vectors of TN is respectively concatenated with those of different TSGNs, and then the dimension of the feature vectors is reduced to the same as that of the feature vector obtained from the original transaction network using PCA in the experiments, for a fair comparison. For each dataset, one randomly split it into 9 folds for training and 1 fold for testing. In order to accurately evaluate the quality of each classification model, in this paper, we will use $F_1$-Score as a metric,

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

where $P$ is precision and $R$ is recall. $F_1$-Score is the harmonic mean of precision and recall, so it can more comprehensively judge the pros and cons of the classification models. To exclude the random effect of fold assignment, the experiment is repeated 300 times using the Random Forest classifier and then the average $F_1$-Score and its standard deviation recorded. Moreover, Percentage Increase ($\% Increase$) is calculated to measure the performance improvement of the proposed models relative to the original model.

$$ \% Increase = \frac{F_{1\text{model}} - F_{1\text{ori}}}{F_{1\text{ori}}} \times 100\%, $$

D. Experimental Results and Analysis

According to the above setting, we conduct some experiments on the six Ethereum datasets. To investigate the effectiveness of TSGN models, for each feature extraction algorithm, the classification results are compared on the basis of TN. Firstly, we present the performance of original model based on TN. The phishing identification performance of TSGN, Directed-TSGN, Temporal-TSGN, and Multiple-TSGN, integrating TN, are then displayed respectively. Note that, the concatenated feature vectors are fed into the same classification model for a fair comparison.

1) TSGN models can enhance phishing account identification ability: TABLE IV summarises our results in terms of the metric $F_1$-Score. On the whole, compared with the transaction networks, TSGN, Directed-TSGN, and Temporal-TSGN models indeed have good performance in enhancing
the phishing address detection. TSGN proposed as the most basic model is comparable with the five baseline models based on TN, which produces slight improvement under different datasets and feature extraction methods. Line_Graph2Vec is the improved version of Graph2Vec, which is equivalent to our Graph2Vec w/ TSGN in model design and model performance. Directed-TSGN increases the performance of the original phishing identification in 26 out of 30 cases, with a relative improvement of 3.28% on average. Moreover, since different TSGNs generated by different attribute mapping strategies can capture the different perspectives of a network, one may expect that the fusion of attributes, i.e., Temporal-TSGN, can produce even better detection results. Apparently, the proposed TSGN models improve the identification performance of the TN (original) model in 29 of 30 cases, achieving the optimal result of 98.43% on EtherG6. Especially, Temporal-TSGN almost always performs best on the six datasets, which incorporating with Diffpool outperforms the results based on other baselines.

2) **Transaction feature exploration should be paralleled with topological structure learning**: It is undisputed that the quality of features extracted from transaction networks will affect the performance of downstream phishing detection model, which is clearly reflected in our experiments. It is obvious that the classification results vary widely based on five different baseline models. Handcrafted based on feature assembly is superior to the automatic feature extraction methods relied on learning model, i.e., Graph2Vec and DeepKernel, with average gains of 9.29% and 8.26% respectively. We also find that the handcrafted features elaborately could greatly boost the performance of our TSGN model, achieving a percentage increase of 13.26% on EtherG1. And then, the experimental results have turned the spotlight on the Diffpool. In this case, our TSGN models produce optimal results exceeding those of other methods. Besides, U2GNN, which tops the global graph classification leaderboard, did not achieve impressive performances, even underperforming Handcrafted. It follows that transaction feature exploiting and topological structure learning of transaction networks are equally important in the scenario of Ethereum phishing identification. The comprehensive and rational exploration of transaction features in the progress of network modeling is the main contribution for the good performance of our TSGNs. Furthermore, we find that the complex topological interaction in the transaction networks offers more insights into the model learning of phishing identification. The models on datasets of Net type outperform those on datasets of Star type in terms of the metric $F_1$-Score in 43 out of 60 cases. While, the models on datasets of Star type have great potential for improvement, which produce higher $\% Increase$ values than datasets of Net type in 11 out of 15 cases. It also shows that the interaction information between 1-hop neighbors could indeed enhance the performance of the detection models.

3) **TSGNs with attributes can effectively reduce the modeling time of large Ethereum networks**: Furthermore, we record the computational time and compare the time consumption of constructing TSGN, Directed-TSGN, and Temporal-TSGN on six Ethereum datasets. The results are presented in Table V where one can see that the time consumption of constructing TSGN, Directed-TSGN, and Temporal-TSGN only show a slight decrease in these small-scale datasets, such as EtherG1, EtherG2, EtherG4, and EtherG5. However, there is a significant decrease in computational time of Directed-TSGN compared with that of TSGN when considering the larger datasets EtherG3 and EtherG6, decreasing from 6 hundred seconds to less than 93 seconds and from 800 seconds to 200 seconds respectively. In addition, the time consumption of Temporal-TSGN presents a further reduction on the basis of Directed-TSGN. Such results suggest that Directed-TSGN and Temporal-TSGN can further enhance the performance of the algorithm for phishing account identification, while also greatly improving the efficiency of the algorithms.

4) **Multiple-TSGN performs well on the multi-edge networks**: Temporal attributes can be adopted to identify and distinguish the multiple same-directed transaction edges between a pair of nodes. Therefore, Multiple-TSGN can dispose of more complex multi-edge transaction networks via extending the scope of application of Temporal-TSGN. In this section, the performance of our TSGN is further evaluated in the case of multi-edge transaction networks, the specific statistics of which are shown in the Multi-edge collection of Table IV. As shown in Table V, experiments on the six Ethereum datasets show competitive results. We can see that the classification performance of Graph2Vec, DeepKernel, and U2GNN methods on multi-edge datasets does not show much difference compared with the cases as shown in Table IV. Interestingly, Multiple-TSGN presents its capacity to integrate information of Multi-edge networks under the Handcrafted method, achieving good performance of 93.24% on average, and Diffpool still preserves the relatively high detection results of 95.86% and 96.71% on EtherG3 and EtherG4, respectively. Besides, compared with TN and TSGNs models, it generates a relative improvement of 16.03% and 8.57% on average, respectively, in terms of the $\% Increase$ metric. On the other hand, Multiple-TSGN increases the performance of the original TN model in all cases, with the $\% Increase$ (TN) ranging from 2.13% to 16.03%. It follows that our Multiple-TSGN model has a considerable performance when dealing with large-scale transaction networks with complex structural attributes. Overall, TSGN conducted on multi-edge cases can further improve the representation capability of the classification methods.

| Dataset | TSGN | Directed-TSGN | Temporal-TSGN |
|---------|------|---------------|---------------|
| EtherG1 | 2.42 | 2.35          | 2.29          |
| EtherG2 | 3.29 | 2.62          | 2.60          |
| EtherG3 | 605.79 | 93.28        | 71.62         |
| EtherG4 | 3.58 | 3.45          | 3.31          |
| EtherG5 | 6.54 | 5.92          | 5.56          |
| EtherG6 | 801.93 | 202.49       | 165.96        |

**TABLE V**

**TIME CONSUMPTION (SEC.) OF CONSTRUCTING TSGN, DIRECTED-TSGN, AND TEMPORAL-TSGN.**
TABLE VI
Classification performance of Multiple-TSGN model on multiple-edge transaction networks.

| Model               | Method         | Handcrafted | Graph2vec | DeepKernel | Diffpool | U2GNN |
|---------------------|----------------|-------------|-----------|------------|----------|-------|
| EtherG1             | 95.00 ± 2.65   | 79.57 ± 3.84 | 79.43 ± 3.01 | 91.71 ± 2.37 | 81.14 ± 4.46 |
| EtherG2             | 95.28 ± 2.39   | 76.86 ± 6.99 | 79.00 ± 4.04 | 91.57 ± 2.07 | 83.43 ± 3.95 |
| EtherG3             | 92.29 ± 2.73   | 70.00 ± 4.19 | 80.57 ± 6.83 | 95.86 ± 1.49 | 81.86 ± 3.20 |
| EtherG4             | 92.71 ± 3.16   | 74.86 ± 6.13 | 78.86 ± 5.34 | 96.71 ± 1.92 | 77.86 ± 5.12 |
| EtherG5             | 93.00 ± 2.82   | 73.14 ± 6.18 | 78.29 ± 5.74 | 90.00 ± 1.92 | 78.43 ± 5.01 |
| Avg. Multiple-TSGN  | 91.14 ± 2.70   | 71.43 ± 5.96 | 77.00 ± 3.16 | 88.57 ± 3.32 | 79.57 ± 3.32 |
| Avg. TSGNs (best)   | 93.24          | 74.31       | 78.86        | 92.40       | 80.38     |
| Avg. TN             | 85.88          | 76.62       | 77.26        | 94.90       | 79.64     |
| % Increase (TSGNs)  | 8.57%          | -3.01%      | 2.07%        | -2.63%      | 0.93%     |
| % Increase (TN)     | 16.03%         | 5.72%       | 6.02%        | 2.13%       | 7.62%     |

Fig. 8. Visualization of a real transaction network and its TSGNs.

E. Visualization

Moreover, for a better demonstration of network structures, we provide the visualization of real example network and its TSGNs as shown in Fig. 8. Fig. 8(a)-(b) show the transaction networks of different types, following with TSGN, Directed-TSGN, and Temporal-TSGN shown in (c)-(e) which are mapped from (b) the net-form transaction network. Multiple-TSGN in (f) is constructed based on the multi-edge net-
the network structure of Directed-TSGN by the transaction temporal attribute, where some edges disappear such as (9,14), (16,14) and (14,1) and forth. Multiple-TSGN is much more complex than the other TSGNs, providing more interaction information from different accounts around the target node, benefiting the phishing detection.

As a simple case study, we utilize the t-SNE to visualize the results of classification on EtherG1 dataset based on Diffpool method to verify the effectiveness of our TSGN model. Here, we choose the Directed-TSGN to visualize since it produces the good performance that enhances the detection performance of Diffpool most, with %Increase of 14.58%. As shown in Fig. 9 the structural features are located in different places by utilizing t-SNE. The left shows the original detection result using Diffpool with TN model, while the right depicts the optimized distribution of the same dataset using Diffpool with Directed-TSGN. One can see that the transaction networks in EtherG1 dataset can indeed be distinguished by the original features of Diffpool, but it appears that the distinction of networks could become more explicit after TSGN mapping with direction, demonstrating the effectiveness of our Directed-TSGN model. It also suggests that our TSGN model provides more predominant features contrasted with original TN model.

In this paper, we propose a novel transaction subgraph network (TSGN) model for phishing detection. By introducing different mapping mechanisms into the transaction networks, we built TSGN, Directed-TSGN, Temporal-TSGN models to enhance the classification algorithms. Compared with the baselines only based on the TN, our TSGNs indeed provide more potential information to benefit the phishing detection. By comparing with the TSGN, Directed-TSGN and Temporal-TSGN are of the controllable scale and indeed have much lower time cost, benefiting the network feature extraction methods to learn the network structure with higher efficiency. Experimental results manifest that, combined with network representation algorithms, these TSGN models can capture more features to enhance the classification algorithm and improve phishing nodes identification accuracy in Ethereum. In particular, when integrating the appropriate feature representation methods, such as Handcrafted and Diffpool, TSGNs can achieve the best results on all datasets.

In the future, we will extend and improve our model for suit more blockchain data mining tasks, such as Pondzi scheme detection, transaction tracking, etc.

ACKNOWLEDGMENT

This work was partially supported by the Key R&D Program of Zhejiang under Grant No. 2022C01018, by the National Natural Science Foundation of China under Grant No. U21B2001 and 61973273, by the Zhejiang Provincial Natural Science Foundation of China under Grant No. LR19F030001, and by the National Key R&D Program of China under Grant No. 2020YFB1006104.

REFERENCES

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” Decentralized Business Review, vol. 1, 2008.
[2] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” Ethereum project yellow paper, vol. 1, 2014.
[3] W. Chen, X. Guo, Z. Chen, Z. Zheng, and Y. Lu, “Phishing scam detection on ethereum: Towards financial security for blockchain ecosystem,” in IJCAI, 2020, pp. 4506–4512.
[4] M. Khonji, Y. Iraqi, and A. Jones, “Phishing detection: a literature survey,” IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2091–2121, 2013.
[5] Y. Huang, H. Wang, L. Wu, G. Tyson, X. Luo, R. Zhang, X. Liu, G. Huang, and X. Jiang, “Understanding (mis) behavior on the eosio blockchain,” Proceedings of the ACM on Measurement and Analysis of Computing Systems, vol. 4, no. 2, pp. 1–28, 2020.
[6] L. Ren, Z. Peng, Y. Liu, J. Li, F. Xie, and Z. Zheng, “Phishing scams detection in ethereum transaction network,” ACM Transactions on Internet Technology (TOIT), vol. 21, no. 1, pp. 1–16, 2020.
[7] F. Poursafaei, G. B. Hamad, and Z. Zulkic, “Detecting malicious ethereum entities via application of machine learning classification,” in 2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRANS). IEEE, 2020, pp. 120–127.
[8] H. Hu and Y. Xu, “Sesguard: Deep scam detection for ethereum smart contracts,” arXiv preprint arXiv:2105.10426, 2021.
[9] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, and Z. Zheng, “Who are the phishers? phishing scam detection on ethereum via network embedding,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020.
[10] I. Alarab, S. Prakoonwit, and M. I. Nacer, “Competence of graph convolutional networks for anti-money laundering in bitcoin blockchain,” in Proceedings of the 2020 5th International Conference on Machine Learning Technologies, 2020, pp. 23–27.
[11] Q. Xuan, J. Wang, M. Zhao, J. Yuan, C. Fu, Z. Ruan, and G. Chen, “Subgraph networks with application to structural feature space expansion,” IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 6, pp. 2776–2789, 2021.
[12] S. Li, F. Xu, R. Wang, and S. Zhong, “Self-supervised incremental deep graph learning for ethereum phishing scam detection,” arXiv preprint arXiv:2106.10176, 2021.
[13] Z. Yuan, Q. Yuan, and J. Wu, “Phishing detection on ethereum via learning representation of transaction subgraphs,” in International Conference on Blockchain and Trustworthy Systems. Springer, 2020, pp. 178–191.
[14] X. Liu, Z. Tang, P. Li, S. Guo, X. Fan, and J. Zhang, “A graph learning based approach for identity inference in dapp platform blockchain,” IEEE Transactions on Emerging Topics in Computing, 2020.
[15] Y. Zhuang, Z. Liu, F. Qian, Q. Liu, X. Wang, and Q. He, “Smart contract vulnerability detection using graph neural networks,” in Proceedings of the 2020 29th International Joint Conference on Artificial Intelligence, 2020, pp. 3283–3290.
[16] D. D. F. Maesa, A. Marino, and L. Ricci, “Detecting artificial behaviours in the bitcoin users graph,” Online Social Networks and Media, vol. 3, pp. 63–74, 2017.
[17] C. C. Abay, C. G. Akcora, Y. R. Gel, M. Kantarcioğlu, U. D. Islambekov, Y. Tian, and B. Thuraisingham, “Chainnet: Learning on blockchain graphs with topological features,” in 2019 IEEE international conference on data mining (ICDM). IEEE, 2019, pp. 946–951.
[18] D. Lin, J. Wu, Q. Xuan, and K. T. Chi, “Ethereum transaction tracking: Inferring evolution of transaction networks via link prediction,” *Physica A: Statistical Mechanics and its Applications*, p. 127504, 2022.

[19] S. Li, G. Gou, C. Liu, C. Hou, Z. Li, and G. Xiong, “Ttagn: Temporal transaction aggregation graph network for ethereum phishing scams detection,” in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 661–669.

[20] G. Li, M. Semerci, B. Yener, and M. J. Zaki, “Graph classification via topological and label attributes,” in *Proceedings of the 9th international workshop on mining and learning with graphs (MLG), San Diego, USA*, vol. 2, 2011.

[21] J. Wang, P. Chen, B. Ma, J. Zhou, Z. Ruan, G. Chen, and Q. Xuan, “Sampling subgraph network with application to graph classification,” *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 4, pp. 3478–3490, 2021.

[22] C. Fu, M. Zhao, L. Fan, X. Chen, J. Chen, Z. Wu, Y. Xia, and Q. Xuan, “Link weight prediction using supervised learning methods and its application to yelp layered network,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1507–1518, 2018.

[23] A. Narayanan, M. Chandramohan, L. Chen, Y. Liu, and S. Saminathan, “subgraph2vec: Learning distributed representations of rooted subgraphs from large graphs,” in *International Workshop on Mining and Learning with Graphs*, 2016.

[24] P. Yanardag and S. Vishwanathan, “Deep graph kernels,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1365–1374.

[25] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, “Hierarchical graph representation learning with differentiable pooling,” in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 4805–4815.

[26] D. Q. Nguyen, T. D. Nguyen, and D. Phung, “Universal graph transformer self-attention networks,” in *Companion Proceedings of the Web Conference 2022 (WWW ’22 Companion)*, 2022.