Demystifying Fraudulent Transactions and Illicit Nodes in the Bitcoin Network for Financial Forensics

Youssef Elmougy and Ling Liu
School of Computer Science, Georgia Institute of Technology
Atlanta, GA 30332, USA

ABSTRACT

Blockchain provides the unique and accountable channel for financial forensics by mining its open and immutable transaction data. A recent surge has been witnessed by training machine learning models with cryptocurrency transaction data for anomaly detection, such as money laundering and other fraudulent activities. This paper presents a holistic applied data science approach to fraud detection in the Bitcoin network with two original contributions. First, we contribute the *Elliptic++ dataset*, which extends the Elliptic transaction dataset to include over 822*k* Bitcoin wallet addresses (nodes), each with 56 features, and 1.27*M* temporal interactions. This enables both the detection of fraudulent transactions and the detection of illicit addresses (actors) in the Bitcoin network by leveraging four types of graph data: (i) the transaction-to-transaction graph, representing the money flow in the Bitcoin network, (ii) the address-to-address interaction graph, capturing the types of transaction flows between Bitcoin addresses, (iii) the address-transaction graph, representing the bi-directional money flow between addresses and transactions (BTC flow from input address to one or more transactions and BTC flow from a transaction to one or more output addresses), and (iv) the user entity graph, capturing clusters of Bitcoin addresses representing unique Bitcoin users. Second, we perform fraud detection tasks on all four graphs by using diverse machine learning algorithms. We show that adding enhanced features from the address-to-address and the address-transaction graphs not only assists in effectively detecting both illicit transactions and illicit addresses, but also assists in gaining in-depth understanding of the root cause of money laundering vulnerabilities in cryptocurrency transactions and the strategies for fraud detection and prevention. The Elliptic++ dataset is released at https://www.github.com/git-disl/EllipticPlusPlus.

KEYWORDS

Blockchain, Anomaly Detection, Financial Forensics

1 INTRODUCTION

The emergence of cryptocurrency, initiated by Bitcoin [28], has kindled an adoption of a new exchange system in which financial transactions can be completed without an intermediary. This popularity has been perceived as a catalyst for the integration of cryptocurrencies into the payment systems of traditional financial institutions, enabling processing of exchanges between fiat money and cryptocurrency. To allow for immense volumes of such exchanges, financial institutions must provide rigid security against high risk transactions from fraudulent activities and assure compliance with anti-money laundering (AML) regulations. Although machine learning (ML) models have become a popular tool for anomaly detection and risk assessment of cryptocurrency transactions in recent years, we argue that fraud detection models trained for financial forensics should revolve around two goals: (i) prioritizing higher recall with a (reasonable) trade-off on precision, since the penalty of classifying an illicit transaction/account as licit far outweighs that of the reverse, and (ii) allowing explainability (and derivation of proof) on the risk of transactions/accounts.

Bitcoin [28] is the most widely used cryptocurrency. Analyzing its blockchain will provide a basis of reference. The Elliptic dataset [36] is the largest labelled Bitcoin transaction data publicly available. The dataset consists of over 203*k* transactions labelled illicit, licit, and unknown. It provides a good entry into fraudulent transaction analysis. However, the Elliptic dataset [36] consists of only Bitcoin transactions, without features of the addresses involved and the different interactions between pairs of addresses. Hence, the forensic analysis using the Elliptic dataset suffers from a prominent downfall: when a model predicts an illicit transaction, the addresses responsible cannot be clearly identified since a transaction may be associated with several input and output addresses.

The first contribution of this paper is the *Elliptic++ dataset*, which extends the Elliptic dataset to include all the Bitcoin wallet addresses (actors) and and their temporal interactions associated to the transactions in the Elliptic dataset. Hence, forensics performed using the Elliptic++ dataset can identify fraudulent transactions and dishonest actors in the Bitcoin network and will also allow explainability as to why a wallet address associated with a transaction is illusive and dishonest. The construction of the *Elliptic++ dataset* contributes a novel way in collecting and visualizing Blockchain data, using wallet addresses as the center of a risk detection model. We collect Blockchain data associated with the Elliptic dataset (transactions). We first augment each transaction in the *transactions dataset* with 17 additional features, then we crawl the Blockchain to create the Elliptic++ dataset, consisting of the *feature-enhanced transactions dataset* and the *actors (wallet addresses) dataset*. The actors dataset includes over 822*k* labelled wallet addresses, each described with 56 features, and over 1.27*M* temporal occurrences (interactions) across the same time steps (as those recorded in the Elliptic dataset). With our Elliptic++ dataset, one can perform anomaly detection of fraudulent activities, such as illicit transactions and fraudulent actors. We also include four types of graphs: the *Money Flow Transaction Graph*, the *Actor Interaction Graph*, the *Address-Transaction Graph*, and the *User Entity Graph*. These graphs contribute to both mining and visualization of the connections of transactions and the interactions among wallet addresses through their transactions.

The second contribution of the paper is to perform fraud detection using the Elliptic++ dataset and the four graph representations by combining diverse ML algorithms and feature optimizations. We observe that Random Forest (RF) with feature refinement offers the best-performing model: on the transactions dataset, it achieves 98.6% precision and 72.7% recall compared to 97.5% precision and

Table 1: Data structure for an example transaction node in the transactions dataset: (1) *features*, row of all 184 feature values for `txId`, (2) *edgelist*, all incoming and outgoing edges involving `txId`, and (3) *classes*, the class label for `txId`.

`txs_features.csv`

txId	Time step	LF_1	···	LF_93	AF_1	···	AF_72	TXS_in	···	BTC_out_total
272145560	24	-0.155493	···	1.135279	-0.159681	···	1.521399	1	···	2.77279994

`txs_edgelist.csv`

txId1	txId2
272145560	296926618
272145560	272145556
299475624	**272145560**

`txs_classes.csv`

txId	class
272145560	1

71.9% recall when using RF without feature refinement; on the actors dataset, RF with feature refinement achieves 92.1% precision and 80.2% recall, compared to 91.1% precision and 78.9% recall when using RF without feature refinement. Furthermore, the fraud detection using the Elliptic++ dataset allows for in-depth understanding of the root cause of fraudulent activities in cryptocurrency transactions through semantic and statistical explainability and shining light on the strategies for fraud detection and prevention.

2 RELATED WORK

Since the introduction of the Elliptic dataset in 2019 by Weber et al. [36], there have been numerous efforts on data labelling and anomaly detection using ML models. Lorenz et al. [24] assumed minimal access to labels, and proposes an active learning solution by leveraging a smaller subset of the available labels. Oliveira et al. [29] proposed GuiltyWalker, a detection method that computes new features based on the structure of a transaction-to-transction graph and the distance to known illicit transactions. Alarab et al. [3] presented a comparative analysis of the Elliptic dataset using different supervised learning methods. Loa et al. [23] proposed Inspection-L, a graph neural network framework for anomaly detection, and similarly, Alarab et al. [2] proposed a graph-based LSTM with a graph convolutional network to detect illicit transactions.

More generally, there is increasing interest in AML in the context of financial transactions and cryptocurrency networks, such as Bitcoin or Ethereum. The lack of labelled data and the imbalanced data are two major challenges for fraud detection when using supervised learning [6, 16, 27] or unsupervised learning [13, 26, 27, 30, 31]. Some efforts have also focused on de-anonymizing mixed transactions by Bitcoin-dedicated graph-based approaches [7, 8, 18, 37]. Regarding Bitcoin account profiling, Michalski et al. [32] built a small dataset of 9,000 addresses and applied supervised learning to characterize nodes in the Blockchain as miner or exchange, indicating the need of countermeasures for preserving the desired level of anonymity. To the best of our knowledge, our Elliptic++ dataset is the largest public Bitcoin account dataset with 822,942 addresses.

Existing works of Ethereum account profiling exclusively focus on graph representation learning methods. For Ethereum phishing account detection, [38] propose Trans2Vec, which conducts biased random walks by taking into consideration transaction time and amount information. Several approaches [21, 22] follow similar design to Trans2Vec. Deep graph neural networks for representation learning have also been used to profile Ethereum accounts. [34] applied Graph Convolution Network (GCN) [17] to infer the identity of Ethereum accounts. Similarly, [42] proposed a hierarchical graph attention encoder (HGATE), which extracts and combines features from the node- and subgraph-level. [20] proposed TTAGNN, which fuses multiple temporal edges by an LSTM encoder and adopts Graph Attention Network (GAT) [35] to generate node embedding. [12] propose a cascade method consisting of statistical feature extraction and a lightGBM-based ensemble algorithm. Due to the difference in fundamental settings (UTXO and account models), the graph representation learning models developed for profiling Ethereum accounts are difficult to extend to the Bitcoin network.

3 THE ELLIPTIC++ DATASET

The Elliptic++ dataset consists of 203k Bitcoin transactions and 822k wallet addresses. It leverages elements from the Elliptic dataset[1] [36], a published dataset deanonymizing 99.5% of Elliptic transaction data[2], and the Bitcoin addresses dataset obtained by using our Bitcoin Blockchain[3] scraping pipeline. A detailed description of the data collection pipeline is included in the Appendix.

3.1 Transactions Dataset

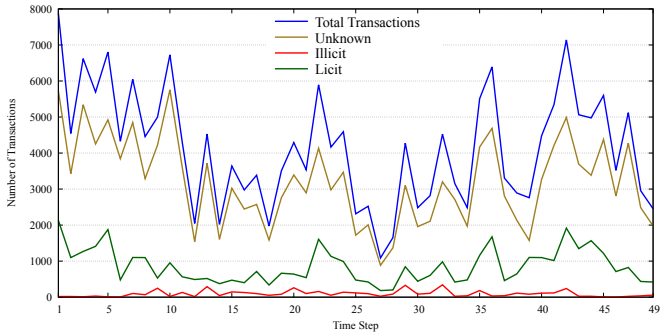
The *transactions dataset* consists of a time-series graph with 49 distinct time steps, 203,769 transactions (nodes), and 234,355 directed edges representing the payment flows. Each transaction node is labelled as licit, illicit, or unknown; with 2% (4,545) labelled as class-1 (illicit), 21% (42,019) as class-2 (licit), and the remaining transactions are unknown with regard to licit/illicit, hence labelled as class-3. Three `csv` files are used, as shown in Table 1. Each transaction node has an entry in `txs_features.csv`, with numerical data for 183 transaction features, and an entry in `txs_classes.csv`, representing its class label (1: *illicit*, 2: *licit*, 3: *unknown*). Each edge has an entry in `txs_edgelist.csv` (indexed by two transaction IDs), representing money flow from one transaction to another. Among the 183 node features, 166 features are inherited from the Elliptic dataset, i.e., the time step, 94 *local features*, representing local information about the transaction, and 72 *aggregate features*, obtained by aggregating transaction information one-hop forward/backward. The remaining 17 node features are gathered by our Elliptic++ data collection pipeline (with the exception of the 0.5% of transactions that were not deanonymized) as *augmented features* and are shown in Table 2. Figure 1 shows the distribution of transactions across the three classes in each of the 49 time steps.

[1] www.kaggle.com/datasets/ellipticco/elliptic-data-set
[2] www.kaggle.com/datasets/alexbenzik/deanonymized-995-pct-of-elliptic-transactions
[3] www.blockchain.com

Table 2: Transactions dataset augmented features.

Feature	Description
BTC_{in}	Total BTC incoming
BTC_{out}	Total BTC outgoing
each has 5 values:	**total, min, max, mean, median**
Txs_{in}	Number of incoming transactions
Txs_{out}	Number of outgoing transactions
$Addr_{in}$	Number of input addresses
$Addr_{out}$	Number of output addresses
BTC_{total}	Total BTC transacted
$Fees$	Total fees in BTC
$Size$	Total transaction size
	single value

Figure 1: Number of transactions by time step.



3.2 Actors (Wallet Addresses) Dataset

The *actors (wallet addresses) dataset* is a graph network of 822, 942 wallet addresses, each with 56 features as shown in Table 3. Five csv files are used, as shown in Table 4. Each address has an entry in `wallets_features.csv`, with numerical data for the time step and 56 address features listed in Table 3 for each transaction it was involved in, and an entry in `wallets_classes.csv`, representing its class label (1: *illicit*, 2: *licit*, 3: *unknown*). A wallet address is labelled *illicit* if it has at least 1 edge with an illicit transaction, otherwise it is labelled *licit* if the ratio of its total unknown transactions (edges) to its total licit transactions (edges) is > 3.7 or *unknown* if ≤ 3.7. The ratio "3.7" is calculated using the total ratio of unknown transactions to licit transactions in the transactions dataset. We also create `AddrAddr_edgelist.csv` to record the pairwise interactions of input and output addresses through Bitcoin transactions. Each entry represents the input address and output address relationship of one transaction. If there are multiple transactions between a pair of addresses, then there are multiple entries in this table. Additionally, we create `AddrTx_edgelist.csv`, where each entry represents the relationship between an input address and a transaction, and `TxAddr_edgelist.csv`, with each entry representing a directed connection between a transaction and an output address. With these data structures, the Elliptic++ dataset can be used to build the address-to-address graph (Section 3.3.2), and the address-transaction graph (Section 3.3.3), in addition to the transaction-to-transaction money flow graph.

Table 3: Actors (wallet addresses) dataset features.

Feature	Description
Transaction related:	
$BTC_{transacted}$	Total BTC transacted (sent+received)
BTC_{sent}	Total BTC sent
$BTC_{received}$	Total BTC received
$Fees$	Total fees in BTC
$Fees_{share}$	Total fees as share of BTC transacted
Time related:	
$Blocks_{txs}$	Number of blocks between transactions
$Blocks_{input}$	Number of blocks between being an input address
$Blocks_{output}$	Number of blocks between being an output address
$Addr\ interactions$	Number of interactions among addresses
5 values:	**total, min, max, mean, median**
$Class$	class label: {illicit, licit, unknown}
Transaction related:	
Txs_{total}	Total number of blockchain transactions
Txs_{input}	Total number of dataset transactions as input address
Txs_{output}	Total number of dataset transactions as output address
Time related:	
$Timesteps$	Number of time steps transacting in
$Lifetime$	Lifetime in blocks
$Block_{first}$	Block height first transacted in
$Block_{last}$	Block height last transacted in
$Block_{first\ sent}$	Block height first sent in
$Block_{first\ receive}$	Block height first received in
$Repeat\ interactions$	Number of addresses transacted with multiple times
	single value

The distribution of address classes are 2% (14, 266) class-1 (illicit), 31% (251, 088) class-2 (licit), and the remaining are unknown (class-3). When populated using temporal information provided by the time steps in the transactions dataset, we obtain 1, 268, 260 wallet address occurrences across all time steps, including 2% (28, 601) illicit addresses, 27% (338, 871) licit addresses, and 71% (900, 788) unknown addresses, as shown in Figure 2.

3.3 Graph Visualization

3.3.1 Money Flow Transaction Graph. This is a directed graph where nodes represent transactions and edges represent directed BTC flows from one transaction to the next. Figure 3 visualizes the distribution of all transactions in time step 32. We choose three transactions (one per class) representing the unknown (yellow), illicit (blue), and licit (red) classes, and display sub-graphs of their

Table 4: Data structure for an example address in the actors dataset: (1) *features*, row of 56 feature values for `address`, (2) *classes*, the class label for `address`, (3) *AddrAddr*, all edges involving `address` in the actor interaction graph, (4) *AddrTx*, all edges as involving `address` as the input address in the address-transaction graph, and similarly (5) *TxAddr*, as the output address.

`wallets_features.csv`

address	time step	txs_input	···	lifetime_blocks	···	Addr_interactions_median
39sfuA8pY4UfybgEZi7uvA13jkGzZpsg5K	23	420	···	18145	···	1

`AddrAddr_edgelist.csv`

input_address	output_address
39sfuA8pY4UfybgEZi7uvA13jkGzZpsg5K	1ML...kTL

`wallets_classes.csv`

address	class
39sf...sg5K	1

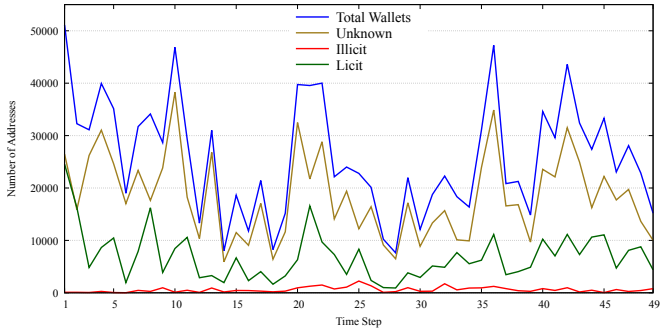
`AddrTx_edgelist.csv`

input_address	txId
39sfuA8pY4UfybgEZi7uvA13jkGzZpsg5K	272145560

`TxAddr_edgelist.csv`

txId	output_address
322554634	**39sfuA8pY4UfybgEZi7uvA13jkGzZpsg5K**

Figure 2: Number of wallet addresses by time step.

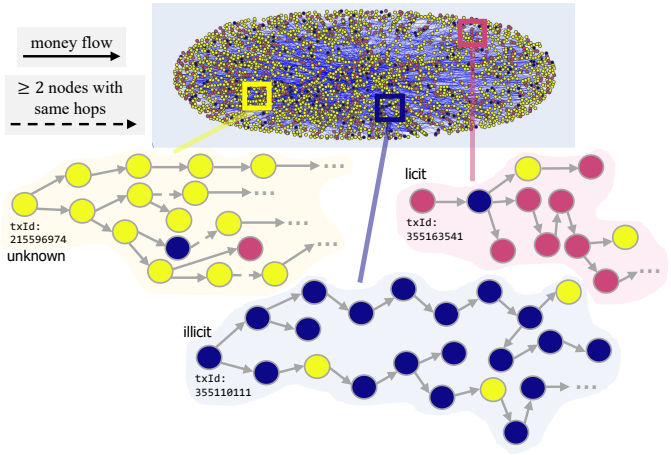


four-hop neighbourhoods as shown in the left, middle, and right of Figure 3. All neighbour nodes are shown, though for visual clarity, some edges are abbreviated with dotted arrows signifying that a particular node has two (or more) outgoing edges of the same pattern as the graph is traversed further. This displays the potential utility of exploring the spatial and temporal patterns surrounding a given transaction.

It is important to note that some time steps produce sparse transaction graphs, e.g. time step 14, while others produce dense transaction graphs, e.g. time step 32. We provide a comparison in the Appendix (see Figure 14). This difference in graph density drastically affects the node neighborhood patterns, with some time steps creating shallow, wide money flow transaction graphs, while other time steps creating deep, narrow money flow transaction graphs. Moreover, although there are only 2% illicit transactions within the dataset, they are evidently spread out across each time step and hence the ML model trained for anomaly detection over the earlier sequence of time steps can be leveraged to perform forensics on the later part of the sequence of time steps.

3.3.2 Actor Interaction Graph. This is a directed graph where nodes represent addresses and edges represent pairwise interactions among input and output addresses of transactions. Figure 4 visualizes the distribution of all addresses in time step 32. Unlike Figure 3 where transaction interactions are captured in terms of money flows, Figure 4 displays the interactions at the address level as opposed to transaction level. The grey area is due to the density of edges. Here, the density of the address graphs at a given time step impact the density of the k-hop neighborhood of a wallet address

Figure 3: Money flow transaction graphs for selected unknown (left), illicit (middle), and licit (right) transactions in time step 32 (top shows distribution of TS 32 transactions).



node. We provide a comparison of actor interaction graphs at time steps 14 and 32 in the Appendix (see Figure 15).

3.3.3 Address-Transaction Graph. This is a heterogeneous directed graph supported by our Elliptic++ dataset. It has two types of nodes, the transaction node (circle) and the address node (square), and two types of edges, the sender-to-transaction edges, each of which represents the BTC flow from an input address (sender) to a transaction, and the transaction-to-receiver edges, each of which represents the BTC flow from a transaction to an output address. Figure 5 shows an address-transaction graph of selected transactions and addresses anchored at a given actor (i.e., the Illicit Actor 13) in time step 32. The flow and quantity of input and output addresses provide

Figure 4: Distribution of wallet addresses in time step 32.

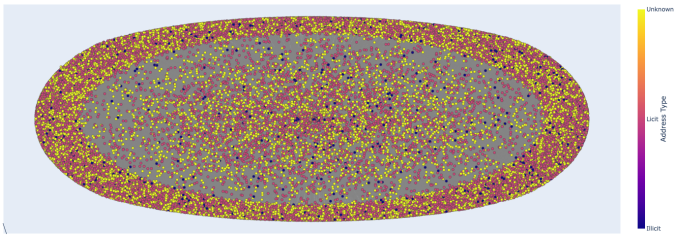
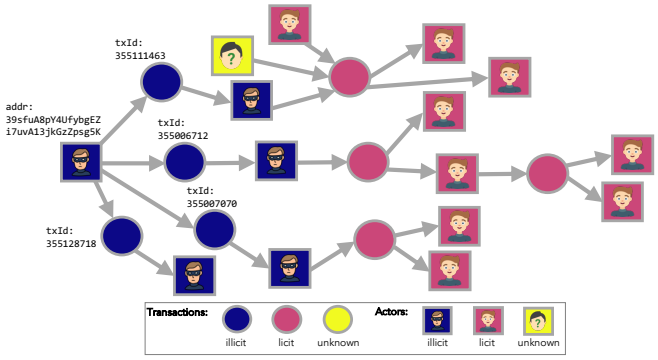


Figure 5: An address-transaction graph for selected nodes in time step 32 featuring Illicit Actor 13 at the root.



information regarding the purposes of a transaction and the relationships among addresses connected by the same transaction. We provide a visual comparison of the distributions of all transactions and addresses at both time steps in the Appendix (see Figure 16).

3.3.4 User Entity Graph. This graph is generated by address clustering analysis rather than direct construction using the Elliptic++ dataset. Clustering addresses involves taking the set of all addresses $\mathcal{A}$ as training data to build a clustering model which creates a set of disjoint subsets $\mathcal{U} = \{U_1, U_2, ..., U_n\}$ (where $U_1, U_2, ..., U_n$ represents n clusters of addresses in $\mathcal{A}$) such that $\bigcup_{i=1}^{n} C_i = \mathcal{A}$. By treating each cluster (sub-graph) as one user, we construct a user-entity graph of n nodes, where each node represents a user in Bitcoin network and the edges represent interactions between unique users through Blockchain transactions. Address clustering is performed in four steps. First, for each transaction, the set of input addresses associated with the transaction is collected. Second, transactions whose address sets are overlapping with one or more addresses are grouped into a unique user. Third, the address-transaction graph is searched to highlight all transactions corresponding to each user. Finally, the highlighted address-transaction graph is converted into a user entity graph. Figure 6 shows the resulting user graph from the same subset of addresses and transactions in Figure 5. Refer to Section A.4 in the Appendix (Figure 18) for a detailed explanation and visualization of the clustering process. Bitcoin address clustering is a hot research topic [4, 25, 33, 41] due to its potential of linking addresses controlled by a specific user, effectively deanonymizing the identity of those users.

4 FRAUD DETECTION METHODOLOGY

4.1 Dataset Preprocessing

We used an effective 70/30 train-test split with respect to time steps for both the transactions and actors datasets, with time steps 1 to 34 for training and time steps 35 to 49 for testing. Figure 7 graphically shows the distribution of data points (top for transactions, bottom for actors) of all three classes by time step for both the training and testing sets. We provide a detailed distribution of the number of transactions and addresses for all three classes in each of the 49 steps in the Appendix (Tables 12 and 13 respectively). Due to the underlying class imbalance across illicit and licit classes, normalization and standardization transformations are applied. The

Figure 6: A user entity graph created by grouping transactions with ≥ 1 common element in each set into unique users and adding edges across users if a path existed between groups of transactions in the original graph.

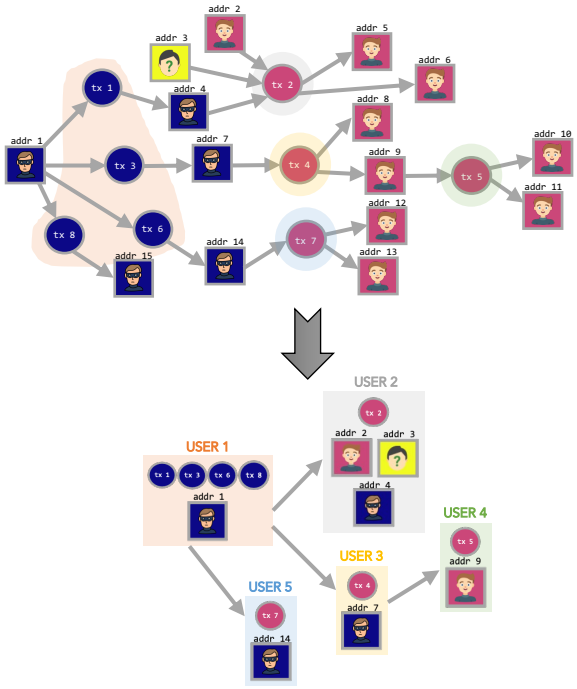
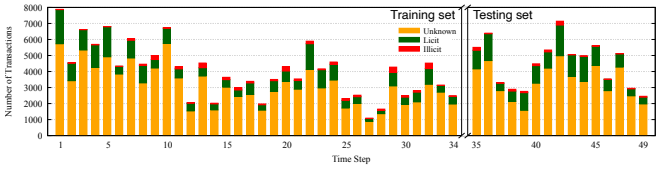
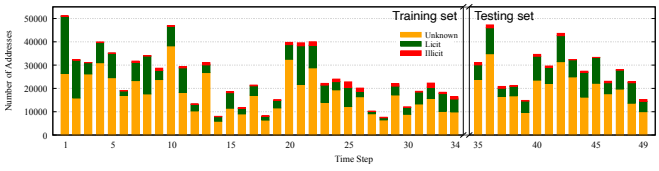


Figure 7: The distribution of the number of transactions and addresses by time step for training and testing sets.



(a) Transactions dataset.



(b) Actors dataset.

augmented features in the transactions dataset and all features in the actors dataset are transformed by scaling each feature using the MinMaxScaler to the range $(0, 1)$, reducing imbalance and assisting with model convergence [19].

4.2 Machine Learning Models

Previous studies [29, 36] indicate that ensemble methods perform better than graph neural networks, hence the ML models used for evaluation included Random Forest (RF) [10], Multilayer Perceptrons (MLP) [9], Long Short-Term Memory (LSTM) [14], and Extreme Gradient Boosting (XGB) [11]. We also include Logistic Regression (LR) [15] as the baseline. LR is a single layer neural

Table 5: The distribution of the number of illicit actors (that appear in 1, 2 − 4, and ≥ 5 time steps) by time step.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
illicit actor appears in 1 time step	68	69	42	186	36	8	233	116	384	26	185	47	342	74	193	174	118
illicit actor appears in 2 − 4 time steps	5	1	3	15	2	1	28	7	21	11	7	2	4	0	6	8	8
illicit actor appears in ≥ 5 time steps	0	0	0	0	2	2	5	4	3	3	2	1	2	0	2	1	6

Time step	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
illicit actor appears in 1 time step	77	122	405	612	692	376	532	1096	704	43	97	372	130	131	759	270	447
illicit actor appears in 2 − 4 time steps	4	5	20	16	18	17	40	9	21	1	7	52	11	24	23	7	11
illicit actor appears in ≥ 5 time steps	2	4	5	4	5	3	12	1	9	2	1	9	4	6	8	5	2

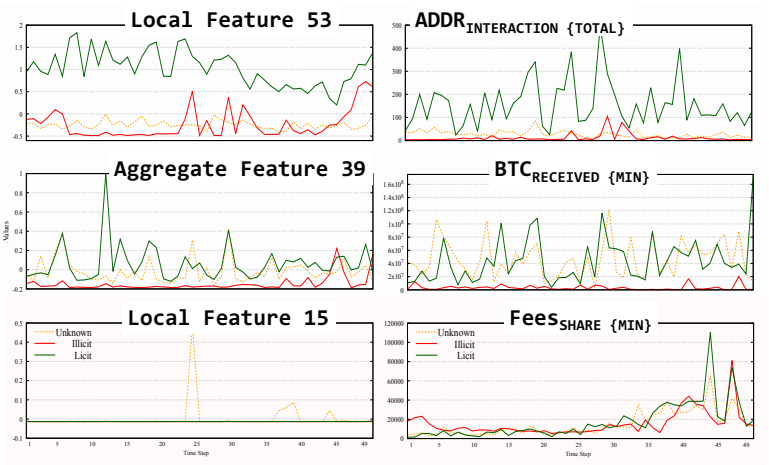
Time step	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
illicit actor appears in 1 time step	427	588	428	180	127	436	168	395	93	277	23	505	193	370	631
illicit actor appears in 2 − 4 time steps	9	5	5	9	0	14	8	12	11	6	9	4	11	17	20
illicit actor appears in ≥ 5 time steps	6	4	7	5	2	3	6	3	1	1	1	0	0	1	1

network which estimates the probability of an event, such as licit or illicit, based on independent variables. RF is an ensemble of classification trees trained on samples with random subsets of features for each decision, evaluating a final decision from averaging all decision trees. MLP is an artificial neural network with at least three layers where data features are fed into input neurons that assign probability vectors for classes as outputs. The Scikit-learn python library was used for LR (default parameters with 1000 max iterations), RF (default parameters with 50 estimators), and MLP (parameters: 1 hidden layer with 50 neurons, 500 epochs, Adam optimizer, 0.001 learning rate). LSTM is a recurrent neural network that has feedback connections and is capable of learning long-term dependencies (sequences of data as compared to single data points). The TensorFlow python library was used for LSTM (hyper-parameters: sigmoid activation, Adam optimizer, 30 epochs, binary cross-entropy loss function, 15 embedding output dims). XGB is a supervised learning algorithm which predicts variables by combining estimates of prior models. The XGBoost python library was used for XGB (default parameters with "multi:softmax" objective and 2 classes).

4.3 Fraud Detection Evaluation Metrics

The metrics used to verify the models were Precision (ratio of correct classifications), Recall (proportion of actual positive labels correctly classified), F1 Score (harmonic mean of precision and recall), and Micro-Avg F1 (Micro-F1) Score (ratio of correct classifications to total classifications). In some cases to distinguish between classifiers with close performance, the Matthews Correlation Coefficient (MCC) is used due to its suitability for unbalanced datasets and its prior use on financial fraud and cryptocurrency-related studies [1, 5, 40, 43]. We provide the formal definition of these metrics in Section A.2 of the Appendix. In addition, to gain deeper understanding of the different ML models and their classification performance for illicit transactions and illicit addresses, we conduct the following three case studies: (i) EASY cases: *all models classify an illicit transaction correctly*; (ii) HARD cases: *all models classify an illicit transaction incorrectly*; and (iii) AVERAGE cases: *some models failed to classify an illicit transaction but* ≥ 1 *models classified correctly.*

Figure 8: Trend comparison for selected features from the transactions (left) and actors (right) datasets. Green highlight shows good correlation, red shows unclear correlation.

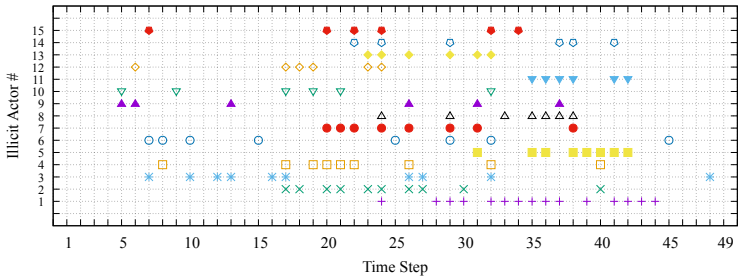


5 RESULTS AND ANALYSIS

5.1 Statistical Analysis of the Dataset

The building blocks of the Elliptic++ dataset are the transaction features and address features, which directly impact the quality of information gained by the models and quality of classification explainability into the root cause of fraudulent activities. Figure 8 shows three chosen features for the transactions (left) and actors (right) datasets, with feature values in y-axis and time steps in x-axis. We conjecture that important dataset features will show a reflective trend that clearly distinguishes between the illicit and licit classes. Such features will provide a level of interpretation into the risk assessment of a transaction or an actor. In Figure 8, the green curves of the features in the top two rows show the trend of licit transactions (left) and licit actors (right), which are distinctly different from the red curves of illicit transactions (left) and dishonest actors (right). Conversely, some features may not contribute to the detection of illicit transactions and actors, such

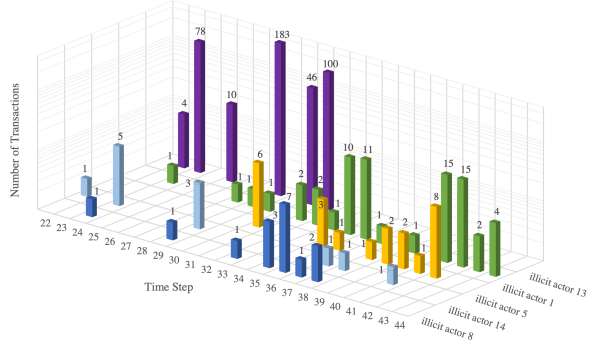
Figure 9: Timeline of illicit actors in ≥ 5 time steps, each point represents at least one illicit transaction involved in.



as the two features in the bottom row. For example, the curves for the illicit and licit transactions in Local Feature 53 can be clearly differentiated through visual analysis, while Local Feature 15 does not display any visual clue on illicit v.s. licit transactions over all time steps. Similar observations are found in the actors dataset.

We can further expand the statistical analysis on the behavioral trends of actors by their features captured in our Elliptic++ dataset, e.g., life span, number of transactions involved, and distribution of actors, which provide additional level of explainability to both the dataset and the detection model performance for fraudulent actors or classification of illicit/licit/unknown actors. Figure 9 shows the timeline of 15 illicit actors that transact in ≥ 5 time steps. For instance, Illicit Actor 1 transacts in 15 time steps, while Illicit Actor 15 transacts in only 6 time steps. A similar figure for illicit actors existing in only 1 time step (14,007 illicit actors) is included in the Appendix (see Figure 19). Moreover, the number of involved transactions within each time step varies across illicit actors as shown in Figure 10 for the five selected actors. Table 5 (shown on the previous page) provides the distribution of illicit actors across time steps categorized into 3 sets.

Figure 10: Number of transactions per time step for Illicit Actor 13 (purple), 1 (green), 5 (orange), 14 (Lblue), and 8 (Dblue).



Regarding the Bitcoin users, clustering addresses using the previously discussed four steps in Section 3.3.4 created 146,783 user entities. Table 6 shows some relevant statistics. Each user controlled a varying number of addresses, with 98.72% of users controlling ≤ 10 addresses, while only 0.02% of users controlled ≥ 1K addresses.

5.2 Model Evaluation and Analysis

Table 7 and Table 9 show the results of all models trained on the transactions dataset and the actors dataset respectively. From Table 7, the results for the transactions dataset in Elliptic++ (labelled TX) show an increase in performance in most of the metrics for all

Table 6: Statistics for Bitcoin users in the Elliptic++ Dataset.

# Users	146,783
# Addresses per User: Min	1
# Addresses per User: Median	1
# Addresses per User: Mean	2.73
# Addresses per User: Max	14,885
% Users w/ 1 – 10 Addresses	98.72%
% Users w/ 11 – 1K Addresses	1.26%
% Users w/ 1K – max Addresses	0.02%

of the models, compared to the Elliptic dataset (labelled EC). This can be attributed to our addition of 17 augmented features to each transaction, which in turn improves the generalization performance and the explainability of both the transaction dataset and the fraudulent transaction detection models. It is observed that RF^{TX} is the best-performing model (followed by XGB^{TX} and MLP^{TX}) with a precision of 97.5% and recall of 71.9%. Although RF^{TX} and RF^{EC} have comparable precision and recall performance, RF^{TX} is better as the MCC values are 0.83 vs 0.81. Table 7 also shows the results of 2– and 3–classifier ensembles by selecting the top 3 models (RF, XGB, MLP). The best performing 2– and 3–classifier ensembles are

Table 7: Illicit transactions results using individual/ensemble of classifiers. EC refers to classification on Elliptic dataset [36], TX is on our Elliptic++ transactions dataset.

Model	Precision	Recall	F1 Score	Micro-F1
LR^{EC}	0.326	0.707	0.446	0.886
LR^{TX}	0.328	0.707	0.448	0.884
RF^{EC}	0.940	0.724	0.818	0.979
RF^{TX}	**0.975**	**0.719**	**0.828**	**0.980**
MLP^{EC}	0.476	0.673	0.558	0.931
MLP^{TX}	0.611	0.613	0.612	0.949
$LSTM^{EC}$	0.665	0.350	0.459	0.946
$LSTM^{TX}$	0.709	0.223	0.339	0.942
XGB^{EC}	0.812	0.717	0.761	0.971
XGB^{TX}	0.793	0.718	0.754	0.969
***2 classifiers ensemble**, selecting top 3 classifiers*				
$RF+MLP^{EC}$	0.987	0.624	0.765	0.975
$RF+MLP^{TX}$	0.989	0.635	0.773	0.975
$RF+XGB^{EC}$	0.960	0.704	0.812	0.979
$RF+XGB^{TX}$	**0.977**	**0.706**	**0.820**	**0.979**
$MLP+XGB^{EC}$	0.457	0.737	0.564	0.926
$MLP+XGB^{TX}$	0.974	0.596	0.739	0.972
***3 classifiers ensemble**, selecting top 3 classifiers*				
$RF+MLP+XGB^{EC}$	0.947	0.719	0.817	0.979
$RF+MLP+XGB^{TX}$	**0.962**	**0.723**	**0.826**	**0.980**

Table 8: Classification results on the testing dataset showing distributions of EASY, HARD, AVERAGE cases among time steps 35 to 49. Total of each case is 49, 243, and 791 respectively. For the AVERAGE case, distributions are shown for cases where only 1 model (all shown), only 2 models (top 2 shown), only 3 models (top 2 shown), and only 4 models (top 1 shown) classified correctly.

Time Step		35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	TOTAL
EASY		32	0	2	5	5	1	1	3	0	0	0	0	0	0	0	49
HARD		4	0	10	7	4	28	6	36	22	20	4	1	21	27	53	243
AVERAGE	LR	0	0	3	0	2	3	0	6	2	3	1	0	1	9	2	791
	RF	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	
	MLP	0	0	1	1	0	2	0	2	0	0	0	0	0	0	0	
	LSTM	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	
	XGB	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	
	RF,XGB	4	0	0	1	2	1	17	2	0	0	0	0	0	0	0	
	LR,MLP	1	0	0	1	0	2	0	2	0	0	0	0	0	0	0	
	RF,MLP,XGB	5	6	0	8	3	4	1	0	0	0	0	0	0	0	0	
	LR,RF,XGB	6	1	10	27	18	10	5	21	0	0	0	0	0	0	0	
	RF,MLP,XGB,LR	124	24	12	57	45	55	81	159	0	1	0	1	0	0	0	

those using Elliptic++. In comparison, the LR, MLP^{EC}, and $LSTM$ models are ineffective with < 50% precision/recall (highlighted in red). Table 9 shows the results for the actors dataset. It is observed that RF^{AR} is the best-performing model (followed by XGB^{AR} and MLP^{AR}) with a precision of 91.1% and recall of 78.9%. Also, the best 2– and 3–classifier ensembles ($RF{+}XGB^{AR}$ and $RF{+}MLP{+}XGB^{AR}$) show increases in precision (95.9%,93.3% vs 91.1%), but decreases in recall (53.0%,57.2% vs 78.9%), indicating the member models of 2– and 3–classifier ensembles are not complimentary and have high negative correlation [39]. The LR^{AR} and $LSTM^{AR}$ models are ineffective with extremely low recall (highlighted in red).

5.3 EASY, HARD, and AVERAGE cases Analysis

To further understand the performance of RF models against other models, and the performance results of their best performing 2– and 3–classifier ensembles, we analyze their performance in terms of the EASY, HARD, and AVERAGE cases as defined in Section 4.3. Table 8 provides a temporal split of the classification results across the test time steps of 35 to 49 for the EASY case, HARD case, and AVERAGE case respectively. It is important to note that AVERAGE cases present the opportunities for further optimizations. For the AVERAGE cases (correct classification by $1 \leq x \leq 4$ models), all combinations are shown for each individual model, for the 2/3 models scenario we show top 2 combinations, and for the 4 models scenario we show top 1 combination. First, the case where the 4 models RF, MLP, XGB, and LR correctly classify the transaction accounts for 71% of the AVERAGE cases. Second, the cases where RF classifies incorrectly only make up for < 1% of the AVERAGE cases. This motivates us to focus on optimization of the RF model with feature refinement.

5.4 Model Optimization by Feature Refinement

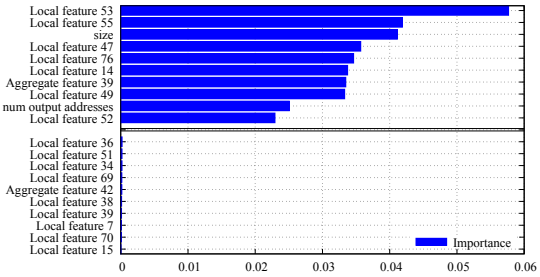
Given that the RF is the best performing model, we explore feature refinement to further optimize RF. By examining the results from decision trees, we combine feature importance, permutation feature importance, and drop column feature importance to show

Table 9: Illicit actors results using individual/ensemble of classifiers. AR is classification on our Elliptic++ actors dataset.

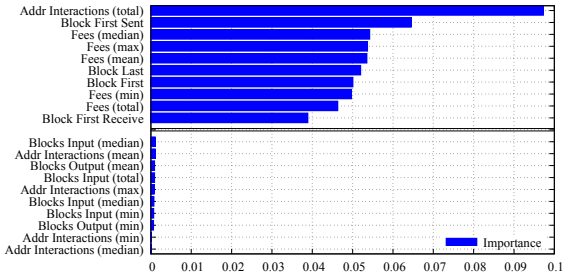
Model	Precision	Recall	F1 Score	Micro-F1
LR^{AR}	0.477	0.046	0.083	0.964
RF^{AR}	**0.911**	**0.789**	**0.845**	**0.990**
MLP^{AR}	0.708	0.502	0.587	0.974
$LSTM^{AR}$	0.922	0.033	0.064	0.965
XGB^{AR}	0.869	0.534	0.662	0.980
***2 classifiers ensemble**, selecting top 3 classifiers*				
$RF{+}MLP^{AR}$	0.967	0.403	0.568	0.978
$RF{+}XGB^{AR}$	**0.959**	**0.530**	**0.682**	**0.982**
$MLP{+}XGB^{AR}$	0.929	0.324	0.481	0.975
***3 classifiers ensemble**, selecting top 3 classifiers*				
$RF{+}MLP{+}XGB^{AR}$	**0.933**	**0.572**	**0.709**	**0.983**

the top 10 and bottom 10 features by importance on both transactions dataset and actors dataset, as shown in Figure 11a for TX and Figure 11b for AR. For the transactions dataset, the 17 augmented features produced a collective 12% importance, with the transaction size feature alone responsible for 4.1%. For the actors dataset, 35% of the features were responsible for 80% of the importance, with the total address interaction as the most important feature. Interestingly, a connection can be made to the set of top and bottom 10 features with the features highlighted in Figure 8. It can be seen that there is a link from the green highlight to the top 10 features, and likewise with the red highlight and bottom 10 features. This solidifies that features providing visual proof of the risk attribute to a larger classification importance, and vice versa. The top and bottom 2 of each feature type in both datasets are included in the Appendix Section A.5 for reference. Using this analysis, we run

Figure 11: Top 10 and bottom 10 features for the transactions dataset (left) and actors dataset (right).

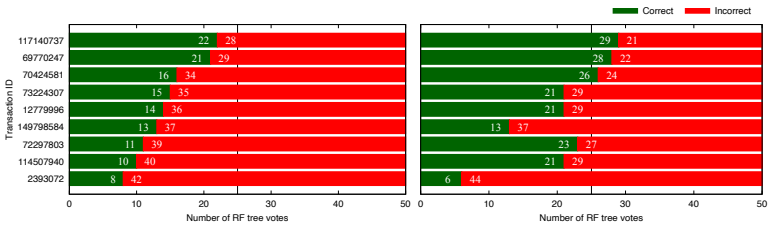


(a) Transactions dataset.

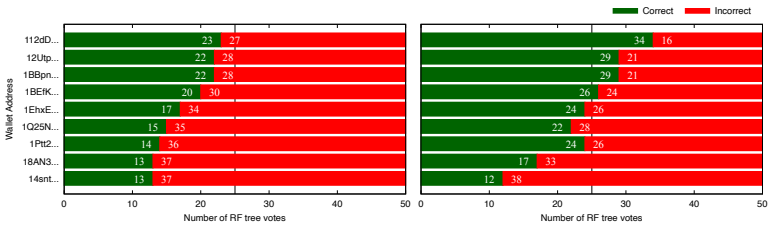


(b) Actors dataset.

Figure 12: Cumulative votes for 50 RF trees for chosen transactions (a) and addresses (b) before (left) and after (right) feature selection and refinement.



(a) Transactions dataset.



(b) Actors dataset.

the best 1−/2−/3−classifiers ensembles with selected features instead of the full set of features. Tables 10 and 11 show the model performance results for transactions and wallet addresses (actors) respectively. Interestingly, the feature refined models show an average improvement of 0.92%, 1.17%, and 0.89% for precision, recall, and F1 score respectively on the transactions dataset, and similarly 1.07%, 3.1%, and 1.13% on the actors dataset.

The increase in recall is also evident in the RF trees voting. Figures 12a and 12b show the cumulative votes of 50 RF trees for 9 chosen transactions/addresses before (left) and after (right) selecting important features for both the transactions and actors datasets respectively. This demonstrates a trend that when some non-contributing features are dropped, those transactions/addresses that are close to the correct manifold in classification will in turn be classified correctly by more trees, increasing in the number of correct votes.

6 CONCLUSION

With the rapid growth of the cryptocurrency ecosystem, there is a growing demand for robust financial forensics on the blockchain networks. This paper makes two original contributions. First, we construct the Elliptic++ dataset, the largest labelled dataset with both Bitcoin blockchain transactions and wallet addresses. This

Table 10: Illicit transactions classification results using selected features, labelled as ψ, Micro-F1 denotes Micro-Avg F1.

Model	Precision	Recall	F1 Score	Micro-F1
RF^{TX}	0.975	0.719	0.828	0.980
RF$^{TX\psi}$	**0.986**	**0.727**	**0.836**	**0.981**
RF+XGB^{TX}	0.977	0.706	0.820	0.979
RF+XGB$^{TX\psi}$	**0.987**	**0.717**	**0.826**	**0.980**
RF+MLP+XGB^{TX}	0.962	0.723	0.826	0.980
RF+MLP+XGB$^{TX\psi}$	**0.968**	**0.729**	**0.834**	**0.980**

Table 11: Illicit actors classification results using selected features, labelled as ψ and compared with AR. Micro-F1 denotes Micro-Avg F1.

Model	Precision	Recall	F1 Score	Micro-F1
RF^{AR}	0.911	0.789	0.845	0.990
RF$^{AR\psi}$	**0.921**	**0.802**	**0.858**	**0.990**
RF+XGB^{AR}	0.959	0.530	0.682	0.982
RF+XGB$^{AR\psi}$	**0.967**	**0.543**	**0.686**	**0.982**
RF+MLP+XGB^{AR}	0.933	0.572	0.709	0.983
RF+MLP+XGB$^{AR\psi}$	**0.945**	**0.601**	**0.718**	**0.984**

dataset will enable the applied data science researchers to conduct financial forensics on the Bitcoin cryptocurrency network and to develop effective fraud detection models and algorithms. Second, we leverage the four unique graph representations, to showcase the fraud detection of illicit transactions, illicit actors (wallet addresses), and the risks of de-anonymization of users using clustering of addresses. We analyze why Random Forest (RF) is the best-performing model for fraud detection, achieving 97.5% and 71.9% precision and recall respectively on the transactions dataset, and 91.1% and 78.9% on the actors dataset. Motivated by our ensemble learning analysis, we show that model training using selective features instead of all extracted features of transactions and of addresses, can further improve RF precision and recall performance by 0.92% and 1.17% respectively for the transactions dataset, and 1.07% and 3.1% respectively for the actors dataset. The Elliptic++ dataset is made publicly available at https://www.github.com/git-disl/EllipticPlusPlus.

REFERENCES

[1] Rachit Agarwal, Shikhar Barve, and Sandeep Kumar Shukla. 2021. Detecting malicious accounts in permissionless blockchains using temporal graph properties. *Applied Network Science* 6, 1 (2021), 1–30.

[2] Ismail Alarab and Simant Prakoonwit. 2022. Graph-based lstm for anti-money laundering: Experimenting temporal graph convolutional network with bitcoin data. *Neural Processing Letters* (2022), 1–19.

[3] Ismail Alarab, Simant Prakoonwit, and Mohamed Ikbal Nacer. 2020. Comparative analysis using supervised learning methods for anti-money laundering in bitcoin. In *Proceedings of the 2020 5th International Conference on Machine Learning Technologies*. 11–17.

[4] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating user privacy in bitcoin. In *International conference on financial cryptography and data security*. Springer, 34–51.

[5] John O Awoyemi, Adebayo O Adetunmbi, and Samuel A Oluwadare. 2017. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 international conference on computing networking and informatics (ICCNI)*. IEEE, 1–9.

[6] Massimo Bartoletti, Barbara Pes, and Sergio Serusi. 2018. Data mining for detecting bitcoin ponzi schemes. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 75–84.

[7] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. 2014. Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 15–29.

[8] Alex Biryukov and Sergei Tikhomirov. 2019. Deanonymization and linkability of cryptocurrency transactions based on network analysis. In *2019 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 172–184.

[9] Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*. Vol. 4. Springer.

[10] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[11] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[12] Weili Chen, Xiongfeng Guo, Zhiguang Chen, Zibin Zheng, and Yutong Lu. 2020. Phishing Scam Detection on Ethereum: Towards Financial Security for Blockchain Ecosystem.. In *IJCAI*. 4506–4512.

[13] Jason Hirshman, Yifei Huang, and Stephen Macke. 2013. Unsupervised approaches to detecting anomalous behavior in the bitcoin transaction network. *Technical report, Stanford University* (2013).

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[15] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[16] Yining Hu, Suranga Seneviratne, Kanchana Thilakarathna, Kensuke Fukuda, and Aruna Seneviratne. 2019. Characterizing and detecting money laundering activities on the bitcoin network. *arXiv preprint arXiv:1912.12060* (2019).

[17] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[18] Philip Koshy, Diana Koshy, and Patrick McDaniel. 2014. An analysis of anonymity in bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*. Springer, 469–485.

[19] Oliver Kramer. 2016. *Machine learning for evolution strategies*. Vol. 20. Springer.

[20] Sijia Li, Gaopeng Gou, Chang Liu, Chengshang Hou, Zhenzhen Li, and Gang Xiong. 2022. TTAGN: Temporal Transaction Aggregation Graph Network for Ethereum Phishing Scams Detection. In *Proceedings of the ACM Web Conference 2022*. 661–669.

[21] Dan Lin, Jiajing Wu, Qi Yuan, and Zibin Zheng. 2020. Modeling and understanding ethereum transaction records via a complex network approach. *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, 11 (2020), 2737–2741.

[22] Dan Lin, Jiajing Wu, Qi Yuan, and Zibin Zheng. 2020. T-edge: Temporal weighted multidigraph embedding for ethereum transaction network analysis. *Frontiers in Physics* 8 (2020), 204.

[23] Wai Weng Loa, Gayan K Kulatillekea, Mohanad Sarhana, Siamak Layeghya, and Marius Portmanna. 2022. Inspection-L: Self-Supervised GNN Node Embeddings for Money Laundering Detection in Bitcoin. (2022).

[24] Joana Lorenz, Maria Inês Silva, David Aparício, João Tiago Ascensão, and Pedro Bizarro. 2020. Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of label scarcity. In *Proceedings of the First ACM International Conference on AI in Finance*. 1–8.

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

[26] Patrick Monamo, Vukosi Marivate, and Bheki Twala. 2016. Unsupervised learning for robust Bitcoin fraud detection. In *2016 Information Security for South Africa (ISSA)*. IEEE, 129–134.

[27] Patrick M Monamo, Vukosi Marivate, and Bhesipho Twala. 2016. A multifaceted approach to Bitcoin fraud detection: Global and local outliers. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 188–194.

[28] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review* (2008), 21260.

[29] Catarina Oliveira, João Torres, Maria Inês Silva, David Aparício, João Tiago Ascensão, and Pedro Bizarro. 2021. GuiltyWalker: Distance to illicit nodes in the Bitcoin network. *arXiv preprint arXiv:2102.05373* (2021).

[30] Thai Pham and Steven Lee. 2016. Anomaly detection in bitcoin network using unsupervised learning methods. *arXiv preprint arXiv:1611.03941* (2016).

[31] Thai Pham and Steven Lee. 2016. Anomaly detection in the bitcoin system-a network perspective. *arXiv preprint arXiv:1611.03942* (2016).

[32] D. Dziubaltowska R. Michalski and P. Macek. 2020. Revealing the character of nodes in a blockchain with supervised learning. *IEEE Access* (2020).

[33] Fergal Reid and Martin Harrigan. 2013. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*. Springer, 197–223.

[34] Jie Shen, Jiajun Zhou, Yunyi Xie, Shanqing Yu, and Qi Xuan. 2021. Identity inference on blockchain using graph neural network. In *International Conference on Blockchain and Trustworthy Systems*. Springer, 3–17.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[36] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591* (2019).

[37] Jiajing Wu, Jieli Liu, Weili Chen, Huawei Huang, Zibin Zheng, and Yan Zhang. 2021. Detecting mixing services via mining bitcoin transaction network with hybrid motifs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52, 4 (2021), 2237–2249.

[38] Jiajing Wu, Qi Yuan, Dan Lin, Wei You, Weili Chen, Chuan Chen, and Zibin Zheng. 2020. Who are the phishers? phishing scam detection on ethereum via network embedding. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2020).

[39] Yanzhao Wu, Ling Liu, Zhongwei Xie, Ka-Ho Chow, and Wenqi Wei. 2021. Boosting ensemble accuracy by revisiting ensemble diversity metrics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16469–16477.

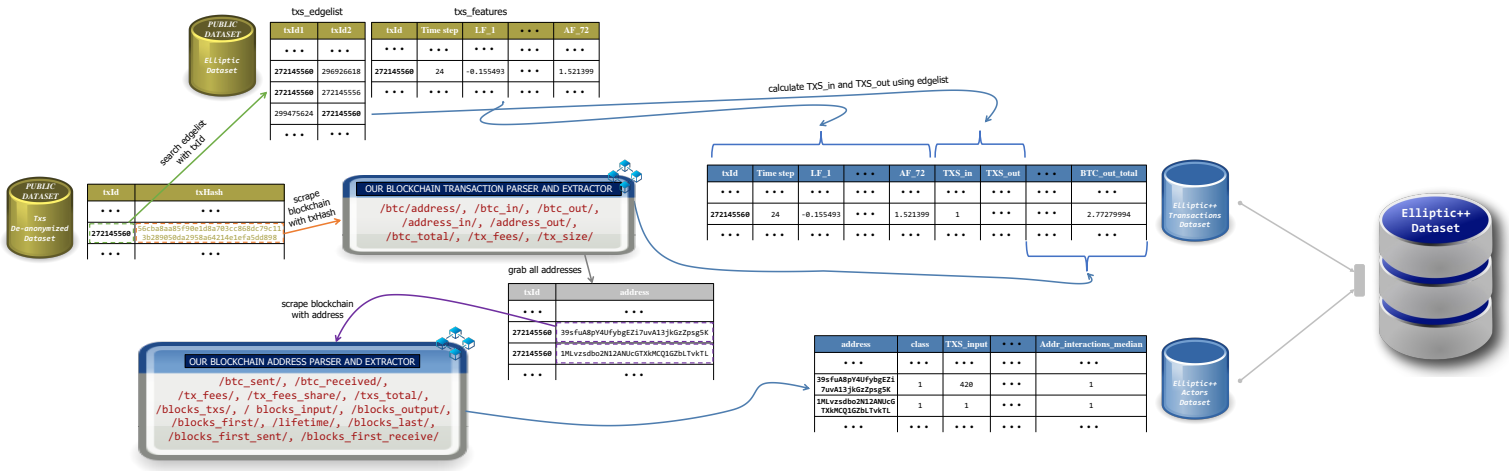
[40] Masoumeh Zareapoor, Pourya Shamsolmoali, et al. 2015. Application of credit card fraud detection: Based on bagging ensemble classifier. *Procedia computer science* 48, 2015 (2015), 679–685.

[41] Yuhang Zhang, Jun Wang, and Jie Luo. 2020. Heuristic-based address clustering in bitcoin. *IEEE Access* 8 (2020), 210582–210591.

[42] Jiajun Zhou, Chenkai Hu, Jianlei Chi, Jiajing Wu, Meng Shen, and Qi Xuan. 2022. Behavior-aware Account De-anonymization on Ethereum Interaction Graph. *arXiv preprint arXiv:2203.09360* (2022).

[43] Francesco Zola, Maria Eguimendia, Jan Lukas Bruse, and Raul Orduna Urrutia. 2019. Cascading machine learning to attack bitcoin anonymity. In *2019 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 10–17.

Figure 13: Elliptic++ dataset collection pipeline using our blockchain parsers and extractors.



A APPENDIX

A.1 Dataset Collection

Figure 13 shows the Elliptic++ dataset collection pipeline. In the data collection pipeline for the Elliptic++ transactions dataset, data is gathered from two sources as introduced in Section 3. All features from the Elliptic dataset are included in the transactions dataset, along with two augmented features, TXS_in and TXS_out which are calculated by counting the number of incoming and outgoing edges for each transaction using the edgelist. Moreover, the transaction hash from transactions in the deanonymized transactions dataset, are fed into our blockchain transaction parser and extractor which scrapes the public blockchain for information regarding the transaction including number of input/output addresses, amount of incoming/outgoing BTC, and much more. These augmented features are combined with the previous features to create the Elliptic++ transactions dataset. In the data collection pipeline for the Elliptic++ actors dataset, output is received from the previous blockchain transaction parser and extractor in the form of a set of input and output addresses for each transaction. Each address is then fed into our blockchain address parser and extractor which scrapes the public blockchain for transaction and time related information. These are all combined to create the Elliptic++ actors dataset. Both these datasets encompass the Elliptic++ dataset.

A.2 Fraud Detection Evaluation Metrics

The accuracy of the models is verified using five metrics: Precision, Recall, F1 Score, Micro-Avg F1 Score, and Matthews Correlation Coefficient. The explanations have the following components: true positive (TP), false positive (FP), false negative (FN), and true negative (TN).

(1) *Precision* - the fraction of positive points that were correctly classified. It is calculated by:

$$Precision = \frac{TP}{TP + FP}$$

(2) *Recall* - the fraction of actual positive points that were correctly classified. It is calculated by:

$$Recall = \frac{TP}{TP + FN}$$

(3) *F1 Score* - the harmonic mean of precision and recall. It is calculated by:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

(4) *Micro-Avg F1 Score* - similar to F1 Score, but using the total number of TP, FP, and FN instead of individually for each class. It is calculated by:

$$MicroAvg \;\; F1 = \frac{TP}{TP + \frac{1}{2} \cdot (FP + FN)}$$

(5) *Matthews Correlation Coefficient (MCC)* - measures the quality of binary classifications, taking into account the difference between predicted values and actual values. It ranges from $[-1, +1]$ with the extreme values -1 and $+1$ in cases of perfect misclassification and perfect classification respectively. It is calculated by:

$$MCC = \frac{(TP \cdot TN) - (FP \cdot FN)}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + FP) \cdot (TN + FN)}}$$

Figure 14: Distribution of transactions in time steps 14 (left) and 32 (right).

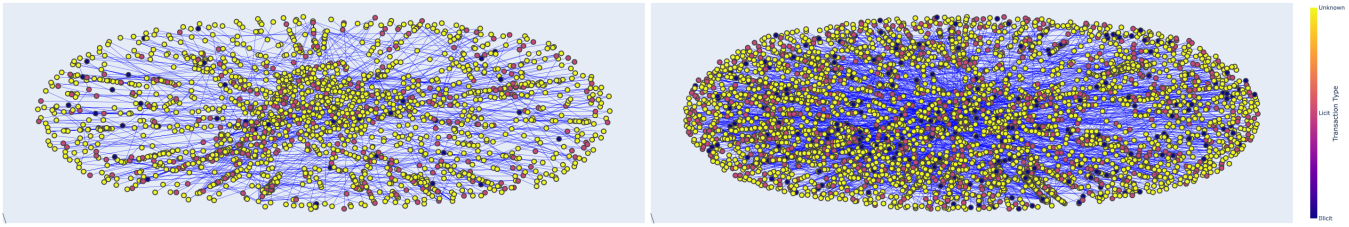


Figure 15: Distribution of wallet addresses in time steps 14 (left) and 32 (right).

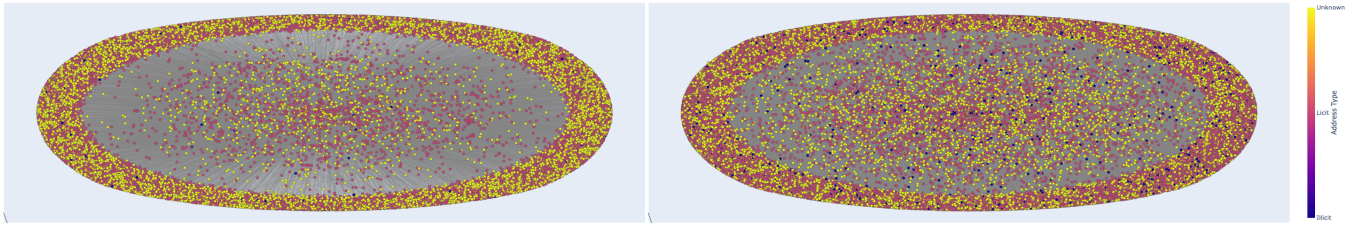
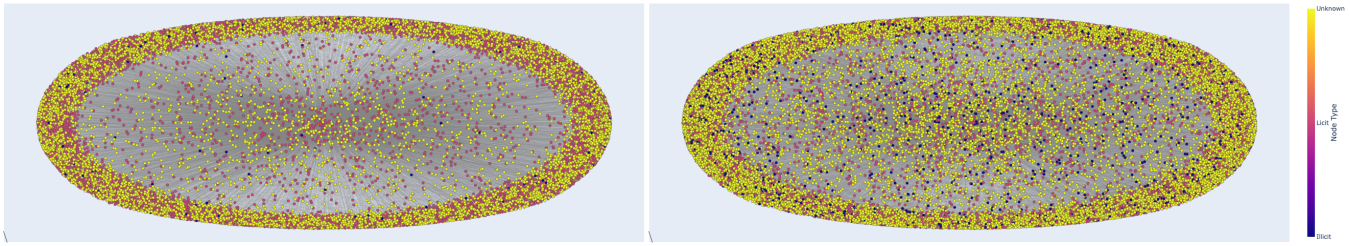


Figure 16: Distribution of transactions and wallet addresses in time steps 14 (left) and 32 (right).



A.3 Graph Visualization

Figures 14, 15, and 16 show the distributions of transactions and/or wallet addresses for the Money Flow Transaction Graph, the Actor Interaction Graph, and the Address-Transaction Graph respectively. Specifically, they show a comparison between distributions in time step 14, which produce more sparse graphs, and time step 32, which produce more dense graphs (the grey area is due to the density of edges).

A.4 Bitcoin Address Clustering

The Bitcoin address clustering process discussed in this study is referred to as the *multiple input addresses heuristic*. This heuristic takes advantage of the fact that in order to transact correctly, a user must provide the signatures corresponding to the private keys and the public key wallet addresses for all the input addresses, thus, a user must have access over all private keys. Hence, allowing for the assumption that all input addresses of a transaction can be attributed to the same user. This is demonstrated in Figure 17, where the input addresses `addr1`, `addr2`, `addr3`, and `addr4` are grouped into User U_1, while input addresses `addr5` and `addr6` are grouped into User U_2. Figure 18 shows the address clustering process using the four steps discussed in Section 3.3.4, producing a user graph on the right.

Figure 17: Bitcoin address clustering based on multiple input addresses heuristic in an address-transaction graph.

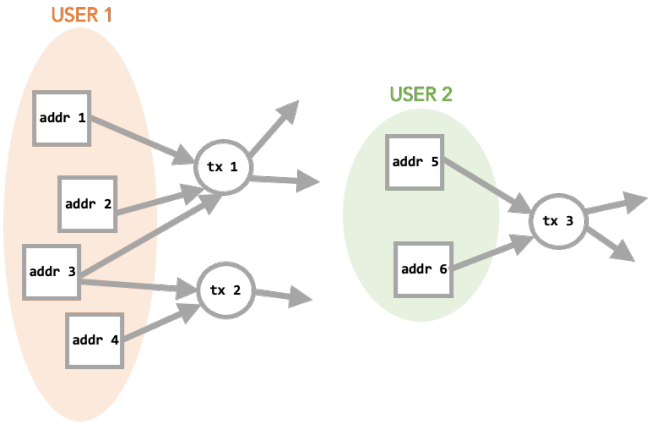
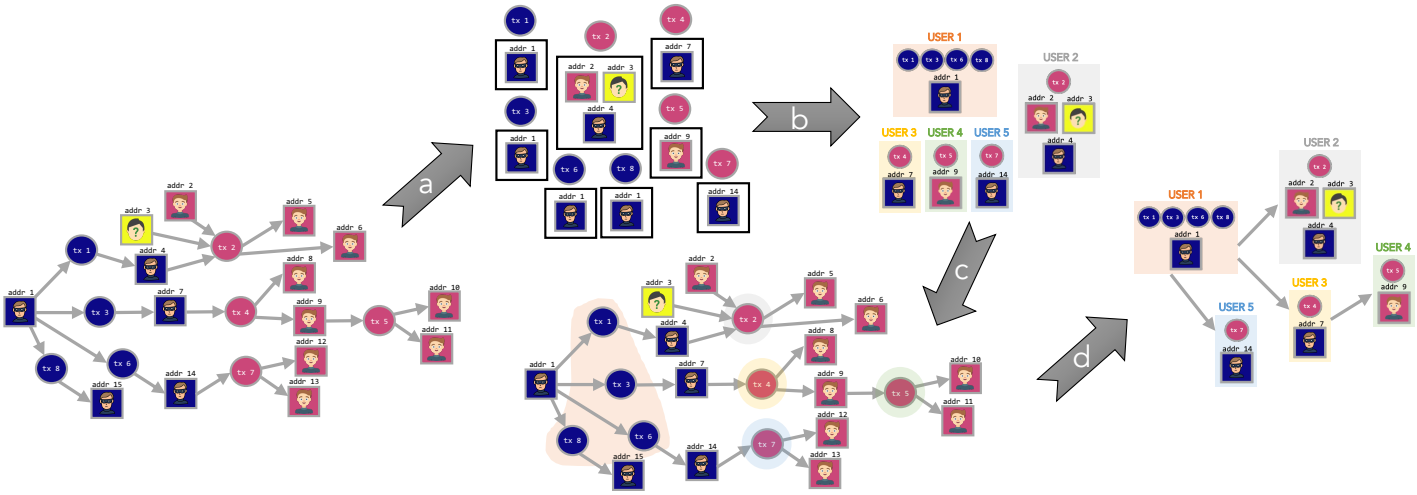


Figure 18: Clustering Bitcoin addresses: (a) collect the set of input addresses for each transaction in the address-transaction graph; (b) group transactions with ≥ 1 common element in each set into unique users; (c) highlight transactions within each user in the original graph; and (d) convert highlighted address-transaction graph into a user graph, with edges across users if a path existed between groups of transactions in the original graph.



A.5 Statistical Analysis of the Dataset

Figure 19 shows illicit actors that exist in only 1 time step (14, 007 total illicit actors). Aside from providing us information about the timeline of the illicit actors, it displays the distribution of the illicit actors among the dataset for these types of actors. For example, in time step 6 there are 8 illicit actors, in time step 14 there are 74 illicit actors, and in time step 32 there are 759 illicit actors.

Figure 20 is an extension of the dataset feature analysis and feature refinement by displaying the top 2 and bottom 2 of each feature class in (a) the transactions dataset (left: local features, right: aggregate features), and (b) the actors dataset (left: transactions features, right: time features). This further solidifies the explainability behind the risk of transactions and actors through the visual (and numerical) distinction among the illicit and licit classes. In all green

Figure 19: Number of transactions for illicit actors appearing in only 1 time step. Each point is a unique illicit actor. Y-axes are in log scale.

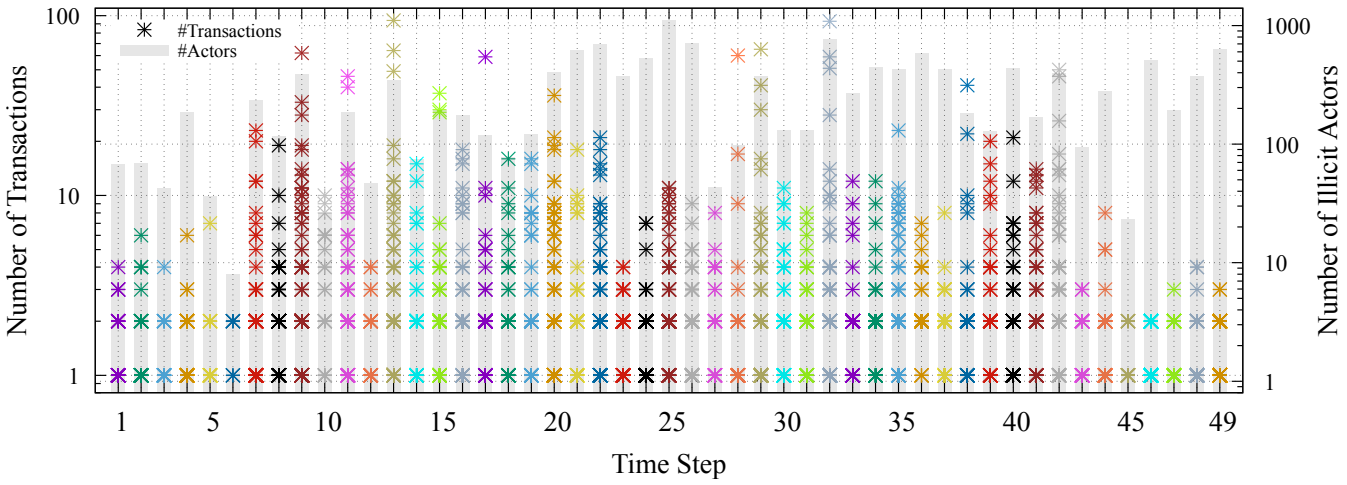
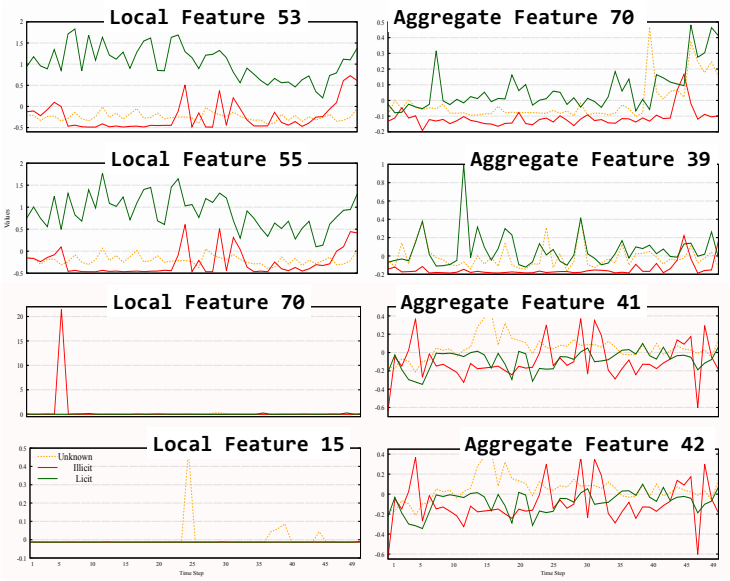
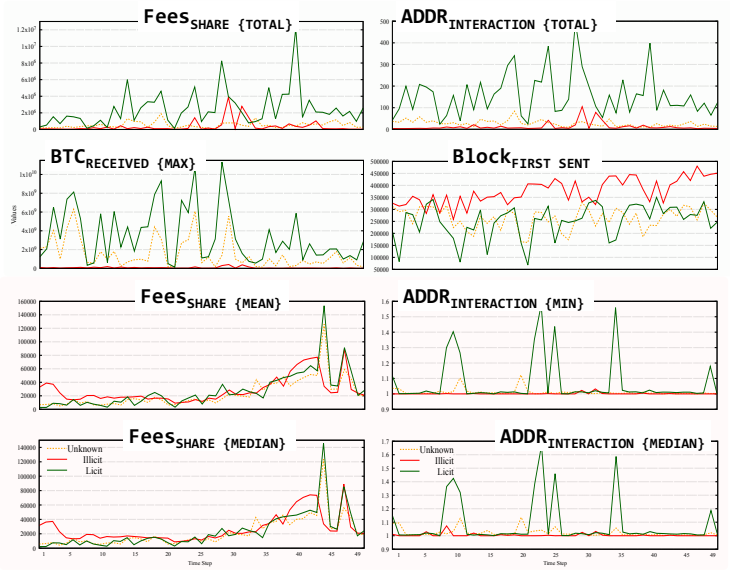


Figure 20: Trend comparison for features in the transactions dataset (left) and actors dataset (right). Green highlight shows good correlation, while red highlight shows unclear correlation.

(a) Comparison of local features (left) and aggregate features (right) in the transactions dataset.



(b) Comparison of transaction features (left) and time features (right) in the actors dataset.



highlighted figures, one can clearly classify between both classes. Though, this is unlike the red highlighted figures where there is no characteristic trend among both classes.

A.6 Data Distribution

Tables 12 and 13 show the numerical distribution within each class by time step for the transactions and actors datasets respectively.

Table 12: Numerical distribution of the number of transactions of each class by time step. First two tables are for the training set and the third table is for the testing set.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Unknown	5733	3427	5342	4253	4921	3843	4845	3292	4218	5755	3600	1541	3719	1605	3021	2445	2574
Illicit	17	18	11	30	8	5	102	67	248	18	131	16	291	43	147	128	99
Licit	2130	1099	1268	1410	1874	480	1101	1098	530	954	565	490	518	374	471	402	712

Time step	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Unknown	1587	2761	3391	2896	4131	2978	3466	1720	2006	883	1369	3101	1959	2106	3202	2710	1971
Illicit	52	80	260	100	158	53	137	118	96	24	85	329	83	106	342	23	37
Licit	337	665	640	541	1605	1134	989	476	421	182	199	845	441	604	981	418	478

Time step	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
Unknown	4166	4685	2808	2135	1577	3270	4210	4986	3693	3384	4377	2807	4275	2483	1978
Illicit	182	33	40	111	81	112	116	239	24	24	5	2	22	36	56
Licit	1159	1675	458	645	1102	1099	1016	1915	1346	1567	1216	710	824	435	420

Table 13: Numerical distribution of the number of wallet addresses of each class by time step. First two tables are for the training set and the third table is for the testing set.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Unknown	26465	15878	26213	31035	24621	17029	23357	17631	23819	38271	18250	10343	26813	5912	11500	9064	17066
Illicit	96	99	58	266	54	17	463	267	960	120	494	77	918	156	447	427	321
Licit	24547	16292	4835	8647	10465	1957	7902	16229	3894	8454	10576	2875	3288	1941	6671	2315	4039

Time step	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Unknown	6413	11633	32483	21725	28827	14101	19396	12231	16434	9130	6489	17180	8882	13328	15679	10109	9914
Illicit	189	320	950	1256	1491	734	1083	2243	1334	89	219	979	287	312	1730	561	908
Licit	1643	3220	6315	16576	9693	7301	3518	8314	2360	1000	920	3811	2931	5119	4873	7675	5546

Time step	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
Unknown	23898	34887	16578	16820	9679	23564	22123	31496	24953	16255	22201	17719	19726	13638	10030
Illicit	943	1226	804	400	282	801	454	967	156	497	42	626	256	433	789
Licit	6229	11130	3455	4039	4881	10233	7033	11142	7302	10631	11057	4723	8106	8783	4385