Characterizing Entities in the Bitcoin Blockchain

Marc Jourdan
IBM Research, Singapore
10 Marina Boulevard
18983 Singapore
mjourdan@sg.ibm.com

Sebastien Blandin
IBM Research, Singapore
10 Marina Boulevard
18983 Singapore
sblandin@sg.ibm.com

Laura Wynter
IBM Research, Singapore
10 Marina Boulevard
18983 Singapore
lwynter@sg.ibm.com

Pralhad Deshpande
IBM Research, Singapore
10 Marina Boulevard
18983 Singapore
pralhad@sg.ibm.com

***Abstract*—Bitcoin has created a new exchange paradigm within which financial transactions can be trusted without an intermediary. This premise of a free decentralized transactional network however requires, in its current implementation, unrestricted access to the ledger for peer-based transaction verification. A number of studies have shown that, in this pseudonymous context, identities can be leaked based on transaction features or off-network information. In this work, we analyze the information revealed by the pattern of transactions in the neighborhood of a given entity transaction. By definition, these features which pertain to an extended network are not directly controllable by the entity, but might enable leakage of information about transacting entities. We define a number of new features relevant to entity characterization on the Bitcoin Blockchain and study their efficacy in practice. We show that even a weak attacker with shallow data mining knowledge is able to leverage these features to characterize the entity properties.**

***Index Terms*—Bitcoin, Privacy, Pattern classification, Bipartite graph.**

I. Introduction

Bitcoin [18] stands out as the first global decentralized currency, and has seen spectacular growth recently, as illustrated by the exponential shape of the value of a transaction fee over the year 2017, see Figure 1. The underlying Bitcoin data structure, the Blockchain, has been perceived as a catalyst to the emergence of broad decentralized applications, from crypto-currency exchanges, to decentralized autonomous organizations (DAO), or tokens, see [1] for a comprehensive review. However, one of the most compelling applications remains the promise of a global decentralized currency, supporting large portions of the global economy.

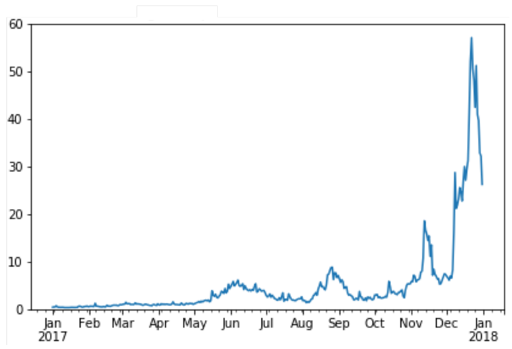

Figure 1. **Transaction fee in USD:** over the year 2017.

A. Emergence of a decentralized global currency

As presented in [13], after only a few years, the bitcoin network has emerged to reflect a complex global payment system with new forms of players representing the traditional actors of the established financial infrastructure. Global transactions between Bitcoin exchanges, illustrated in Figure 2, have drastically complexified in the last 5 years. It is important for regulatory purposes as well as for the development of new crypto-currency paradigms to understand Bitcoin anonymity properties. Indeed, a healthy economy requires application of the *rule of law* on financial transactions [6], which usually entails traceability and transparency on identities. On the other hand, in order to attract users, privacy considerations associated with crypto-currency have to be adequate and competitive with privacy properties of transactions using fiat money.

We say that a transaction medium is *private* if no information on the transacting entity is revealed by the transaction graph. In the Bitcoin context, each individual transaction is publicly associated with a pseudonymous identity. Hence the strongest possibly guarantee is that the set of transactions associated with a given pseudonymous identity does not reveal information about the hidden transacting entity.

The main contribution of this work is the illustration that patterns of transactions involving the transacting entity but also patterns of neighboring transactions (on the transaction graph) are characteristic of the entity, in that these patterns can be turned into a fingerprint of transacting entity classes. We further give evidence that using these neighboring transaction patterns allows reaching state-of-the-art entity classification results.

B. Related work

Because the Bitcoin transaction graph is completely public it is clear that Bitcoin flows can be traced, although this is limited by merging transactions [24]. Recent analysis of ransomware attacks and associated bitcoin transactions can be found in [9], [16], [21].

Furthermore, joining Bitcoin transaction with off-network activities can contribute to linking identities of certain Bitcoin transactions [23], for instance the authors of [10] link Bitcoin users with Tor hidden services.

A number of studies are focused on the problem of *address clustering* consisting of identifying the address set associated with a transacting entity, and typically rely on heuristics

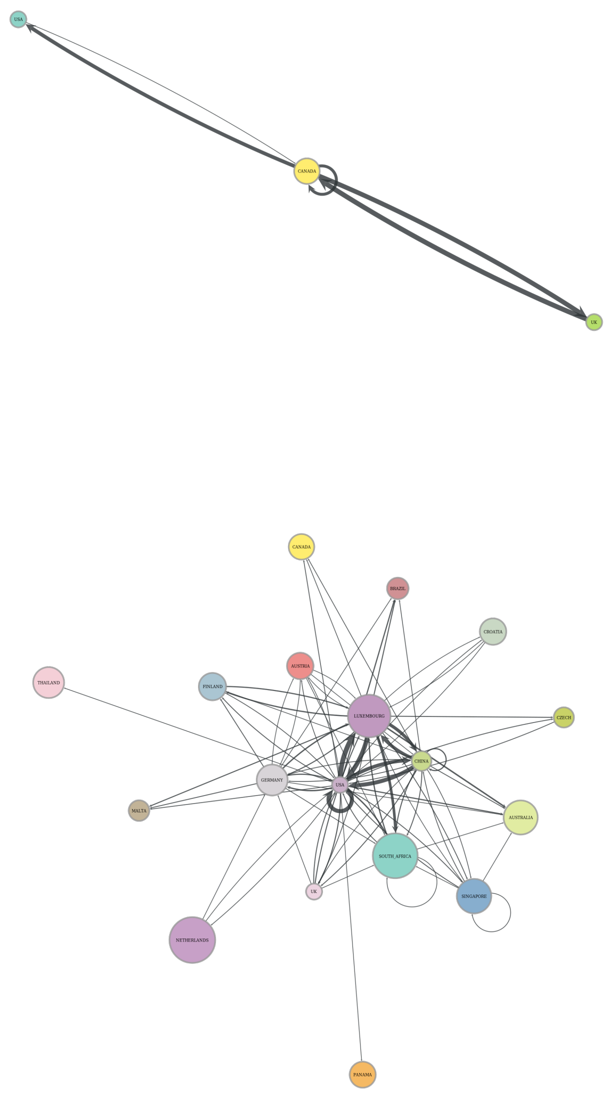

Figure 2. **Bitcoin exchanges transactions:** for the month of March 2013 (top) and March 2018 (bottom). The width of the edge is proportional to the total value of transactions during the month for the associated exchange pair (the size of the nodes is arbitrary).

motivated by properties of the Bitcoin Blockchain protocol, such as the requirement of a unique signature for transaction inputs, see for instance [4], [14] and references therein for more details. The authors of [8] highlight that these methods crucially depend on address re-use behavior.

Another attack motivated by the Bitcoin protocol involves the inference of peer-to-peer communication structure [5] and information leakage from the message dissemination pattern. A different type of attack is presented in [20], where the authors show that statistical analysis of bloom filters could help to identify the set of addresses owned by Bitcoin wallets. It has also been shown that patterns of newly minted bitcoins could play a role in revealing information of certain Bitcoin users [15].

Thematically closer to our work, several studies have considered the extent to which data mining methods can be applied to the entity characterization problem, with for instance the use of transaction-specific features in [26], able to achieve 70% accuracy for classifying entities into several types. In [22], the authors introduce the notion of transaction motifs with application to the detection of bitcoin exchanges, and achieve greater than 80% accuracy.

C. *Contributions of this work*

Motivated by the success of [22], we consider the more general problem of classifying entities into multiple classes, based on extended transaction neighborhood properties. The study of transaction graph neighborhood structure is promising for at least two reasons.

First, the use of graph network features has been shown to be efficient for graph learning problems [27]. The promise of such methods is illustrated by the spread of graph databases and related applications [25]. The authors of [19] for example are able to successfully re-identify nodes from a noisy graph structure provided as part of a Kaggle contest.

Second, if neighboring network transactions are confirmed to be informative of entity identities, they constitute a fundamental limit to Bitcoin privacy guarantees. Indeed, while an individual has control of his personal address usage and transaction patterns, he does not have control of the behavior of the entities he is transacting with. While services such as CoinJoin serve to limit information leakage of Bitcoin flows via the merging of individual transactions [17], there is at this stage no service providing full neighborhood obfuscation.

The main contribution of this work are the following:

- definition of novel features for entity classification from a graph neighborhood perspective,
- analysis of the performance of various classification methods,
- discussion of the implications of these entity characterization results in the context of Bitcoin anonymity.

The structure of the paper is as follows. In Section II we define the graph model we propose for the Bitcoin Blockchain. In Section III, we present our classification models as well as details of the graph neighborhood features developed. Finally we provide in Section IV numerical results that demonstrate the effectiveness of entity characterization using actual Bitcoin Blockchain data. Section V provides concluding remarks.

II. Blockchain graph model

The Bitcoin Blockchain is a succession of blocks $\mathscr{B}$. Each block $b \in \mathscr{B}$ contains a set of transactions $T(b) = \{t_i\}_i \subset \mathscr{T}$. We first present a bipartite address-transaction graph model, and then explain how we derive a discrete-time entity-transaction graph model with features relevant to the entity characterization program.

A. Address-transaction graph

We model the Bitcoin Blockchain as a directed weighted bipartite graph $\mathscr{H} = (\mathscr{A}, \mathscr{T}, \mathscr{L})$, where $a \in \mathscr{A}$ represents an address, $t \in \mathscr{T}$ represents a transaction between addresses, and $l \in \mathscr{L}$ represents an edge between an address and a transaction. We partition the edge set into edges incoming to a transaction, and edges outgoing from a transaction, as follows: $l \in \mathscr{L} = \mathscr{I} \cup \mathscr{O}$, where $\mathscr{I}$ stands for input edges, i.e. incoming edges of a vertex t, and $\mathscr{O}$ for output edges, i.e. outgoing edges of a vertex t. Multiple edges can exist between a pair $(a, t) \in \mathscr{A} \times \mathscr{T}$.

For each transaction $t \in \mathscr{T}$, we define $I(t) \subset \mathscr{I}$ and $O(t) \subset \mathscr{O}$ as the corresponding edges between the transaction and the addresses associated. For each $i \in \mathscr{I}$ (resp. $o \in \mathscr{O}$), we uniquely define the associated transaction $t(i)$ (resp. $t(o)$) and address $a(i)$ (resp. $a(o)$); these are well defined because the graph is bipartite.

We ignore the address and transaction fields that are specific to the protocol or not relevant to our study, see for instance [2] for related graph models for permissioned blockchains, or [3] for a more fundamental analysis of a transaction graph model and inherited protocol properties. In this work, we consider the following vertex and edge properties:

- *edge*: inputs, $\mathscr{I}$, (resp. outputs, $\mathscr{O}$) are represented by their amount in BTC, $v(i)$ for $i \in \mathscr{I}$ ($v(o)$ for $o \in \mathscr{O}$), sent (resp. received) by an address, $a(i)$ (resp. $a(o)$), for a given transaction, $t(i)$ (resp. $t(o)$),
- *transaction vertex*: a transaction contained in a block $b(t)$ has a fee, $f(t)$, and a time $\tau(t)$, corresponding to the time when the block it belongs to is validated, $\tau(t) = \tau(b(t))$.
- *address vertex*: an address has a creation date and a balance.

In the next section we explain how we derive the entity-transaction graph from the address-transaction graph.

B. Entity-transaction graph

In the Bitcoin Blockchain a user may employ several addresses. We therefore introduce the concept of an "entity" where entity, e, is fully characterized by a set of addresses $A(e) = \{a_i^{(e)}\}_i$, which can be interpreted as a logical user. We discuss subsequently the various methods for defining these sets, such as the common spending heuristic.

The entity-transaction graph is a directed weighted bipartite graph $\mathscr{G} = (\mathscr{E}, \mathscr{T}_e, \mathscr{L}_e)$ of entities and associated transactions. Let $e \in \mathscr{E}$ denote an entity and $t \in \mathscr{T}_e$, a transaction between entities. Entities and transactions are connected by edges $l \in \mathscr{L}_e = \mathscr{I}_e \cup \mathscr{O}_e$, where $\mathscr{I}_e$ stands for input edges, and $\mathscr{O}_e$ represents output edges.

As with addresses, for each transaction $t \in \mathscr{T}_e$ we define $I_e(t) \subset \mathscr{I}_e$ and $O_e(t) \subset \mathscr{O}_e$ as the corresponding edges between the transactions and entities. For each $i \in \mathscr{I}_e$ (resp. $o \in \mathscr{O}_e$), as for the address-transaction graph, we uniquely define the associated transaction $t(i)$ (resp. $t(o)$) and entity $e(i)$ (resp. $e(o)$). Inputs, $\mathscr{I}_e$, (resp. outputs, $\mathscr{O}_e$) represent the amount in BTC, $v(i)$ for $i \in \mathscr{I}_e$ ($v(o)$ for $o \in \mathscr{O}_e$), sent (resp. received) by an entity, $e(i)$ (resp. $e(o)$), for a given transaction. We define the set of addresses, $A(I_e(t))$ (resp. $A(O_e(t))$), associated with a set of inputs (resp. outputs). The mechanism to build the entity-transaction graph from the address-transaction graph is illustrated in Figure 3.

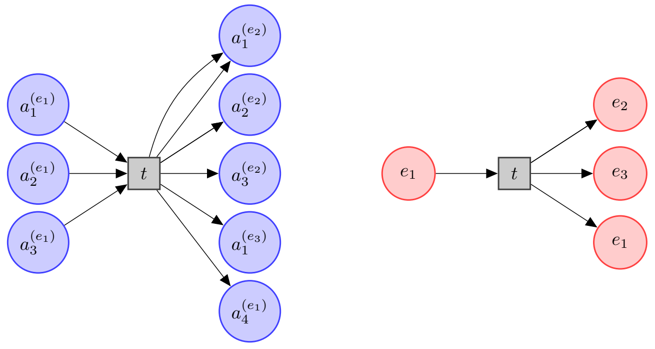

Figure 3. **Entity graph**: (right) is obtained by aggregation of the address graph (left).

The vertices and edges of the entity-transaction graph inherit the properties of the address-transaction graph by standard summation of continuous properties and aggregation of graph topology. Additionally we consider that vertices and edges of the entity-transaction graph inherit some statistics of the entity-transaction graph that are lost by summation (e.g. an entity graph edge inherits the count of address graph edges which it subsumes).

We further define a set of categories $c \in \mathscr{C}$, corresponding to entity class labels. We shall consider in the remainder of this work the entity classification problem, namely identification of the class label $c \in \mathscr{C}$ associated with an entity $e \in \mathscr{E}$, based on the address-transaction graph.

C. Discrete time graph

We apply temporal aggregation to the directed weighted bipartite graph of entities $\mathscr{G} = (\mathscr{E}, \mathscr{T}_e, \mathscr{L}_e)$ to obtain a discrete-time data structure with commensurate properties.

Definition 1 (Discrete-time operator)**.** A discrete-time aggregation operator is an operator :

$$\Delta^{(\tau_1,\tau_2)} : \mathscr{G} \mapsto \mathscr{G}^{(\tau_1,\tau_2)}$$

where time-aggregated entities and transactions $\mathscr{E}^{(\tau_1,\tau_2)}$, $\mathscr{T}_e^{(\tau_1,\tau_2)}$ are derived from $\mathscr{E}, \mathscr{T}_e, \mathscr{L}_e$ and required to satisfy the following constraints:

- Entity activity: only entities transacting during the time period are considered. $\mathscr{E}^{(\tau_1,\tau_2)} = \{e \in \mathscr{E}, \exists t \in \mathscr{T}_e, \tau(t) \in [\tau_1, \tau_2] \wedge ((e,t) \in I_e(t) \vee (t,e) \in O_e(t))\}$
- Transaction: the edges, $(e_i, e_j) \in \mathscr{T}_e^{(\tau_1,\tau_2)}$, represent the aggregation of transactions between two entities within the time window, $\{t \in \mathscr{T}_e, \tau(t) \in [\tau_1, \tau_2] \wedge e(I_e(t)) = e_i \wedge \exists o \in O_e(t), e(o) = e_j\}$.

In the following, we work with the discrete-time graph $\mathscr{G}^{(\tau_1,\tau_2)} = (\mathscr{E}^{(\tau_1,\tau_2)}, \mathscr{T}_e^{(\tau_1,\tau_2)})$ obtained by applying the discrete-time operator, with typical discrete time intervals of a day, a week, or a month.

D. *Motifs*

The notion of *motif* in the Bitcoin Blockchain was introduced in [22] as a useful concept in entity classification studies, specifically in the context of Exchange detection. The authors define the so-called 2-motif. In this work, we consider more generally the case of N-motif and present a few relevant special cases.

Definition 2 (1-motif)**.** A 1-motif is a path of length 2 on the entity-transaction graph, $(e_1, t, e_2) \in \mathscr{E} \times \mathscr{T}_e \times \mathscr{E}$, in $\mathscr{G}$.

$$motif_1(t) = \{(e_i, t, e_j)\}_{i,j}.$$

If $e_1 = e_2$ we call it a Loop 1-motif, otherwise it is called Distinct 1-motif.

The 1-motif, is a direct transaction between entities. More generally, a direct N-motif is a path of length $2N$ in the directed weighted bipartite graph starting and ending with an entity.

Definition 3 (Direct N-motif)**.** A N-motif is a path of length $2N$, $(e_1, t_1, \ldots, t_N, e_{N+1}) \in \mathscr{E} \times \mathscr{T}_e \times \cdots \times \mathscr{T}_e \times \mathscr{E}$, in $\mathscr{G}$ starting and ending with an entity. A $\text{motif}_N(t_1, \ldots, t_N) = \{(e_{i_1}, t_1, \ldots, t_N, e_{i_{N+1}})\}$ is required to satisfy the following constraint:

- Direct: at least one output from each transaction is an input to the next transaction. $\forall k \in \{1, \ldots, N-1\} \exists (o, i) \in O(t_k) \times I(t_{k+1}), a(o) = a(i)$.

If $e_1 = e_{N+1}$ we call it a Direct Loop, otherwise Direct Distinct. From this condition we deduce immediately that the transactions are ordered in time: $\tau(t_1) < \cdots < \tau(t_N)$. Considering only Direct N-motif avoids redundancy of exploration and focuses on fast flow of value. In this work, we do not consider non-Direct paths that would correspond to more complex transfer-and-hold patterns.

We illustrate the case of a 3-motif in Figure 4.

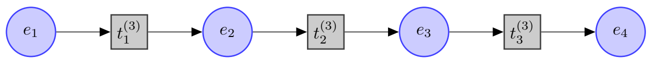

Figure 4. **3-motif**: consisting of a path of length 6 on the bipartite entity-transaction graph.

Statistics of 1,2, and 3-motifs over the dataset considered are presented in Table I, with the following entity categories: Exchange, Gambling, Mining, Service, Darknet, as per labelling from Wallet Explorer. An indication of the power of transaction graph neighborhood structure is that certain motifs dominate within certain entity categories, e.g. direct transactions with distinct entities are more characteristic of Exchanges than of other entity categories.

Motif attributes such as BTC volume and number of addresses are similarly inherited from the address-transaction network, by summation and aggregation.

III. Entity classification

In this section we present the methods used for inferring the category associated with each entity, using our graph neighborhood features on the Bitcoin transaction graph. We first recall the heuristics used for associating distinct addresses to a single entity.

A. *Common spending heuristic for address clustering*

The common spending heuristic consists of clustering addresses that are inputs to the same transaction with the same entity. Formally,

Hypothesis 1 (Common Spending).

$$\forall t \in \mathscr{T}, \quad \exists! e \in \mathscr{E}, \quad \forall i \in I(t), a(i) \in A(e)$$

The common spending heuristic is equivalent to the assumption that having access to multiple private keys at a given point in time is a defining property of an entity. Indeed, in order to submit a transaction T to the Blockchain protocol, the transaction must be signed, implying using the private key of each address in the input set of the transaction, see Figure 5.

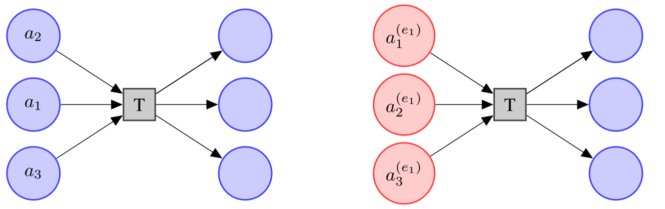

Figure 5. **Common Spending Heuristic**: all addresses input to a transaction are associated with the same entity.

The notion of transitive closure allows extending the set of addresses associated with a given entity.

Hypothesis 2 (Transitive Closure).

$$\exists (t_1, t_2) \in \mathscr{T}^2, s.t. \{a_1, a_2\} = A(I(t_1)), \{a_1, a_3\} = A(I(t_2)) \implies \exists! e, \{a_1, a_2, a_3\} \subset A(e)$$

Under Hypotheses 1, 2, each transaction only has one input entity: $\forall t \in \mathscr{T}_e, |I_e(t)| = 1$. In the following section we present the types of features used for entity classification.

B. *Features*

We wish to demonstrate the effectiveness of graph neighborhood features for entity characterization. As such we propose the following five feature classes:

- Address features involving only address properties,
- Entity features that can be computed from the set of addresses associated with a given entity,
- Temporal features related to the evolution of specific transaction properties,
- Centrality features encoding the value of classical centrality measures [7],
- Motif features corresponding to transaction paths involving the entity of interest.

The remainder of this section provides more details on each set of features.

Address-specific features include attributes such as the total BTC received, the total BTC balance, the number of

Table I
MOTIF DISTRIBUTION: 1-MOTIF DISTINCT IS INDICATIVE OF EXCHANGES (64.6% OF SUCH MOTIFS CORRESPOND TO EXCHANGES). ON THE OTHER HAND 3-MOTIF DIRECT LOOP IS INDICATIVE OF SERVICES (63.0% OF SUCH MOTIFS CORRESPOND TO SERVICES).

Type	Sub-type	Quantity	Exchange	Gambling	Mining	Service	Darknet
1-motif	Loop	16.085.493	26.3%	25.7%	4.9%	39.6%	3.5%
1-motif	Distinct	5.390.310	64.6%	1.1%	2.7%	30.3%	1.2%
2-motif	Direct Loop	10.196.844	21.1%	28.1%	0.1%	48.5%	2.1%
2-motif	Direct Distinct	20.469.285	46.9%	6.3%	3.7%	38.7%	4.4%
3-motif	Direct Loop	30.914.975	24.6%	11.5%	0.1%	63.0%	0.9%
3-motif	Direct Distinct	85.822.858	54.0%	4.4%	3.6%	34.1%	3.9%

input/output transactions, the number of predecessor/successor addresses, unique and otherwise, the number of predecessors addresses that are also successors, and the number of sibling addresses in output.

Analogous features are defined at the entity level as well as the number and proportion of coinbase transactions.

1-motif features include the value of incoming/outgoing transactions in BTC and USD, the number of incoming/outgoing addresses, the number of incoming/outgoing transactions per day, their total value in BTC and USD, and their total fee.

2-motif and 3-motif features are analogous but include also particularities of this graph structure such as, for 2-motif, the number of inputs (resp. outputs) in the first (resp. second) transaction of an incoming/outgoing/loop motif, the total value of the inputs (resp. outputs) of the first (resp. second) transaction of an incoming/outgoing/loop motif in BTC and USD, as well as the number of incoming/outgoing/loop motifs, the number of addresses involved as center of an incoming/outgoing/loop motif, the value transferred in the middle and the fees of the transactions in BTC and USD, and the number of predecessors/successors for an incoming/outgoing motif. 2-motif features are illustrated in Figure 6.

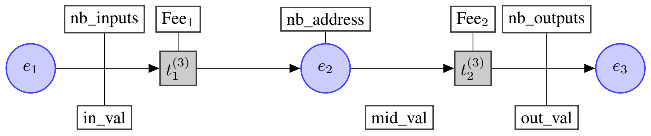

Figure 6. **2-motif features** (rectangular white boxes) annotated over a 2-motif, including both edge features and vertex features over transaction paths.

Centrality-based features include measures of betweenness centrality, closeness centrality, degree centrality, in-degree centrality, out-degree centrality, PageRank, and load centrality. These features are computed on the discrete-time aggregation of the entity graph over a week, a month and a year.

Temporal features are those such as the number of weeks, months, years of activity, the number of entity traded with per week, month, year, the number of receiving, sending days, the activity period duration, and the active day ratio.

We make use of in total 10 address features, 8 entity features, 16 temporal features, 42 centrality features, 44 1-motif features, 81 2-motif features, and 114 3-motif features.

Remark. For each continuous feature such as volume of Bitcoin transactions, we calculate the mean and the standard deviation in order to allow the classifier to discriminate up to second order statistics.

Transaction values are considered both in terms of BTC and USD, in order to support the analysis over the multiple years over which the BTC value in USD changed.

C. Classification method

Given our interest in understanding the behavioral features characterizing certain categories of Bitcoin users, we propose to use a decision tree method, which provides feature relevance statistics via bootstrapping. A decision tree $h_m(x)$ partitions the feature space into J_m disjoint regions $R_{1m}, \dots, R_{J_m m}$ and produces a constant value in each region. The output of $h_m(x)$ for input x can be written as the sum:

$$h_m(x) = \sum_{j=1}^{J_m} b_{jm} \mathbf{1}_{R_{jm}}(x)$$

where b_{jm} is the model response for input features from region R_{jm}. We consider an ensemble of trees, which we learn sequentially by minimizing the weighted sum

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x),$$

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)).$$

according to a gradient boosting procedure, where $L(\cdot)$ is the loss function.

In order to identify the ensemble tree hyper-parameters most suited to our problem, we approximate the performance of the tree ensemble for given parameter values using Gaussian Processes. We learn a surrogate $\hat{f}(\cdot)$ of the loss function $f(\cdot)$ based on previous evaluations $\{(\theta_1, f(\theta_1)), \dots, (\theta_k, f(\theta_k)\}$, and identify the parameter minimizing this surrogate function. The calibration procedure can be summarized as follows:

Algorithm 1. For $t \in \{1, \dots, T\}$:

- Given observations $\{(\theta_i, f(\theta_i)), \forall i \in \{1, \dots, t\}\}$, we build a surrogate $\hat{f}_t$ using Gaussian processes. Each value $f(\theta_i)$ is the loss function value obtained after having trained a decision tree with the hyper-parameter θ_i.
- Given the surrogate function $\hat{f}_t$, we identify the parameter θ_{t+1} providing a good compromise between minimizing the surrogate $\hat{f}_t$ and exploring the parameter space. The evaluation of the surrogate function for a parameter

value consists in running the gradient boosting routine described above.

We benchmark the result of our decision tree algorithm against a logistic regression algorithm, defined below:

$$\forall x_i \in \mathscr{E}, \quad \hat{f}_{entity}(x_i) = y_i = \underset{c \in \mathscr{C}}{\operatorname{argmax}} f_c(x_i)$$

where

$$f_c(X) = \mathbb{P}(Y = c|X) = h((X^{\Phi})^T \beta_c), \quad h(t) = \frac{e^t}{1 + e^t}.$$

IV. Numerical Results

In this section we present our experimental results, as well as implications of these results for Bitcoin transaction anonymity.

A. Blockchain dataset

We consider the set of blocks of height inferior or equal to 514.971, corresponding to blocks created before March 24th 2018, 15:19:02, which contains about 500.000.000 addresses. Address labels are obtained from WalletExplorer[1].

We apply the common spending heuristic and transitive closure operation described in Section III-A to the labeled dataset obtained from WalletExplorer, and extend it slightly. We interact with the Bockchain via the BlockSci toolbox v.0.4.5 released on March 16th 2018 [11], on a 64 GB machine. The final labeled dataset used in numerical experiments consists of 30.331.700 addresses, associated with $|\mathscr{E}_{known}| = 272$ entities representing 5 entity categories in the following proportions:

- *Exchange*: 108 entities, 7.892.587 addresses.
- *Service*: 68 entities, 17.606.608 addresses.
- *Gambling*: 65 entities, 2.775.810 addresses.
- *Mining Pool*: 19 entities, 78.488 addresses.
- *DarkNet Marketplace*: 12 entities, 1.978.207 addresses.

While the set of labeled address is of significant size, it is important to observe the entity category class imbalance, with the dominant *Service* class representing more than 50% of the dataset, and the smallest *Mining Pool* category representing less than 0.5% of the dataset.

B. Model calibration procedure

The decision tree model described in Section III-C is deployed in Python via the LightGBM implementation [12]. The Gaussian Process-based optimization procedure for hyper-parameter optimization is implemented using the Python skopt library[2] with initial parameter values obtained from a coarse random search.

We use a typical 70/30 training/test partition of our dataset. The learning rate hyper-parameter of the decision tree model is optimized over the interval $[0.01, 0.5]$ after having done a random search over $[0, 2]$; the resulting value is 0.18. To train LightGBM, we use an early stopping procedure which stops the training if the log loss does not decrease over ten consecutive iterations. The procedure stops after 61 iterations with a loss of 0.35.

The inverse of the L^2 regularization parameter of the logistic regression model is optimized over the interval $[0.01, 3]$ after having done a random search over $[0, 5]$; the value obtained is 0.55.

The Gaussian Processes procedure is used with 50 iterations, which is a reasonable compromise between a small computation time, less than one hour for LightGBM in our hardware setting, and a good exploration of the interval. For fairness we use the same criterion for the logistic regression model. Along the same lines, we only optimize one hyper-parameter for LightGBM, namely the learning rate.

C. Feature importance

We analyze the performance of the classification model first from the perspective of an unsophisticated attacker incrementally adding features to a generic model based on their ease of access. We then model a sophisticated attacker with extensive modeling knowledge, collecting the full set of features, calibrating model hyper-parameters, and identifying the minimal set of relevant features required for a successful attack.

1) Weak attacker: Consider an attacker who collects features in the following order, from simplest to most complex:

- Address features requiring access only to the address set,
- Entity features requiring access to the address set and address clustering heuristics,
- Temporal features,
- Centrality features requiring crawling the Blockchain transaction network for connectivity information,
- 1,2,3-motif features: requiring crawling the Blockchain transaction network for both connectivity and transaction information (amount, fee, etc.).

We consider both the advanced decision tree model as well as the logistic regression model for classification, representing the range of sophisticated and simple methods. To model the weak attacker, we do not invoke the hyper-parameter calibration procedure, and use the default parameter settings. F1 and accuracy scores are reported in Table II. Given that 1-motifs are simpler to compute in practice for an attacker, we place them before Temporal and Centrality features when reporting the results.

The results provide two relevant insights. First it is notable that even a weak attacker using a simple model (logistic regression) with default hyper-parameter settings, and using only the 10 simplest features, is able to identify the entity type with 41% accuracy, i.e. a factor of two improvement as compared to a random guess over the 5 classes.

Second, it is of significance that while the model choice does not significantly impact the classification performance, using more sophisticated features provides drastic improvement. Indeed, the user of 3-motif features, encompassing the behavior of the 3-hop graph neighborhood of a given identity, contributes more than 30% relative improvement to

[1]https://www.walletexplorer.com/
[2]https://scikit-optimize.github.io/

Table II
INCREMENTAL GROUPING OF FEATURES AND ASSOCIATED PERFORMANCE METRICS.

Features	—Features—	Alg.	Accuracy	F_1	Precision
Address	10	LR	0.415	0.303	0.351
Entity	18 (+8)	LR	0.476	0.369	0.445
1-motif	62 (+44)	LR	0.524	0.471	0.474
Temporal	78 (+16)	LR	0.512	0.493	0.498
Centrality	120 (+42)	LR	0.561	0.545	0.551
2-motif	201 (+81)	LR	0.585	0.574	0.573
3-motif	315 (+114)	LR	0.841	0.835	0.857
Address	10	LGBM	0.5	0.487	0.492
Entity	18 (+8)	LGBM	0.476	0.429	0.415
1-motif	62 (+44)	LGBM	0.622	0.597	0.613
Temporal	78 (+16)	LGBM	0.659	0.649	0.654
Centrality	120 (+42)	LGBM	0.610	0.597	0.603
2-motif	201 (+81)	LGBM	0.683	0.654	0.667
3-motif	315 (+114)	LGBM	0.890	0.886	0.897

the accuracy score (from 0.68 to 0.89 accuracy for the decision tree model).

2) Strong attacker: We now consider a strong attacker collecting the full set of features, and then selecting a small set of highly significant features. We model this process by applying the hyper-parameter calibration procedure described earlier. We then obtain the ranked feature set from the decision tree model. Table III provides the top ten features of the decision tree model.

Table III
RANKED FEATURES: BY IMPORTANCE ACCORDING TO THE DECISION TREE MODEL.

Rank	Type	Name
1	3-motif	unique_entity_3_successor
2	Entity	prop_coinbase
3	2-motif	loop_2_std_nb_inputs
4	2-motif	loop_2_mean_nb_inputs
5	3-motif	outgoing_3_mean_nb_outputs
6	3-motif	outgoing_3_std_nb_outputs
7	3-motif	outgoing_3_std_fee_1_btc
8	3-motif	outgoing_3_mean_nb_inputs
9	1-motif	incoming_mean_fee_usd
10	3-motif	loop_3_mean_fee_2_btc

The results indicate that the 3-hop graph neighborhood features dominate. Note, however that there are 114 such 3-hop graph neighborhood features out of 315 features in total.

It can be seen from Table III that outgoing features are more relevant that incoming features. This supports a "causal" interpretation that features specific to an entity can be well detected from its downstream transaction trace.

We now examine the minimal set of features required to obtain state-of-the-art accuracy. We present in Figure 7 the F1 and accuracy improvement over the complete feature relevance ranking from the decision tree algorithm for both the logistic regression and the decision tree model.

It is clear from the figure that even with a simple logistic regression model, the 20 most relevant features are sufficient to obtain high classification accuracy. Using a more sophisticated

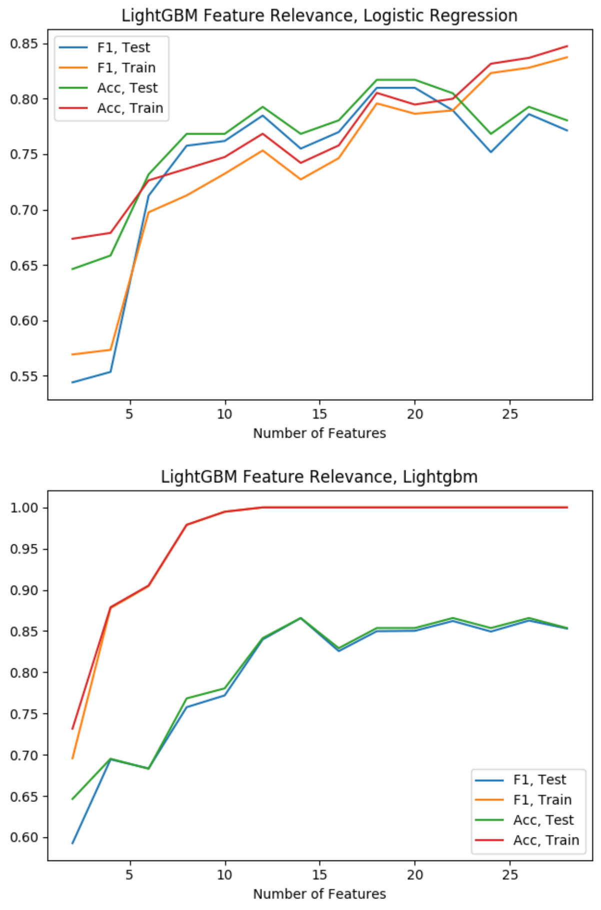

Figure 7. **Feature Selection** using LightGBM

decision tree model allows reducing the number of features to 15, after which improvement plateaus.

D. Overall model performance

Finally, we consider the sophisticated attacker using hyper-parameter optimization. Table IV presents the Accuracy and F1 score for both classification methods.

Table IV
OPTIMIZING F_1 WITH 315 FEATURES, 50 ITERATIONS OF THE HYPER-PARAMETER OPTIMIZATION PROCEDURE, GLOBAL RESULTS.

Algorithm	Accuracy	F_1	Precision
Logistic Regression	0.85	0.85	0.87
LightGBM	0.92	0.91	0.92

Table V, provides the class-specific overall performance results.

The results dominate state-of-the-art results from the literature, which may be due to the use of novel advanced graph neighborhood features, as evidenced from Table II, along with the hyper-parameter optimization. At the class level, good classification accuracy (above 95%) is achieved over the set of exchanges, gambling services and general services as well

Table V
OPTIMIZING F_1 WITH 315 FEATURES, 50 ITERATIONS OF THE HYPER-PARAMETER OPTIMIZATION PROCEDURE, CLASS-LEVEL RESULTS.

Category	Algorithm	Accuracy	F_1	Precision
Exchange	LR	0.91	0.91	0.91
Gambling	LR	0.9	0.82	0.75
Mining	LR	0.5	0.67	1.0
Service	LR	0.85	0.87	0.89
Darknet	LR	0.75	0.75	0.75
Exchange	LGBM	0.94	0.92	0.91
Gambling	LGBM	0.95	0.97	1.0
Mining	LGBM	0.5	0.67	1.0
Service	LGBM	0.95	0.88	0.83
Darknet	LGBM	1.0	1.0	1.0

as the darknet category. Mining pool behavior is less-well captured as evidenced by the low accuracy, indicating that there is no consistent transaction pattern identified for this class.

V. CONCLUSION

We formulate the problem of analyzing Bitcoin Blockchain transaction graph anonymity properties as a classification problem over a set of categories of Bitcoin users. Our results indicate that it is feasible for a weak attacker to characterize entities using a set of new graph neighborhood features that we propose, and that is feasible for a strong attacker to do the same with as little as 15 of the most relevant features.

This suggests a number of interesting avenues for further work. Since it is possible as we have shown to accurately classify transacting entities on the Bitcoin Blockchain, the question arises as to whether it is possible to develop an effective generative model of the transaction network. If so, it would enable a wealth of studies into the effect of changes in network protocols or regulatory frameworks on the evolution of the Bitcoin economy.

REFERENCES

[1] Cuneyt Gurcan Akcora, Yulia R. Gel, and Murat Kantarcioglu. Blockchain: A graph primer. *CoRR*, abs/1708.08749, 2017.

[2] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, Sharon Weed Cocco, and Jason Yellick. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *EuroSys*, pages 1–15. ACM, 2018.

[3] Christian Cachin, Angelo De Caro, Pedro Moreno-Sanchez, Björn Tackmann, and Marko Vukolic. The transaction graph for modeling blockchain semantics. *IACR Cryptology ePrint Archive*, 2017:1070, 2017.

[4] Dmitry Ermilov, Maxim Panov, and Yury Yanovich. Automatic bitcoin address clustering. In *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*, pages 461–466. IEEE, 2017.

[5] Giulia Fanti and Pramod Viswanath. Deanonymization in the bitcoin P2P network. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1364–1373, 2017.

[6] Y.J. Fanusie and T. Robinson. Bitcoin laundering: An analysis of illicit flows into digital currency services. *Center on Sanctions and Illicit Finance, Elliptic*, 2018.

[7] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[8] Martin Harrigan and Christoph Fretter. The unreasonable effectiveness of address clustering. *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pages 368–373, 2016.

[9] Danny Yuxing Huang, Maxwell Matthaios Aliapoulios, Vector Guo Li, Luca Invernizzi, Elie Bursztein, Kylie McRoberts, Jonathan Levin, Kirill Levchenko, Alex C Snoeren, and Damon McCoy. Tracking ransomware end-to-end. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 618–631. IEEE, 2018.

[10] Husam Al Jawaheri, Mashael Al Sabah, Yazan Boshmaf, and Aiman Erbad. When A small leak sinks A great ship: Deanonymizing tor hidden service users through bitcoin transactions analysis. *CoRR*, abs/1801.07501, 2018.

[11] Harry A. Kalodner, Steven Goldfeder, Alishah Chator, Malte Möser, and Arvind Narayanan. Blocksci: Design and applications of a blockchain analysis platform. *CoRR*, abs/1709.02489, 2017.

[12] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3146–3154, 2017.

[13] M. Lischke and B. Fabian. Analyzing the bitcoin network: The first four years. *Future Internet*, 8(1):7, 2016.

[14] D. D. F. Maesa, A. Marino, and L. Ricci. Uncovering the bitcoin blockchain: An analysis of the full users graph. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 537–546, October 2016.

[15] D. McGinn, D. McIlwraith, and Y. Guo. Toward open data blockchain analytics: A bitcoin perspective. *CoRR*, abs/1802.07523, 2018.

[16] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140. ACM, 2013.

[17] Malte Möser and Rainer Böhme. The price of anonymity: empirical evidence from a market for bitcoin anonymization. *J. Cybersecurity*, 3(2):127–135, 2017.

[18] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[19] Arvind Narayanan, Elaine Shi, and Benjamin IP Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1825–1834. IEEE, 2011.

[20] Jonas David Nick. Data-driven de-anonymization in bitcoin. Master's thesis, ETH-Zürich, 2015.

[21] Masarah Paquet-Clouston, Bernhard Haslhofer, and Benoit Dupont. Ransomware payments in the bitcoin ecosystem. *CoRR*, abs/1804.04080, 2018.

[22] Stephen Ranshous, Cliff A Joslyn, Sean Kreyling, Kathleen Nowak, Nagiza F Samatova, Curtis L West, and Samuel Winters. Exchange pattern mining in the bitcoin transaction directed hypergraph. In *International Conference on Financial Cryptography and Data Security*, pages 248–263. Springer, 2017.

[23] Fergal Reid and Martin Harrigan. An analysis of anonymity in the bitcoin system. In *SocialCom/PASSAT*, pages 1318–1326. IEEE, 2011.

[24] Dorit Ron and Adi Shamir. Quantitative analysis of the full bitcoin transaction graph. In Ahmad-Reza Sadeghi, editor, *Financial Cryptography and Data Security*, pages 6–24, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[25] Gabriel Tanase, Toyotaro Suzumura, Jinho Lee, Chun-Fu Chen, Jason Crawford, Hiroki Kanezashi, Song Zhang, and Warut D. Vijitbenjaronk. System G distributed graph database. *CoRR*, abs/1802.03057, 2018.

[26] Kentaroh Toyoda, Tomoaki Othsuki, and P. Takis Mathiopoulos. Multi class bitcoin-enabled service identification based on transaction history summarization. In *IEEE Conference on IoT, GCC, CPSC, SD, B, CIT, Congress on Cybermatics*, 2018.

[27] Saurabh Verma and Zhi-Li Zhang. Hunt for the unique, stable, sparse and fast feature learning on graphs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 88–98, 2017.