Blockchain Trading Graph Embedding*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

This document is a model and instructions for LaTeX. Please observe the conference page limits. [1]

II. Background

A. Blockchain and Ethereum

Satoshi Nakamoto published Bitcoin whitepaper [2] in October of 2008. As the earliest application of blockchain, Bitcoin is the most striking example of the concept of a decentralized cryptocurrency system. The production of Bitcoin depends on massive computations executing a special algorithm instead of any organization, which guarantee the consistency in the distributed ledger system.

With the development of blockchain technology, more successors have merged and tried to extend the functions related to different applications. The most significant one is Ethereum [3], providing Turing-complete smart contracts, which opens new possibilities of applications. People can develop distributed applications (DApps) with complex functions based on the Ethereum smart contract. These DApps provide the solutions for various fields other than basic transactions, such as decentralized exchange, initial coin offering (ICO), lending and so on.

However, as one of the most salient features, the anonymity of blockchain makes it is hard to find the real identities behind addresses as well as it protects the users' privacy. It makes trading analysis difficult and offers a breeding ground for frauds. According to Cointelegraph, the Ethereum network has experienced considerable phishing, Ponzi schemes and other scams events, accounting for about 10% of ICOs [4].

B. Graph Embedding

Graph analysis has been attracting increasing attention in the recent years which enables researchers to understand the various network system in a systematic manner. Graph analytic tasks can be broadly abstracted into four categories: node classification[5], link prediction[6], clustering[7] and visualization[8]. For example, node classification aims at determining the label of nodes based on other labeled nodes and the topology of the network.

Graph embedding provides an effective way to solve the graph analytics problem which convert the graph into a low dimensional space in which the graph information is preserved. In the past decade, there has been a lot of research in the field of graph embedding, and the most significant methods are factorization based methods[9][10][11], random walk based methods[12][13] and deep learning based methods[14][15].

Embedding graphs into low dimensional spaces is not a trivial task and the challenges of graph embedding depend on the problem setting. The input of graph embedding is a graph which constructed from raw data. In [16], the most studied graph embedding input is heterogeneous graph which both nodes and edges belong to multiple types respectively. Typical heterogeneous graph mainly exist in the scenarios such as community-based question answering, multimedia network and knowledge graphs. According to what we have learnt, this paper is the first work to analyze the blockchain trading graph based on graph embedding techniques.·

III. Preliminary Analysis

In this section, we first discuss the accounts and transactions in ethereum which represent the nodes and edges in ethereum trading graph respectively.

Identify applicable funding agency here. If none, delete this.

A. Accounts

In Ethereum, there are two kinds of accounts, external owned accounts and smart contracts, namely EOAs and SCs for short. The major difference between is that SCs are ruled by executable codes inside whereas EOAs are controlled by people who hold the public-private key pairs. Other than that, they do not look a whole lot different in Ethereum system. Both of them have unique addresses, the address of EOA is determined by the public key and the address of SC is encoded when SC is created. We use the terms address and account interchangeably in the remainder of this paper.

The real identities behind these addresses can be multitudinous and the anonymity of blockchain makes the identification more difficult. Generally, these addresses can be classified into different categories according to user roles on the blockchain.

1) *Miners & Mining Pools*

Similar to Bitcoin, Ethereum takes *Proof-of-Work* as its consensus protocol[1]. The **miners** are the individuals or groups who validate transaction information by solving the cryptographic puzzles. Whoever is the first to find a valid hash of block will get the reward in the form of ETH which is paid by users sending transactions.

In the early stage, most people take part in the mining process independently. With the more participants into this high profit industry, the mining competition gradually becomes fiercer. And an efficient solution is working together to solve the PoW problems. Miners with mining machines can register on a special institution named **mining pool** where aggregates all the registrants' computing power to solve mining problem and distributes the reward to the registrants according to their proportion of contributed computing power.

As of September 2018, top 3 mining pools takes more than 65% of hash rate in Ethereum[2].

2) *ERC-20 & ICO*

ERC-20 is a technical standard used for smart contracts on the Ethereum blockchain for implementing tokens[17]. It defines a common list of rules that an Ethereum token has to implement, giving developers the ability to program how new tokens will function within the Ethereum ecosystem. An ERC-20 token transfer happens in specific SCs which are called **ERC-20 token contracts**, and the transferring process will be illustrated later.

The ERC-20 token standard became popular with crowdfunding companies working on initial coin offering (ICO) cases due to the simplicity of deployment, together with its potential for interoperability with other Ethereum token standards[18]. As of July 26 2018, there were more than 103,621 ERC-20 token contracts[3]. Among the most successful ERC-20 token sales are EOS, Filecoin, Bancor, Qash, and Nebulas, raising over 60 million each[4].

Participants in the initial ICO round are **primary market investors** who buy the ERC-20 token from ERC-20 smart contracts of the crowdfunding companies. And these addresses where token sale proceeds are **token sales**.

3) *Exchanges*

The exchanges are the platforms for trading between ETH, fiat money (e.g., USD) and even other digital currency (e.g., BTC and ERC-20 tokens), which play an important role in Ethereum ecosystem. The exchanges can be categorized into centralized exchanges and decentralized exchanges (also known as DEXs).

The centralized exchange allocates a deposit address to each user who wants to make transaction in the exchange. These addresses are called **exchange deposits** and belong to the exchange since users do not have the private key of these addresses. In recharge process, user transfers coins to the given deposit address from her own wallet and these coins will be transferred to the **exchange root** address automatically. In turn, users send requests to exchange to withdraw their coins from a address called **exchange withdrawal**. And in most cases, the exchange root and exchange withdrawal mean the same address.

The DEXs are a new technology that facilitate cryptocurrency trading on a distributed ledger. Being completely on-chain, all orders interact with each other directly through the blockchain. This makes it fully decentralized, but also expensive and slow. Besides, another difference is that user will get a new address with corresponding private key when registers to the DEX, which means the address belongs to user itself instead of exchange.

4) *Phishes & Hacks*

Since virtual property transactions are now becoming increasingly commonplace and that leads to many security issues. At the same time, the frauds associated with ETH and ERC-20 tokens have also increased. We call these addresses related to frauds **phishes & hacks**.

In Ethersacan, there are more than 2500 addresses are labeled as Phish/Hack, which takes up the highest proportion. Most of them are disguised as ERC-20 token sales or DApps such as casino.

B. Transactions

In the Ethereum, transaction is the basic unit in each block which represents an action between two accounts. Transaction can be categorized as external one and internal one based on the sponsor of the transaction. A transaction is the external one if it is sent from an EOA while the internal one results from executing a smart contract due to an external transaction. And an external transaction may lead to many internal transactions[19].

Four types of transaction can be found by parsing Ethereum blocks, including *CALL*, *CRATE*, *REWARD* and *SUICIDE*. As

[1] Although Ethereum *Casper* will abandon the PoW protocol, the current Ethereum is on the third stage *Metropolis* which still mainly relies on solving hash problem.

[2] Investoon, https://investoon.com/charts/mining/eth.

[3] "Etherscan Token Tracker Page", https://etherscan.io/tokens

[4] "Token Data, data and analytics for all ICO's and tokens", https://www.tokendata.io

TABLE I
TYPICAL ACCOUNTS

Identity	Type	Description
Miner	EOA	The node who take part in the block validation process.
Mining Pool	EOA	The pooling of resources by miners, who share their processing power over a network.
ERC-20 Token Contract	SC	Smart contract that allow customers to transfer ERC-20 tokens.
Primary Market Investor	EOA	Participants in the initial ICO round.
ERC-20 Token Sale	EOA & SC	Address that allow customers to buy ERC-20 tokens.
Exchange Deposit	EOA & SC	.
Exchange Root	EOA & SC	.
Exchange Withdrawal	EOA & SC	.
Phish & Hack	EOA & SC	Fraud address related to phishing and hacks.

shown in Fig.?, ETH transferring and smart contract invoking comes with a *CALL* transaction usually and *CREATE* is used to deploy smart contracts. *REWARD* transactions appears on the head of block, which depicts the reward that block miner obtained from system. The sender of *REWARD* transaction is a special address *0x00...00*. In the *SUICIDE* transaction, the smart contract will execute destroy method to kill itself at the end of it's cycle. *CALL*, *CREATE* and *SUICIDE* transactions can be either external transactions or internal transactions since the initiator can be both EOA and SC.

Various activities are realized on Ethereum based on the above mentioned transactions. Money transfer, contract creation and contract invocation are three major activities happening on Ethereum[19]. The on-chain assets include ETH coin and ERC-20 token. An typical ETH transfer is shown in Fig. ? in where the initiate address and target address can be both EOA and SC. In the *CALL* transaction information, the amount to be transferred is a non-zero number. However, the process of ERC-20 token transfer is more complicated. The sponsor A (also called initiator) make a *CALL* transaction to the ERC-20 smart contract to tell that he want to transfer ERC-20 token to somebody B. Then the smart contract will check the request and complete the deal if A has enough ERC-20 token. Note that the target address of the *CALL* transaction is the ERC-20 SC instead of B address and the transfer value is 0 ETH since the actual transfer happens in the smart contract where the ERC-20 token is transferred from A account to B account in the smart contract inner database.

IV. ETHEREUM TRADING GRAPH ANALYSIS

Based on effective graph analytics, we can investigate more information hidden behind transaction data on blockchain. For example, by analyzing the Ethereum trading graph (ETG, for short), accounts can be classified as different identities.

In this section, we illustrate the construction of ETG and analyze some features which makes it different from other networks or graphs.

A. Problem Definition and Modeling

Generally, we consider the ETG as a directed graph $G = (V, E)$, where node $v \in V$ represents an account and $e \in E$ depicts the edge between two nodes. Actually, V is the set of all addresses in Ethereum includes both EOAs and SCs and we use the terms address, account and node interchangeably in the remainder of this paper. E is a set of ordered pairs, where $E = \{(v_i, v_j)|v_i, v_j \in V\}$. The order of an edge indicates the direction of activity (e.g., assets transfer and smart contract invocation) from v_i to v_j.

The problem can be defined as follows: given ETG $G = (V, E)$, we aim to represent each node v in a low-dimensional vector space $\vec{y_v}$. By representing ETG as a set of low dimensional vectors, graph analysis algorithms can then be computed efficiently.

Typical network embedding techniques such as random-walk based and deep learning based models use the pure network structure to map into the embedding space [20]. Our model is primarily motivated as an extension of GCNs (Graph Convolutional Networks) since it shows effectiveness for entity classification in large-scale relational data [15]. Generally, a multi-layer Graph Convolutional Network with the following layer-wise propagation rule:

$$H^{(l+1)} = \delta(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

where $H^{(l)}$ is the matrix of activations in the l-th layer, and $W^{(l)}$ is a trainable weight matrix in the l-th layer. $\delta(\cdot)$ denotes an activation function such as the ReLU$(\cdot)$ = max$(0, \cdot)$. $\tilde{A} = A + I_N$ where A is the adjacency matrix of the graph G and I_N is the identity matrix. $\tilde{D}$ is a diagonal matrix which $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$.

The method can be understood as special cases of a simple differentiable message-passing framework.

$$h_i^{(l+1)} = \delta(\sum_{j\in N}\frac{1}{c_{i,j}}W^{(l)}h_j^{(l)}) \tag{2}$$

where $h_i^{(l)}$ is the hidden state of node v_i in the l-th layer of the neural network. And c_{ij} is a problem-specific normalization constant which can defined in advance such as $c_{i,j} = \sqrt{d_i d_j}$ where d_i is the degree of node v_i.

The approach outperforms other methods such as deepwalk [12] in experiments on citation networks and knowledge graph dataset. However, we found that using such GCN model directly achieves poor effect on ETG which has many different properties from traditional networks (such as social media networks and citation graph). It brings the following challenges.

- In the original ETG, there are different relations such as assets transfer and smart contract invocation. Those relations are radically different from one another and can not be measured in a uniform weighted graph.

- Even taking a single relation graph, there are multiple edges between two nodes. For instance, repeated transactions between the same account pairs often happen. A simple method is to merge them and it will lose some information.
- Asymmetric

B. Multi-relations

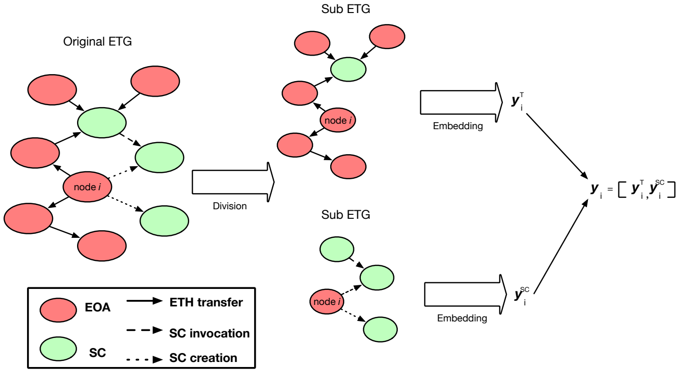


Fig. 1. Example of a figure caption.

In ETG, edges stand for different activities such as money transfer, contract creation and contract invocation, which can not be measured in a uniform weight model. For example, a weight of assets transfer maybe the ETH amount, however an invocation to smart contract does not have such numerical value. This inspired us to divide the raw ETG into different relation graphs.

Relational Graph Convolutional Networks (rGCNs) is proposed to develop an encoder model for edges in the relational graph [21]. The propagation model can be expressed as

$$h_i^{(l+1)} = \delta(\sum_{r\in R}\sum_{j\in N_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l)} + W_0^{(l)}h_i^{(l)}) \tag{3}$$

where $r \in R$ represents a kind of relation and N_i^r denotes the set of neighbor indices of node v_i under relation r. Besides, single self-connection is introduced as a special relation type to each node.

In return, we divide the edge set into four relations, CALLs with ETH, CALLs without ETH, CREATIONs and REWARDs, according to the their transaction type. Note that the ERC-20 token transfers are categorized as CALLs without ETH which includes normal smart contract invocations as well. The reason is that ERC-20 token transfer is a kind of contract invocation and the transaction value is 0 in an ERC-20 *CALL* transaction. Another reason is that even converting some ERC-20 tokens into ETH is available, the exchange-rate fluctuations make the unification meaningless.

C. Time-density

Even in a specific relation graph, there are repeated edges between the same node pairs. This occurs quite naturally since an account may transfer or invoke to another account repeatedly.

Note that these activities are located at different time intervals along the time axis which are characterized by the block height. Intuitively, a simple solution is to merge those edges by weight summation and it will lose time information.

Here we introduce an index named time-density which can be represented as strictly increasing function of block height variance.

$$\tau_{ij}^r = g(var(\mathrm{bn}_{ij}^r)) \tag{4}$$

where bn_{ij}^r is the block height set of relation r between node v_i and v_j. And the new adjacency matrix in relation r can be represented as

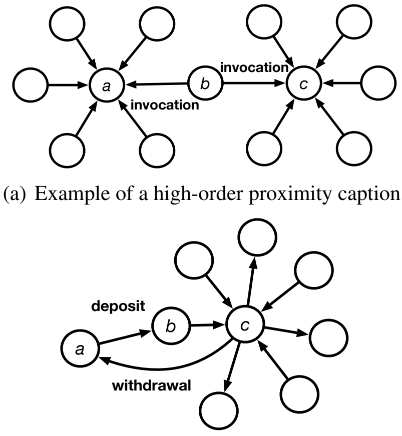
$$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}V \tag{5}$$

where $V_{ij} = \tau_{ij}$.
TBA

D. High Order and Asymmetric Proximity

In weighted graph, the edge weight A_{ij} in adjacency matrix A is generally treated as a measure of similarity between nodes v_i and v_j. And the higher the edge weight, the more similar the two nodes are expected to be. Edges weight A_{ij} is called *first-order proximity* between nodes v_i and v_j.

Further, *second-order* compares the neighborhood of two nodes and treat them as similar if they have a similar neighborhood [16]. Two nodes in ETG are more similar if they have similar connectivity structures instead of they are just connected by an edge with larger weight or share similar neighborhoods. As shown in Figure 3(a), nodes v_a and v_c are smart contracts and node v_b is normal user. Obviously, v_a is not adjacent to v_c but they have similar neighbor structure. Embedding models with *first-order proximity* and *second-order proximity* will keep them far apart although they have similar connection structures.



(a) Example of a high-order proximity caption.

(b) Example of an asymmetric proximity caption.

Fig. 2. Examples of an asymmetric proximity.

To preserve higher order proximities, the hidden layer number in our model is set as 2.

TBA

Another property of closeness in ETG is *asymmetric proximity*. For instance, as shown in Figure. 3(b), node v_a is a Ethereum investor address and node v_c is an exchange root address. Generally, edge weight can be $A_{ab} = A_{bc} = A_{ca}$ since deposit and withdrawal come in pairs in symmetric model. However, the proximity $(v_a, v_c)$is not equal with proximity (v_c, v_a) due to their asymmetric local structures.

Zhou et. proposed a scalable asymmetric proximity preserving graph embedding method based on random walk [22]. In their model, the probability that v_a arrives at v_c is far less than the one that v_c arrives at v_a, due to their asymmetric local structures. However, there is no research on asymmetric proximity in GCN model.

To preserve asymmetric proximity, we..

TBA.

V. Experiments

In this section, we conduct experiments to evaluate the performance of our method via node classification on ETG.

A. Data Collection and Graph Construction

We collect all data by running Ethereum client[5] which maintains the same copy of blockchain with all historical transactions. Note that we choose the transaction logs on Ethereum from January 1, 2018 to March 31, 2018 (xxxx external transactions and internal transactions both) as the input of graph construction since it is the most active period with various activities.

By parsing the transactions, 16,599,825 active accounts are obtained, including xxxx EOAs and xxxx SCs. Then we construct the original ETG based on these accounts and transactions.

Specially, we extend the pre-processing scheme to adapt our model. First we construct four relation graphs, which contains ETH transfer graph, contract creation graph, contract invocation graph and mining reward graph. In each graph, repeated edges between the same node pair are merged via the method introduced in section IV-C.

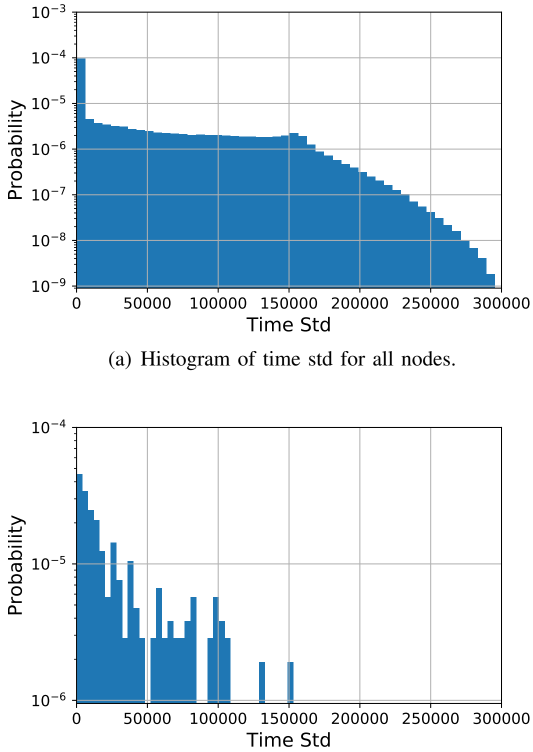
Last, a test set of accounts with label introduced before is provided to evaluate classification accuracy. It is hard to reveal the identity of addresses since the anonymity of blockchain. We obtain these labeled examples in two ways, *Etherscan*[6] and *Searchain*[7].

B. Experimental Set-Up and Baselines

Unless otherwise noted,

As a baseline for our experiments, we compare against state-of-the-art classification accuracy from XXX, XXX, XXX and XXX.

All embedding and classification programs were run on the server, which includes Intel Xeon E5 CPU with 55 processors and 128GB of memory, and the GPU used for deep learning is Nvidia 1080Ti.



(a) Histogram of time std for all nodes.

(b) Histogram of time std for hack&phish nodes.

Fig. 3. Examples.

C. Results

References

[1] M. Swan, *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc., 2015.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] V. Buterin *et al.*, "Ethereum white paper," 2013.

[4] P. Cerchiello, A. M. Toma *et al.*, "Icos success drivers: a textual and statistical analysis," University of Pavia, Department of Economics and Management, Tech. Rep., 2018.

[5] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," in *Social network data analytics*. Springer, 2011, pp. 115–148.

[6] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[7] C. H. Ding, X. He, H. Zha, M. Gu, and H. D. Simon, "A min-max cut algorithm for graph partitioning and data clustering," in *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*. IEEE, 2001, pp. 107–114.

[8] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[5] Parity Ethereum Client, https://www.parity.io/ethereum/

[6] Etherscan LabelCloud, https://etherscan.io/labelcloud

[7] Searchain, http://www.searchain.io/

[9] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, “Distributed large-scale natural graph factorization,” in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 37–48.

[10] M. Belkin and P. Niyogi, “Laplacian eigenmaps and spectral techniques for embedding and clustering,” in *Advances in neural information processing systems*, 2002, pp. 585–591.

[11] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[12] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[13] A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.

[14] D. Wang, P. Cui, and W. Zhu, “Structural deep network embedding,” in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 1225–1234.

[15] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.

[16] P. Goyal and E. Ferrara, “Graph embedding techniques, applications, and performance: A survey,” *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[17] Theethereum, “ERC-20 Token Standard - The Ethereum Wiki,” https://theethereum.wiki/w/index.php/ERC20_Token_Standard, 2017, [Online; accessed 30-August-2017].

[18] BitcoinForBeginners, “ What is an ERC20 Token,” https://www.bitcoinforbeginners.io/cryptocurrency-guide/what-is-an-erc20-token/, 2018, [Online; accessed 14-June-2018].

[19] T. Chen, Y. Zhu, Z. Li, J. Chen, X. Li, X. Luo, X. Lin, and X. Zhange, “Understanding ethereum via graph analysis,” in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 1484–1492.

[20] P. Goyal, H. Hosseinmardi, E. Ferrara, and A. Galstyan, “Capturing edge attributes via network embedding,” *arXiv preprint arXiv:1805.03280*, 2018.

[21] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, “Modeling relational data with graph convolutional networks,” in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[22] C. Zhou, Y. Liu, X. Liu, Z. Liu, and J. Gao, “Scalable graph embedding for asymmetric proximity.” in *AAAI*, 2017, pp. 2942–2948.