Pattern Analysis of Money Flow in the Bitcoin Blockchain

Natkamon Tovanich, Rémy Cazabet

Pattern Analysis of Money Flows in the Bitcoin Blockchain

Natkamon Tovanich[1] and Rémy Cazabet[2]

[1] Blockchain & B2B Platforms Chair, École Polytechnique,
Institut Polytechnique de Paris, 91120 Palaiseau, France, natkamon.tov@gmail.com,
[2] Univ de Lyon, CNRS, Université Lyon 1, LIRIS, UMR5205,
69622 Villeurbanne, France, remy.cazabet@gmail.com

Abstract. Bitcoin is the first and highest valued cryptocurrency that stores transactions in a publicly distributed ledger called the blockchain. Understanding the activity and behavior of Bitcoin actors is a crucial research topic as they are pseudonymous in the transaction network. In this article, we propose a method based on taint analysis to extract taint flows—dynamic networks representing the sequence of Bitcoins transferred from an initial source to other actors until dissolution. Then, we apply graph embedding methods to characterize taint flows. We evaluate our embedding method with taint flows from top mining pools and show that it can classify mining pools with high accuracy. We also found that taint flows from the same period show high similarity. Our work proves that tracing the money flows can be a promising approach to classifying source actors and characterizing different money flow patterns.

Keywords: Bitcoin, money flow, taint analysis, graph embedding

1 Introduction

Bitcoin is the oldest and most used cryptocurrency, attracting broad interest from the general public and researchers. In contrast to traditional financial networks, transactions can be observed by anyone on the public blockchain, on which users exchange Bitcoins pseudonymously. This data allows researchers to study economic activities in fine detail. One of the objectives of those research is to understand how the Bitcoin socio-technical system works, particularly, 1) Who are the important actors of the Bitcoin economy? [19,20,22]; 2) How is the network of transactions organized? [19,27,34]; and 3) How to identify and track illegal activity? [7,4,35].

Tracing the flow of money—where the money goes, to whom, and when—is also an essential task in cryptocurrencies and critical for financial forensics to trace money from suspicious sources and characterize different users' behaviors. We investigate two main questions regarding the relationship between the money source and subsequent transactions: 1) Does money flow differently in the Bitcoin network depending on its source?; and consequently, 2) Can we characterize a source actor given the observation of the flow of its coins in the network?

We propose an original way to synthesize the money flow from a given source into a concise dynamic network called a *taint network*. We subsequently apply whole graph embedding methods to automatically assign taint networks to their origin actor. Beyond the demonstration that each actor has a characteristic flow allowing us to recognize its position in the network [1,22,32], our method can also be helpful for actor tracking as well as actor deanonymization tasks. The embedding of money flows from different actors is a promising feature for downstream tasks in machine learning models to classify the role of actors [15,38] or predict illegal transaction activities in the Bitcoin blockchain [35].

2 Related Work

Due to the pseudonymity of Bitcoin actors, the main research challenges concentrate on 1) deanonymizing those actors and 2) characterizing their roles in the transaction network. Early works proposed clustering heuristics to deanonymize addresses likely to belong to the same actors [6,9,29,37].

Our work focuses on identifying and characterizing actors in the transaction network. Previous works applied graph analysis and machine learning to classify the role of actors or whether the transactions are illicit or not. Most works derived a set of descriptive features (e.g., the number and frequency of transactions, in- and out- degrees, and the number of different addresses used) [2,5,11,20,23] or the high order moments of transaction time [18]. These approaches rely on the actor's behavior, which easily manipulates it to hide its activities.

Other approaches thus rely on graph motifs, i.e., the set of subgraph patterns describing the neighbors of an actor [15,28,36,38]. Due to computational reasons, those works construct the static graph features only from direct neighbors (2-motif) or neighbors of their neighbors (3-motif). They do not use the identity of these neighbors but simply numeric descriptions (e.g., the total amount sent or received and transaction fees). Node2vec has been used to embed the actor position in the address network [23]. Nonetheless, a few works include the temporal aspect to impose a temporal locality constraint on the motifs [35,36].

Contrary to these approaches, our proposed method does not rely on the actor's activity or its direct neighbors but on the temporal network describing the whole flow of coins sent by an actor. We rely on the principle of tainted flows to trace the coins from source actors. Taint analysis has been used most notably in the context of tracking money from illegal sources [1,3,8,24,32]. However, those works mainly focus on the destination of tainted coins. Our work expands this approach to analyze the full money flow from multiple sources. We are not merely interested in the destination of tainted coins but in characterizing the temporal networks created by those flows.

3 *Taint Flow* extraction

Our objective in this work is to design a new method to characterize a **bitcoin source** based on the flow of its coins in the transaction network. The underlying

hypothesis is that the way a coin travels in the transaction network is characteristic of its source. However, actors reached by tainted coins after an indefinite time and coins diffused in a more considerable amount of bitcoins in a transaction cannot be characteristics of their origin. After presenting the tainting process on Bitcoin's blockchain, we thus introduce two principles to keep only relevant information in the flow: *dissolved* coins and *tag actors*.

3.1 Bitcoin taint flow

Bitcoin uses the unspent transaction output (UTXO) transaction model [25]. According to this model, transactions do not transfer money from one account to another. Instead, each output—each UTXO—of the transaction represents an amount of coin belonging to a known bitcoin address—a cryptographic public key. The rightful owner of the UTXO uses the corresponding private key to claim the money. It spends the UTXO(s) by signing them as input(s) in a new transaction and sends new UTXO output(s) to recipients' addresses.

A Bitcoin *transaction network* can be modeled as a chain of UTXOs. A transaction (tx) is represented as a node. A directed edge represents a transfer of UTXO(s) from one transaction to another. We will refer to the in-edge and out-edge of tx as input ($tx.in$) and output ($tx.out$), respectively. Each UTXO edge (e) is characterized by the amount of Bitcoin ($e.amount$) and the owner's address of that UTXO ($e.address$). It also contains references to the receiving ($e.receive$) and spending transaction nodes ($e.spend$).

We define the *taint flow* as a directed acyclic graph (G_{flow}) tracing the sequence of transactions from a **source of interest** until **dissolution**. The source of interest can be one or several actors and limited to a given time interval based on the focus of the study. To construct a money flow, we recursively taint all UTXO outputs ($tx.out$) until the coins are **dissolved**.

Definition 1 (Dissolution). *We consider that a tainted coin is* ***dissolved*** *when its future positions in the transaction network will no longer be characteristic of its original position. More formally, a coin is dissolved when it is spent in a transaction with a purity value below a minimum threshold.*

Purity measure (ρ) has been used to determine when the money is dissolved and stop following the transaction outputs [8]. Purity is the percentage of tainted money from the origin transaction set, defined as:

$$\rho(tx) = \frac{\sum_{e \in tx.in} \rho(e.receive) \cdot e.value}{\sum_{e \in tx.in} e.value} \tag{1}$$

The purity of a transaction without inputs is 1 by definition because it is the root transaction in the transaction flow. In this study, we set a purity threshold $\rho_{min} = 0.001$, which means that a coin is considered dissolved when it is spent in a transaction together with 1,000 times the amount of un-tainted coins. Besides, we stop following the flow when the transaction is $>$ 1 year apart from the

Input : τ_o is a payout transaction as a seeding node of the payout flow.
Input : ρ_{min} is a minimum purity threshold.
Input : $time_{max}$ is a maximum time threshold.
Output: *edges* is the edge list of the payout flow.
$queue \leftarrow PriorityQueue([\tau_o])$;
$edges \leftarrow List()$;
while *queue is not empty* **do**
  $tx \leftarrow queue.pop()$;
  if $\rho(tx) \geq \rho_{min}$ *and* $tx.time \leq time_{max}$ **then**
    for *e in tx.out* **do**
      $e.amount_{flow} \leftarrow e.amount \times \rho(tx)$;
      $edges.append(e)$;
      $queue.append(e.spend)$;
    end
  end
end

Algorithm 1: Reward payout flow extraction

source transactions ($time_{max}$). Algorithm 1 describes the process of retrieving transaction outputs and adding them to the money flow graph.

Our algorithm applies *haircut tainting*, which assumes that the tainted money is divided equally to all output transactions in proportion to their amount [1,32].

3.2 Actors and Tag Actors

We defined actors as a set of addresses corresponding to a person, a group of persons, an organization, or any other entity owning a set of private keys to claim the ownership of UTXOs from public key addresses. A simple but effective heuristic assumes that the input addresses in a transaction should belong to the same owner [29,12]. We use the input address clustering heuristic implemented in the BlockSci library [16] that also filter CoinJoin transactions [10] to discover a set of addresses ($e.address$) belonging to the same actor (also called address cluster, $e.cluster$). When analyzing a tainted flow, the relevant information is the actors involved in this flow. Therefore, a flow is summarized as a set of transactions between actors.

A taint flow can be large and sparse, making it difficult to compare with other flows [1]. To improve on this limit, we propose to work on variants of flows in which we keep only important actors named *tag actors*.

Definition 2 (Tag Actor). *To characterize a flow, we can describe it using a subset of all encountered actors, called* ***tag actors****. Tag actors are prominent ones that are likely to stay constant in time and to be reached by many flows from different sources.*

We propose two ways of defining tag actors: 1) **frequent actors** consist in keeping a fraction of the most frequent actors; and 2) **known actors** are chosen

based on external data. In this work, we use WalletExplorer dataset [14] that provides a collection of 375 known actors, in particular services or companies, linking their addresses with the name and type of the service (e.g., exchange platform, gambling service, and marketplace).

4 Taint Flow Embedding

Since taint flows are represented as graphs, we use whole graph embedding approaches to assess the similarity of the taint flows and characterize their patterns. The principle of those methods, such as Graph2vec [26] or Anonymous walk embedding [13] is to assign a low-dimensional vector representation to each graph such as two graphs considered similar according to a chosen network structure representation are close in the resulting embedding space.

Recently, the Geo2DR methodology [31] was introduced to allow one to design custom embedding methods and construct whole graph embedding. The methodology consists of two phases: 1) induction of descriptive substructure patterns and 2) learning of vector representations. In this case, we deal with taint flows that have directed acyclic graphs, temporal nature, and different types of node labels. Therefore, we define our custom process to produce graph walks in the first phase.

4.1 Induction of Descriptive Substructure Patterns

We use a random walk-based approach to extract substructure patterns from taint flows. We compare different variants of random walks and nodes labels vocabularies:

- **RW** - Unbiased Random Walks. We generate random walks starting only from the source node, without considering weights, following edge temporal directions. A walk ends when encountering a dissolved node.
- **SPW** - Shortest Path Walk. To generate an instance of the shortest path walk, we randomly choose a leaf node (dissolved) and walk through the shortest path between the source and a dissolved node.

We prune the walks based on the following sets of tag actors:

- **All clusters**: We use no pruning and keep all actors.
- **Frequent clusters**: We keep only clusters that appear in more than 50% of all taint flows as tag actors. The objective is to increase the fraction of shared vocabulary between flows to make learning more efficient.
- **Known actors**: We use as tag actors only those known from an external source, WalletExplorer [14]. In the **name** variant, the node label corresponds to its name. In the **type** variant, the label corresponds to its type, one of exchange, wallet, service, marketplace, mixer, lending, and gambling.

We replace all mining pool clusters and names with a “mining” label to prevent the model from training the embeddings from the source actors.

Temporal pattern In this variant, we use the same methods, but we integrate the time aspect into the walks, assuming that the time to reach different parts of the network might be characteristic of the source. We use a tuple (original label, time) where time is defined as $\lfloor \log_2(\Delta t) \rfloor$, with Δt the time elapsed between the encoded transaction and the source transaction, in days. Examples of temporal pattern labels are thus: (ID: 63566, day: 7), (Name: Bitstamp.net, day: 7), or (Type: Exchange, day: 7). We use rounded log values to avoid sparse vocabulary.

4.2 Learning Vector Representations

We use Distributed Memory Model of Paragraph Vectors (PV-DM) to train the embedding of the flow. PV-DM is one of the two variations of neural network models presented in the Doc2Vec paper [17]. The model is trained to maximize the prediction accuracy of the center vocable, given the surrounding vocables. We chose the PV-DM model because it preserves the order sequence of the walk rather than predicting a bag of words in a sentence in the PV-DBOW model. In our experiment, we set a typical embedding size ($n = 128$) to compare the different labeling strategies.

5 Flow-based Actor Identification

In this section, we demonstrate how the taint flow embeddings can be leveraged to identify Bitcoin actors from transaction sources. We extract taint flows from Bitcoin mining pools and experiment with two actor-disambiguation tasks: 1) identification of source actors using supervised learning and 2) automatic discovery of actors based on clustering. Finally, we evaluate the capacity of the embedding to differentiate between temporal origins.

5.1 Taint Flows of Bitcoin Mining Pools

We focus our experiment on the identification of mining pools. Mining pools are among the most important actors in the Bitcoin ecosystem. They correspond to companies that regroup the activity of various miners—from individuals to mining farms—under a single entity, with the prospect of sharing the mining rewards obtained to mitigate the effect of chance on their source of revenue [30,33]. We chose mining pools because they are well-studied actors, persisting long enough in time, for which we can be confident in the data for validation.

We extract taint flows from the top-3 mining pools for each month between 2013 and 2016 to represent different sources of flows from large representative mining pools. We chose this specific period because the WalletExplorer dataset stopped updating the actors used as known actors in 2016 [14].

For each top-3 pool in a month, we taint all coins received from coinbase transactions—i.e., newly generated coins—on a random day of that month as a set of source transactions. We then construct taint flows and embed them with different methods according to the process defined in the previous sections.

As a result, our dataset consists of 144 taint flows from 11 mining pools. They are of various sizes and can be too large to use traditional embedding methods on standard computers. The average number of transactions is 611,327 (sd: 362,073, median: 569,576, max: 2,181,876) while the average number of clusters is 303,955 (sd: 178,145, median: 285,991, max: 1,131,919). There are 3,697 frequent clusters existing in more than 50% of all flows.

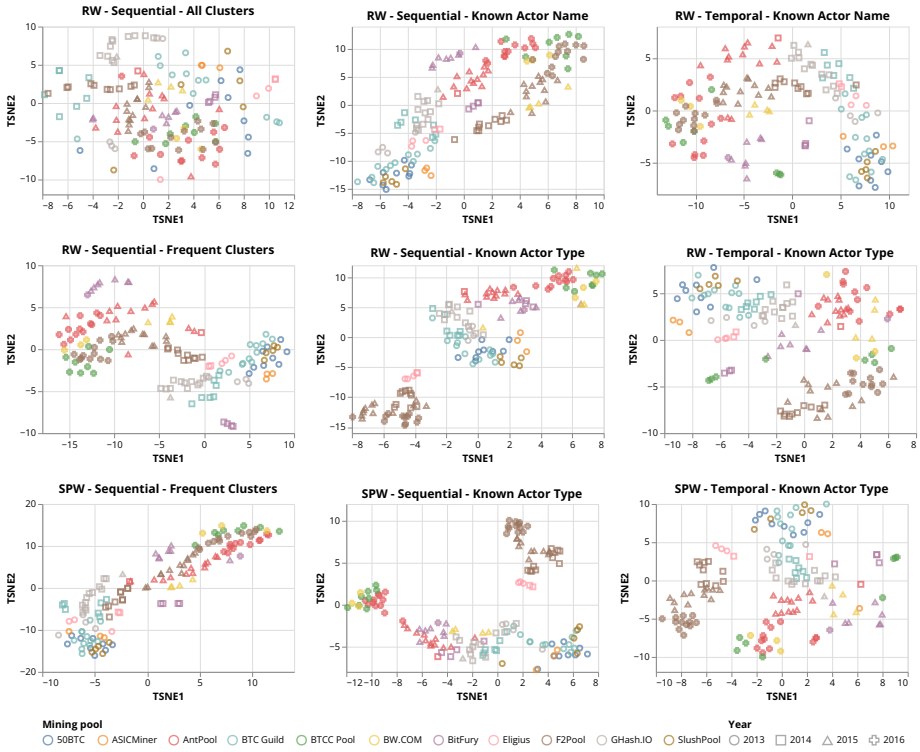

Fig. 1. T-SNE projections of selected taint flow embeddings

The result of the representation learning, embedded in two dimensions using t-SNE [21], is visualized in Fig. 1. In most cases, the multiple taint flows of the same mining pool (nodes of the same color) seem close in the embedding space. The method thus captures at least some elements of the identity of the source based on its taint flow. We also observe that the temporal aspect plays an important role and is well captured in the figures. For instance, in *RW - Temporal - Known Actor Name* and *RW - Sequential - Frequent Clusters* embedding, we see a shift—right to left—from circles to squares and triangles, and finally crosses, corresponding to the increasing years of the source.

5.2 Actor Identification Task

We train a k-Nearest Neighbor (k-NN) classification model with $k = 3$ to identify mining pools from the embedding space. k-NN is a simple model that can capture non-linear decision boundaries. We have to keep k small due to the relatively low number of observations. To evaluate the model performance, we use leave-one-out cross-validation (LOOCV), i.e., we consider that all sources are known but one and try to predict the identity of that unknown source from the others.

Baseline We define two baselines to compare the model performance with our embedding methods:

1. **Actor network features:** We extract a set of descriptive features from all cluster networks, including the number of nodes and edges, density, and degree assortativity. We also calculate nodes in- and out- degrees, clustering coefficient, and eigenvector centrality and report the minimum, 1^{st}–3^{rd} quantiles, maximum, mean, standard deviation, and mean absolute deviation for each feature.
2. **Graph2vec:** We train Graph2vec models [26] to embed frequent clusters and known actor name networks. We cannot train the model with the *All clusters* setting because the size of the graph can be enormous and make it impractical to compute the embedding.

Table 1. Evaluation of taint flow embeddings of top-3 mining pools in 2013-2016

Method	**Accuracy**		**F1-Score**		**NMI**		**ARI**		**AMI**		**Time Corr.**	
Actor network features	0.250		0.152		0.120		0.096		0.017		0.118	
1. Graph2Vec												
Frequent clusters	0.146		0.086		0.195		0.173		0.117		0.263	
Known actor name	0.299		0.242		0.127		0.103		0.057		-0.085	
2. Sequential	RW	SPW	RW	SPW	RW	SPW	RW	SPW	RW	SPW	RW	SPW
All clusters	0.479	0.313	0.386	0.201	0.274	0.281	0.252	0.227	0.114	0.068	0.298	0.185
Frequent clusters	0.771	0.681	0.665	0.526	0.333	0.367	0.315	0.335	0.160	0.172	0.459	0.635
Known actor name	0.736	0.542	0.592	0.366	0.572	0.332	0.509	0.297	0.261	0.126	0.625	0.701
Known actor type	0.764	0.688	0.646	0.552	0.425	0.591	0.408	0.544	0.180	0.277	0.282	0.511
3. Temporal	RW	SPW	RW	SPW	RW	SPW	RW	SPW	RW	SPW	RW	SPW
All clusters	0.389	0.222	0.288	0.131	0.298	0.298	0.278	0.279	0.130	0.133	0.253	0.193
Frequent clusters	0.486	0.535	0.395	0.383	0.331	0.326	0.313	0.307	0.172	0.168	0.379	0.517
Known actor name	0.583	0.521	0.450	0.350	0.453	0.461	0.402	0.411	0.209	0.204	0.453	0.491
Known actor type	0.778	0.708	0.674	0.601	0.644	0.617	0.618	0.580	0.480	0.418	0.282	0.356

Table 1 reports the Accuracy and F1 measure of the classification models. Our walk-based embeddings provide higher accuracy and F1-score than the baseline models, and pruning strategies improve performances as the *All clusters* approach consistently obtains the worst results. The best results are obtained using either *Temporal Known Actor Type* or *Sequential Frequent Clusters*. Overall,

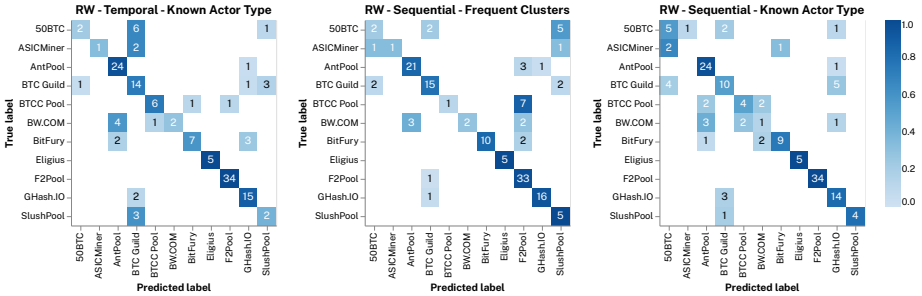

Fig. 2. Confusion matrix of the top-3 classification models

we found that *Random walks* provides better results than *Shortest Path Walks* in both sequential and temporal models.

Fig. 2 presents the confusion matrices of the top-3 classification models: *Temporal Known Actor Type*, *Sequential Frequent Clusters*, and *Sequential Known Actor Type*. We observe that most of the actors for which we have many samples (e.g., AntPool, F2Pool, Ghash.IO) are well classified. Mining pools with few examples are more prone to errors, particularly ASICMiner, which has only three occurrences in our dataset and is mostly wrongly classified. This stresses the importance of having enough learning data in future works.

5.3 Actor Clustering Task

We assess if source actors can be discovered with the unsupervised approach by training k-means clustering on flow embeddings and checking the the cluster founds with the known sources. We train k-means models from 2 to 11 clusters and select the model with the highest *Silhouette* score. Table 1 reports the clustering evaluation with three standard scores: normalized mutual information (NMI), adjusted rand score (ARI), and adjusted mutual information (AMI).

We found that *Known Actor Type* clusters are the most efficient method for sequential and temporal embeddings. In contrast to the classification task, the *Shortest Path Walk* provides a better clustering for sequential embedding. Nonetheless, the three clustering metrics are relatively low. As source time and identity are highly correlated, we suspect the clustering might mix these two aspects and form the cluster containing pools in the same period.

5.4 Time Correlation

As we have observed in Fig. 1, the vectors of taint flow also embed a notion of time. This can be explained by Bitcoin's highly dynamic ecosystem in which actors appear, disappear, rise, and fall in popularity over time. Our approach depends on its source's identity and when this flow starts. Hence, it is possible that the embedding model group actors that frequently occur at the same period be closer in the embedding space.

To quantify how embeddings capture the evolution of taint flow patterns, we compute a *time correlation* score by computing the Spearman correlation coefficient between the distance in time expressed in months and the distance in the embedding for all pairs of taint flows. The time correlation in Table 1 shows a high correlation for *Sequential Known Actor Name* The correlation is lower in temporal embeddings compared with sequential ones.

Interestingly, the temporal aspect is poorly captured when using actor types instead of their identity. The correlation of *Known Actor Type* suggests that despite the rise and fall in popularity of specific actors, the type of actors reached by a flow from a particular source tends to stay constant, as shown in Fig. 1. This result highlights the importance of using time-independent vocabulary, in this case, the role of the actors, to train the unbias embedding model.

6 Discussion and Conclusion

In this work, we propose original methods to extract taint flows that take into account the temporal aspect of the Bitcoin transaction network and represent them using graph embedding techniques. We train classification and clustering models to evaluate how the tainted flows of coins can be used to identify the source actors in the Bitcoin transaction network.

Our experiments with mining pool taint flows show that although we could not reach a perfect precision in the actor identification task, a simple supervised approach yields a high accuracy. Unsupervised clustering is less convincing at this stage but could be improved by taking time into account and increasing the number of observations. The analysis of those results highlights what makes a taint flow characteristic of its source actor: 1) the starting time is significant as actors in the Bitcoin network emerge and disappear over time, and 2) the identity of encountered actors is not the only relevant element since we can also reach a good result using the characteristics of actors, especially their roles (actor type). This stresses the importance of using labeled data to improve model performance and raises another research direction to infer actor roles from on-chain data.

Our work demonstrates the relevance of using taint flows to characterize their source. However, we can achieve a better model performance with more taint flow data and more sophisticated classification and clustering models. Additional information such as the country of origin, network centrality, and more precise actor types, could also be a direction of improvement. Our method could be applied to other cryptocurrencies or other forms of diffusion, such as information in social media, by appropriately adapting the diffusion flow's construction. In future work, we intend to apply this approach to characterize money flows in other domains, particularly illegal and cybercrime activities, as well as propose a new technique to extract and explain meaningful patterns from those flows.

Acknowledgement

This project was partly founded by BITUNAM grant ANR-18-CE23-0004.

References

1. Ahmed, M., Shumailov, I., Anderson, R.: Tendrils of crime: Visualizing the diffusion of stolen bitcoins. In: Graphical Models for Security. pp. 1–12. Springer (2019)
2. Akcora, C.G., Li, Y., Gel, Y.R., Kantarcioglu, M.: BitcoinHeist: Topological data analysis for ransomware prediction on the bitcoin blockchain. In: The 29th International Joint Conference on Artificial Intelligence. pp. 4439–4445. IJCAI (2020)
3. Balthasar, T.d., Hernandez-Castro, J.: An analysis of bitcoin laundry services. In: Nordic Conference on Secure IT Systems. pp. 297–312. Springer (2017)
4. Bartoletti, M., Lande, S., Loddo, A., Pompianu, L., Serusi, S.: Cryptocurrency scams: analysis and perspectives. IEEE Access 9, 148353–148373 (2021)
5. Bartoletti, M., Pes, B., Serusi, S.: Data mining for detecting bitcoin ponzi schemes. In: Crypto Valley Conference on Blockchain Technology. pp. 75–84. IEEE (2018)
6. Cazabet, R., Baccour, R., Latapy, M.: Tracking bitcoin users activity using community detection on a network of weak signals. In: International Conference on Complex Networks and Their Applications. pp. 166–177. Springer (2017)
7. Chainalysis Team: The 2022 Crypto Crime Report. Chainalysis Inc. (2022), https://go.chainalysis.com/2022-Crypto-Crime-Report.html
8. Di Battista, G., Di Donato, V., Patrignani, M., Pizzonia, M., Roselli, V., Tamassia, R.: Bitconeview: visualization of flows in the bitcoin transaction graph. In: IEEE Symposium on Visualization for Cyber Security. pp. 1–8. IEEE (2015)
9. Ermilov, D., Panov, M., Yanovich, Y.: Automatic bitcoin address clustering. In: IEEE International Conference on Machine Learning and Applications. pp. 461–466. IEEE (2017)
10. Goldfeder, S., Kalodner, H., Reisman, D., Narayanan, A.: When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies. arXiv preprint arXiv:1708.04748 (2017)
11. Harlev, M.A., Sun Yin, H., Langenheldt, K.C., Mukkamala, R., Vatrapu, R.: Breaking bad: De-anonymising entity types on the bitcoin blockchain using supervised machine learning. In: The 51st Hawaii International Conference on System Sciences. ScholarSpace / AIS Electronic Library (2018)
12. Harrigan, M., Fretter, C.: The unreasonable effectiveness of address clustering. In: IEEE UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld. pp. 368–373. IEEE (2016)
13. Ivanov, S., Burnaev, E.: Anonymous walk embeddings. In: International Conference on Machine Learning. pp. 2186–2195. PMLR (2018)
14. Janda, A.: Walletexplorer.com, https://www.walletexplorer.com/info
15. Jourdan, M., Blandin, S., Wynter, L., Deshpande, P.: Characterizing entities in the bitcoin blockchain. In: IEEE International Conference on Data Mining Workshops. pp. 55–62. IEEE (2018)
16. Kalodner, H., Möser, M., Lee, K., Goldfeder, S., Plattner, M., Chator, A., Narayanan, A.: Blocksci: Design and applications of a blockchain analysis platform. In: 29th USENIX Security Symposium. pp. 2721–2738. USENIX Association (2020)
17. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning. pp. 1188–1196. JMLR (2014)
18. Lin, Y.J., Wu, P.W., Hsu, C.H., Tu, I.P., Liao, S.w.: An evaluation of bitcoin address classification based on transaction history summarization. In: IEEE International Conference on Blockchain and Cryptocurrency. pp. 302–310. IEEE (2019)
19. Lischke, M., Fabian, B.: Analyzing the bitcoin network: The first four years. Future Internet 8(1), 7 (2016)

20. Liu, X.F., Ren, H.H., Liu, S.H., Jiang, X.J.: Characterizing key agents in the cryptocurrency economy through blockchain transaction analysis. EPJ Data Science 10(1), 21 (2021)
21. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research 9(11) (2008)
22. Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., Savage, S.: A fistful of bitcoins: characterizing payments among men with no names. Communications of the ACM 59(4), 86–93 (2016)
23. Michalski, R., Dziubałtowska, D., Macek, P.: Revealing the character of nodes in a blockchain with supervised learning. IEEE Access 8, 109639–109647 (2020)
24. Möser, M., Böhme, R., Breuker, D.: An inquiry into money laundering tools in the bitcoin ecosystem. In: APWG eCrime researchers summit. pp. 1–14. IEEE (2013)
25. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Tech. rep. (2008), http://bitcoin.org/bitcoin.pdf
26. Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., Jaiswal, S.: graph2vec: Learning distributed representations of graphs. arXiv preprint arXiv:1707.05005 (2017)
27. Nerurkar, P., Patel, D., Busnel, Y., Ludinard, R., Kumari, S., Khan, M.K.: Dissecting bitcoin blockchain: Empirical analysis of bitcoin network (2009–2020). Journal of Network and Computer Applications 177, 102940 (2021)
28. Ranshous, S., Joslyn, C.A., Kreyling, S., Nowak, K., Samatova, N.F., West, C.L., Winters, S.: Exchange pattern mining in the bitcoin transaction directed hypergraph. In: International Conference on Financial Cryptography and Data Security. pp. 248–263. Springer (2017)
29. Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and Privacy in Social Networks, pp. 197–223. Springer (2013)
30. Romiti, M., Judmayer, A., Zamyatin, A., Haslhofer, B.: A deep dive into bitcoin mining pools: An empirical analysis of mining shares. In: Workshop on the Economics of Information Security (2019)
31. Scherer, P., Lio, P.: Learning distributed representations of graphs with geo2dr. In: ICML Workshop on Graph Representation Learning and Beyond (2020)
32. Tironsakkul, T., Maarek, M., Eross, A., Just, M.: Probing the mystery of cryptocurrency theft: An investigation into methods for cryptocurrency tainting analysis. In: Cryptocurrency Research Conference (2019)
33. Tovanich, N., Soulié, N., Heulot, N., Isenberg, P.: The evolution of mining pools and miners' behaviors in the bitcoin blockchain. IEEE Transactions on Network and Service Management (2022)
34. Vallarano, N., Tessone, C.J., Squartini, T.: Bitcoin transaction networks: an overview of recent results. Frontiers in Physics p. 286 (2020)
35. Weber, M., Domeniconi, G., Chen, J., Weidele, D.K.I., Bellei, C., Robinson, T., Leiserson, C.E.: Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. In: KDD Workshop on Anomaly Detection in Finance (2019)
36. Wu, J., Liu, J., Chen, W., Huang, H., Zheng, Z., Zhang, Y.: Detecting mixing services via mining bitcoin transaction network with hybrid motifs. IEEE Transactions on Systems, Man, and Cybernetics: Systems (2021)
37. Zhang, Y., Wang, J., Luo, J.: Heuristic-based address clustering in bitcoin. IEEE Access 8, 210582–210591 (2020)
38. Zola, F., Eguimendia, M., Bruse, J.L., Urrutia, R.O.: Cascading machine learning to attack bitcoin anonymity. In: IEEE International Conference on Blockchain. pp. 10–17. IEEE (2019)