

Illicit Bitcoin Wallets' Modi Operandi and Sub-Network Discovery using Stochastic Block Modeling

Petre-Cornel GRIGORESCU*

Bucharest University of Economic Studies, Bucharest, Romania

*Corresponding author, grigorescupetre19@stud.ase.ro

Abstract. *Financial crime in cryptocurrency networks remains difficult to manage as illicit actors camouflage themselves in the noisy open network and rely on complex obfuscation patterns or modi operandi. Tested on a real Bitcoin dataset of 49 snapshots in time, this paper evaluates whether nested, degree-corrected stochastic block modeling can effectively reveal the sub-networks composed of illicit actors beyond simple graph connectivity or numerical attributes and demonstrates the practical applicability of such models for forensic analytics by significantly scoping wide, noisy networks into actionable blocks based on the behavior of wallets.*

Keywords: financial crime detection, cryptocurrency, stochastic block modeling, unsupervised machine learning, elliptic++ dataset.

Introduction

Recent developments in the state-of-the-art machine learning field have brought new, more sophisticated methods for network analysis, subnetwork discovery and unsupervised modeling of graphical data such as payments, including in cryptocurrencies such as Bitcoin. Currently, many of these methods are yet untested for some of the most critical fields in our society, including the detection and prevention of financial crime, which is one of the biggest hurdles of modern institutions when dealing with fraud through cryptocurrencies. In this paper, we assess the performance of a semi-supervised application of stochastic block modeling, which on its own is unsupervised, and evaluate its contribution to the discovery of hidden illicit actors by inferring role/propensity-based blocks, or groups of wallets with similar characteristics and connectivity profiles, rather than pure community partitioning. To this end, we use 49 snapshots of the Bitcoin network as made available by the Georgia Institute of Technology with the Elliptic++ dataset (Elmougy & Liu, 2023). The potential results could unlock the possibility for implementation of stochastic block modeling in financial institutions and regulatory bodies for improved investigative operations, automated black and gray lists of wallets and better regulatory control of cryptocurrency exchanges and key IBANs.

Literature review

Upon review at the time of writing this section, existing literature does not cover, to our knowledge, the application of Stochastic Block Modeling, also known as Bayesian Stochastic Block Modeling (Peixoto, 2017) within the area of semi-supervised or unsupervised cryptocurrency grouping for financial crime prevention and detection, gaps which this paper seeks to address. Stochastic applications on cryptocurrencies are relatively niche, but have nonetheless brought scientifically important contributions, such as the work of (Alves et al., 2020) on applying stochastic block modeling to stock market dynamics for probing the efficient market hypothesis, as well as (Færch et al., 2021), who brought the method to the Twitter information network with the purpose of exploring community interactions and unravel the network structure into its separate collaborative

components. A detailed survey by (Kang et al., 2020) and our own targeted search reveals that stochastic block modeling has not been applied directly to transactional cryptocurrency data, including Bitcoin data, although other stochastic models were tested previously with mixed results, such as Discrete-time Markov Processes, Continuous-time Markov Processes, Markov Decision Processes, Queueing Models and Hidden Markov Models. Of these, the closest applications to our paper’s intended contribution are those dealing with behavioral actor grouping, but these are done for better understanding and profiling crypto network dynamics such as mining policies and rewards, consensus, endorsement and validation within networks and the interaction of actors (wallets) with regards to these concepts.

The dataset we use is the *Elliptic++ Data Set* created by the Georgia Institute of Technology (Elmougy & Liu, 2023), a large set of real Bitcoin transactions designed for forensic research, essentially composing in its totality a graph of unique Bitcoin addresses (wallets) connected by transfers of the digital commodity. The team that curated the dataset additionally provides valuable labeling information for a number of these transactions, summarized in Table 1.

Table 1. Summary of dataset details, values and notes, split by category

Category	Metric	Value	Notes
Source	Name	Elliptic++ Data Set	Created by Georgia Institute of Technology (Elmougy & Liu, 2023)
Source	Scope	Bitcoin transaction forensics	Graph of unique Bitcoin addresses (wallets) connected by transfers
Labels	Class definitions	Class 1 = illicit Class 2 = licit Class 3 = unknown	Illicit if initiating entity linked to AML-blacklisted address (e.g., scams, malware, ransomware, terrorist financing)
Labels	Labeling constraint	Excludes multi-entity-controlled inputs	Leads to ~2% positive (illicit) rate
Features (transactions)	Engineered feature breakdown	Up to 166 engineered features	94 local (observation-level) + 72 aggregated (± 1 hop in graph)
Features (wallets)	Wallet feature count	56	Wallet/address-level features

Source: author’s own elaboration based on information provided by (Elmougy & Liu, 2023).

One of the most important features for this paper is the target variable (class of wallets/transactions), which we summarize in Table 2 as follows. Of these, we are particularly interested in wallet-related information, as the partitioning concerns actors and not their transactions.

Table 2 Summary of label details, value counts, by label class

Dataset	Nodes	Edges	Extra edge info	Illicit (Class 1)	Licit (Class 2)	Unknown (Class 3)
Elliptic++ Transactions	203,769 (transactions)	234,355 (money flows)	-	4,545 flows	42,019 flows	157,205 flows
Elliptic++ Actors (Wallet Addresses)	822,942 (wallet addresses) 1,268,260 (interaction nodes)	2,868,964 (addr-addr)	1,314,241 (addr-tx-addr)	14,266 wallets	251,088 wallets	557,588 wallets

Source: author’s own elaboration based on information provided by (Elmougy & Liu, 2023).

The Elliptic++ dataset is often used as a sandbox environment for testing and improving state-of-the-art (SOTA) models, as well as revamping traditional machine learning methods by enriching the available training information using various methods. Some of the more notable works have thus far tested random forest, logistic regression and XGBoost using both the raw feature sets or using embeddings from Graph Convolutional Networks (GCN) (Weber et al., 2019; Vassallo et al., 2021) with the GCN-integrated results typically exceeding performance on the raw feature set, albeit these results being on the Elliptic dataset, and not its upgraded Elliptic++ version. Customized models have also been produced and have typically broken performance records on the test set in succession, such as (Lo et al., 2022) and more recently (Ferretti et al., 2024), with the latter being the only work that specifically tackles wallet (actor)-level classification. As such, Table 3 will serve as our sparring benchmark for evaluating classification results, in the absence of other identifiable wallet-level benchmarks on the Elliptic++ dataset.

Table 3. Champion wallet-level benchmarks on the Elliptic++ dataset classification task

Rank	Model	Precision	Recall	F1	F1-micro
1 (tie)	Heter GAT	0.922	1.000	0.959	0.921
1 (tie)	HAN	0.922	0.999	0.959	0.921
3	Heter SAGE	0.948	0.897	0.922	0.860
4	HGT	0.975	0.820	0.891	0.815

Source: “Table 4: Metrics for wallets” (Ferretti et al., 2024).

The authors of Table 3 report the heterogeneous graph attention network (Heter-GAT) as the highest performing model in their experiment, tied with a Hierarchical Attention Network (HAN), both deep learning models that can learn from the network structure of the dataset as well as from the raw tabular features. The results, however, do not necessarily lead to the discovery of unknown illicit wallets from a small number of labeled ones, which makes these models excel at beating the classification task, but does not provide additional information beyond what is already known to improve the effectiveness of investigations. The standard for reporting results lies in the metrics displayed in Table 3, namely Precision, Recall, the F1 Score, and F1 (micro average). F1 Score finds the overall balance between recall and precision through a harmonic mean calculation, resulting in a balanced evaluation of test performance, essentially giving equal weight to all classes, while the micro-average variant practically shows the F1 Score for the positive class (Takahashi et al., 2022).

This gap is what motivates the core component of our analysis: the Nested Degree-Corrected Stochastic Block Model (nested DCSBM), a method of unsupervised structural inference or more simply put, network and sub-network detection in graphs, directly applicable to the exploration and analysis of transactions (Vu-Le et al., 2025). Standard Stochastic Block Models (Standard SBMs) split the network into “blocks” whereby the nodes party to the same block are stochastically equivalent and are thus conditionally interchangeable within the same block, having approximately the same probability of connection(s) to any other block. Nodes sharing a block will thus “behave” in fundamentally the same manner (playing the same role) within their respective sub-network. That being said, standard SBMs perform well in capturing the behavior or role structure (propensity) within a network based on the expected number of degrees/connections, and the blocks will be formed mainly based on inter-block connections (Lee & Wilkinson, 2019).

Standard SBMs are thus rudimentary in their partitioning capabilities due to the overpowering contribution of the expected node degrees. Contrasting to standard SBMs, DCSBMs add a necessary layer of complexity to resolve the limitation of equivalent expected degrees within

the same block by normalizing the blocks with the degrees, allowing for variation in node propensity and enabling the capture of central nodes (hubs) as well as sparse connections (strays) that play the same role with radically different connectivity. An added benefit of correcting degrees is that the blocks are no longer interpretable mainly by connectivity and more by propensity (Funke & Becker, 2019). Nesting within the context of (DC)SBMs induces the model to fit the data hierarchically, meaning that we expect there to be sub-blocks to the base blocks. This is performed in order to get more granular results or more heterogeneous blocks, effectively capturing the network structure at multiple scales (Peixoto, 2017; Peixoto, 2023). Unlike modularity or map-equation algorithms that primarily recover dense communities (Held et al., 2016), SBMs can recover structural roles, such as hubs, collectors, dispersers, as well as heterogeneous connection properties, which match our goal of *modi operandi* discovery.

Most work on the Elliptic++ dataset frames the task primarily as transaction classification under a temporal split, whereas our focus is on behavioral partitioning + investigative scoping and only secondarily on downstream classification. On a technical level, the objective of this paper differs in two ways from related work in the sense of testing a single unified model's performance for illicit node detection:

1. We incorporate a first stage unsupervised graph partitioning model into a second stage supervised classification model.
2. We analyze and propose a parametrizable method of implementing the results of the first stage model based on risk appetite.

While community-detection methods such as Louvain, Leiden or Infomap are natural comparators for the first stage model, they optimize objectives (modularity, map equation) that target densely connected communities, whereas our first stage partitioning is a degree-corrected generative block model meant to recover role-based blocks that are not necessarily community-like. As the downstream features and risk-thresholding are defined from the SBM output blocks, replacing SBM with a modularity-based model will not produce a direct comparison without heavy adjustments to the design of our pipeline. We therefore treat cross-algorithm substitution and comparison for the first stage model as separate work, warranting its own future research.

Methodology

During the data exploration phase, we take a closer look at the wider network within the data through charts such as wallet and transaction labels' prevalence across time, density of nodes and edges, as well as the proximity of the average wallet to an illicit one. These serve to understand the dynamics of the data captured and better prepare the modeling and post-hoc validation of the model.

We model this network as a wallet-to-wallet graph, and due to transactions with multiple inputs/outputs present (unlabeled) we induce the edges between wallets and weigh them by multiplicity. Related work has studied the viability of the Louvain algorithm for clustering the Elliptic dataset (Zheng et al., 2020) and while there is no existing Infomap application on the same dataset as ours, (Qiu et al., 2026) model risk associated with cryptocurrencies (among which is also Bitcoin), using Infomap as one of their methods for analyzing large-scale networks and integrating the results with second stage models, a technique similar to ours.

In the unsupervised modeling part of this experiment set we use a DCSBM described in the literature review, as implemented in the graph-tool library for Python (Peixoto, 2023). The DCSBM takes nodes (wallets) and inferred edges (transactions) as inputs and seeks to arrange them in a hierarchical partitioning such that the graph entropy is minimized, leading to the most likely

posterior distribution of nodes into blocks, based on a number of refinement iterations, and for this process the algorithm makes use of Markov Chain Monte Carlo (MCMC) simulations that explore the space of possible partitions and identify the most efficient way to partition the nodes. In our study we look at the most granular level (0) of the hierarchy, though 16 levels, with small variations, are identified globally and within each time step. We evaluate model performance in two experiments: illicit/infected wallet network, where the entire network is virtually illicit (to mainly reveal *modi operandi*); full network, where we evaluate performance on behavior-based grouping for a 2% positive rate. The application of this model type is in line with the validation results provided by (Funke & Becker, 2019) as well as more recently validated work by (Vu-Le et al., 2025). DCSBMs minimize the entropy of the graph based on a non-parametric Bayesian probabilistic algorithm and determines the optimal number of partitions (blocks) and hierarchical structure (depth) without any pre-specified starter number of partitions, making this a wholly unsupervised algorithm within a semi-supervised applied framework, as labels are used strictly for post-hoc validation and not as inputs for the model. In order to validate any technical contributions with regards to the formed blocks, we train a number of surrogate classification models (RF, MLP, GCN, XGBoost, SAGE) on the same data but switching label sets in and out depending on label propagation, discussed at a later point in the paper.

Inference stability

As the name reveals, stochastic block modeling relies on probabilistic assignment and refinement of nodes (wallets) into blocks, as opposed to classic partitioning or density-based clustering algorithms. This leads to the valid concern of result stability across runs and how instability impacts the feasibility of productionizing this framework. To solve this issue, we propose employing ensemble consensus for clustering, which aligns and re-aligns block assignments from an arbitrary number of fitted SBMs on the same data but with different random initializations, which in turn discover different local minima and thus multiple ways of partitioning the network. The core of the ensemble alignment lies in the graph-tool library's PartitionModeState class which is designed exactly for this issue of label switching in SBMs; the class assigns one SBM run as a reference and mathematically aligns all other runs to it by calculating maximum overlap between blocks and depending on alignments, renames e.g. "Block 15" to "Block 3" if these are equivalent, ultimately aligning most of the equivalent blocks to a single name. From these aligned blocks, consensus is determined by re-allocating each wallet (node) to the block it was most often allocated to across the runs. This consensus method, as well as others, are described in detail in their paper of reference (Peixoto, 2021).

Analogous to how random forest algorithms use trees as weak predictors, the general expectation is that variability is reduced the larger the ensemble is, with increasingly marginal improvements after a certain point. We test up to 50 individual SBMs per time step and ultimately settle on an ensemble of 30 runs due to computational constraints, as one pipeline with 50 SBMs takes more than 10 hours to complete. In order to assess the stability of the output we make use of the J-Score, a Jaccard Index-based measure of cluster accuracy as proposed by (Ahmadinejad & Liu, 2021). The method essentially identifies the best bi-directional match between runs for each true class T ($T \oplus C$) and cluster C ($C \oplus T$) separately, computes weighted sums of these best-matching Jaccard values, before finally computing the J-Score as the harmonic mean of the two sums. In our pipeline, this is done using bi-directional pairing across 5 runs, resulting in 10 pairs, the J-Scores of which we take the mean and standard deviation for later dissemination in the results section. This test is performed on time steps 44-49 and the partitioning, ensemble consensus, and

J-Score calculations are all done independently for each of the 6 snapshots. The reasoning for using J-Score instead of Jaccard stems from the probabilistic nature of SBMs, similar to the rationale for ensemble partitioning. SBM partitioning will still vary across runs regardless of how much stabilization is enforced as it splits large, stable communities into smaller, slightly varied ones. The Jaccard Index would penalize these structural splits because of its 1-to-1 matching, while the J-Score still mathematically accounts for the structural cohesion of similar blocks even if a wider community were fragmented across runs. Lastly, we compute 95% confidence intervals using non-parametric bootstrapping over 1000 iterations per time step.

Visual exploration

We extract block-level features to profile the blocks and visualize the results using large graphs, split by time steps, through self-integrated HTML “maps” with nodes colored by block assignment, as well as portrayed with different markers depending on label (star = illicit, diamond = direct neighbor to illicit / infected; circle = unlabeled) and some excerpts of these maps are included in the paper’s results section.

We split the label propagation into four interchangeable methods, as follows:

1. Intra-block label propagation
2. Inter-block label propagation
3. Intra+Inter-block label propagation
4. Inter+intra block label propagation

Each of these scenarios creates, essentially, a new set of training set labels by filling in the unknowns based on the SBM block assignments independently within each time step. We perform intra-block label propagation by extending a certain label, be it licit or illicit, to the “unknown”-labeled wallets inside the same block, based on the prevalence of those classes. Going from 0.05 to 1.0 in 0.05 increments, we “infect” the unknown wallets with the dominant class that crosses the relevant threshold tested. Inter-block label propagation is based on interactions between blocks; in our application we set a minimum of 5 edges between two different blocks to allow labels to propagate between them. Similarly, by applying different thresholds for propagation, different “neighboring” blocks can exchange label information and even dominate one or the other if the blocks are completely unlabeled. The combinations of these two label propagation strategies are intended to test if a more complete approach can lead to lower testing errors – intra-block propagation can lead to completely unlabeled blocks left untouched; these are then completed by the inter-block propagation, without losing the valuable intra-block heterogeneity captured by the SBMs.

Surrogate model performance

For evaluating our performance results against existing benchmarks, we train and test a suite of “surrogate” models commonly used in other published works, such as Random Forest, XGBoost, GCN, SAGE, MLP. For added credibility, we additionally train and test one of the champion benchmark models, a heterogenous GraphSAGE (Heter SAGE) which ranks third in reported performance by (Ferretti et al., 2024) and strictly for the Heter SAGE, we implement the same regimen as the authors did to obtain their benchmark, fixing learning rate to 0.001 and training for 1000 epochs with AdamW optimization. Due to computational and time constraints we were not able to train and test their other models. For each surrogate model in part, the dataset is divided using a temporal split between the first 34 steps (1-34) for training and the remaining 15 steps (35-49) for testing, which is aligned exactly with other experiments from literature.

To attribute performance differences to the additional information from thresholded propagation, we use a strictly inductive evaluation. The surrogate models are trained exclusively on the temporal training period and evaluated exclusively on the out-of-time test set. Label propagation is applied only to the training period and never performed on the hold-out test time steps. Therefore, no labels from the test period are used in any form (propagation, tuning or feature construction). At inference time we assume access to the current snapshot's unlabeled transaction graph and raw wallet features, the SBM partitioning and subsequent block features being computed using only this unlabeled information. For each test time step t , features are computed using wallets observed only at snapshot t (not using any snapshot $> t$). We do not make use of any edges, wallet attributes or summary statistics derived from time steps later than the prediction window. The block features we extract summarize interaction propensities (i.e., who transacts with whom and how) and are designed to be interpretable as *modi operandi* signals, e.g., block sizes, the most common block partner, average input/output flows of the blocks' members, and so on.

Graph construction, SBM inference, features, temporal train-test split, and surrogate model training budgets are held fixed. The only variability therefore comes from the propagation threshold, and the propagated training labels dependent on it. Hyperparameters were selected with the Optuna library for Python (Akiba et al., 2019) by optimizing the validation F1 score over 30 trials computed on an internal 70-30 training-validation split before retraining with the optimal hyperparameters on the full training set. To ensure compatibility with existing literature, the test metrics are computed strictly on the labeled subset of wallets as a pure binary classification task (licit vs illicit), while unknown wallets are preserved in the data to maintain the topological integrity of the network graphs but are excluded from all supervised training loss calculations and final reported test metrics. The test set remains therefore entirely unpropagated, and as the SBM block assignments fed as input to the surrogate models are inferred independently per time step, we mitigate forward-looking temporal data leakage by ensuring no future labels are accessed and no propagation is performed on the hold-out test set period.

As label propagation yields a slightly different training set for every scenario and threshold, a single scenario results in 20 training sets, meaning we obtain 81 ($4 \cdot 20 + 1$) sets of slightly different “best hyperparameters”, the reporting of which would be impractical; we therefore include in the annex one representative set of optimized hyperparameters.

Results and discussions

Objectives

In this paper we seek to reveal the underlying structure of illicit wallet behaviors within the Bitcoin network and provide insight into the formalization of *modi operandi* of illicit actor networks within cryptocurrency ecosystems, starting from the largest ecosystem of all. Our results start from validating that the methodology yields models that perform similarly well or better in the wallet test set performance, before going on to extend the coverage of machine learning in AML from cost-saving / efficiency gains to increasing effectiveness, leading to discovery and dismantling of wider, better camouflaged illicit networks within the Bitcoin ecosystem or other transaction spaces. Our stochastic block modeling approach therefore creates behavior and connectivity-based partitioning and assesses the value of this method in the profiling of wallets (actors) and propagating labels within and between blocks starting from a few known actors of the Bitcoin network.

Data Exploration

The dataset does not provide specific timestamps but rather 49 discrete snapshots of the Bitcoin network captured through the original Elliptic platform data structure. These are not representative of a particular period such as a week or month and are instead static captures of the network at a particular point in time, with successive time steps. This is not an exhaustive view of the Bitcoin network but is nonetheless a fairly expansive data capture at 1.27 million interactions from 822 thousand labeled wallet addresses (Elmougy & Liu, 2023). In Figure 1 we observe the class prevalences of the wallet and transaction datasets across time steps. As the focus of this paper is the wallet (addresses) section of the full dataset, any calculations about transaction classes are inferred from the wallet labels and not from the transaction labels. We note that for one, the positive rates across time steps for the wallets vary more than the inferred transaction network of these same wallet addresses, meaning that certain illicit wallets, however few they may be, are notably more active in the early and later periods of our time series. Conversely in the 20-to-40-time step range, while we observe a higher wallet positive rate, the illicit transaction rates remain relatively stable, suggesting a lower transaction rate in that snapshot range. Additionally, as at any particular snapshot more wallets will exist (due to historic activity), the number of wallets is much higher than the level of transactions captured at a particular snapshot in time.

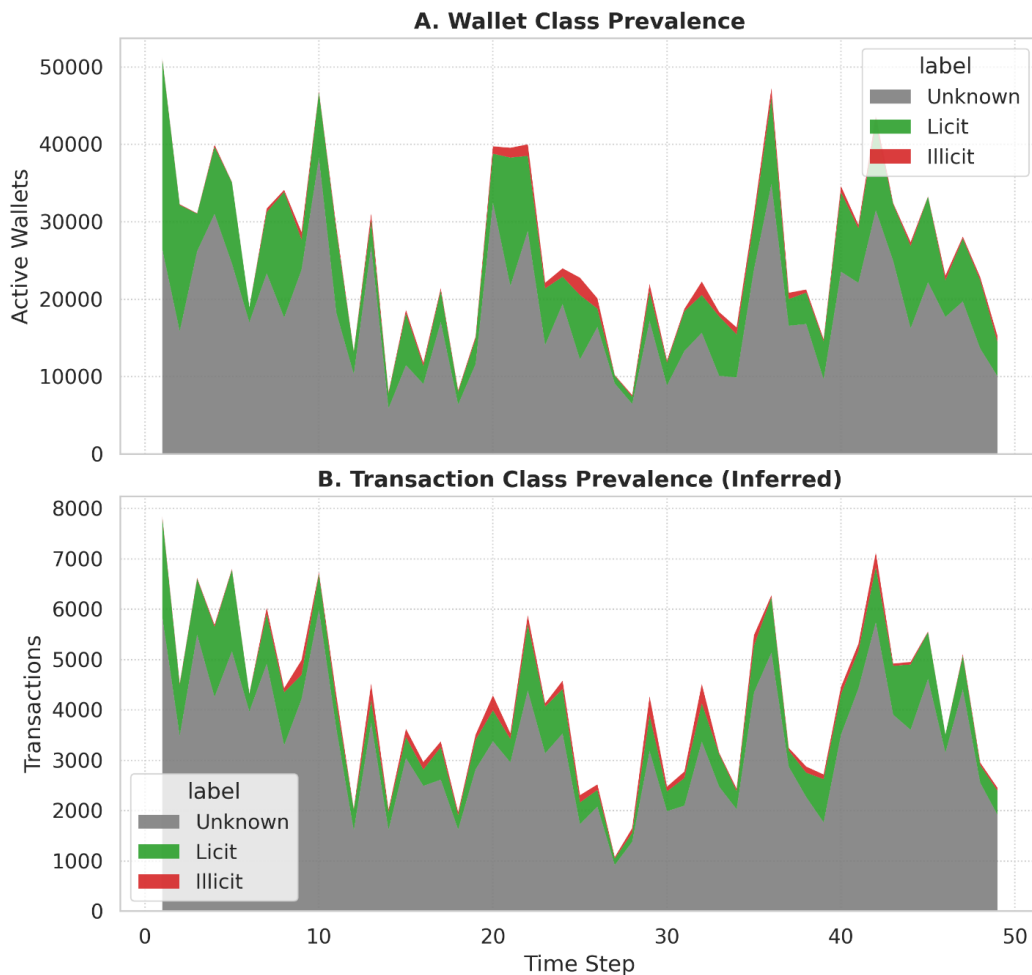


Figure 1. Wallet and Transaction class prevalences, by time step

Source: author's own elaboration using the Elliptic++ dataset.

The following empirical cumulative distribution function (ECDF) in Figure 2 is a step-function graph that visualizes the shortest distance path to the nearest illicit money flow for the classes in the dataset. The plot is grounded in the illicit flows, which sum up to 1 at hop 0 (origin). We note that we reach the 90% threshold only around the 6th hop, which aligns with the widely known and at times contested “small world experiment” by Milgram (1967), which shows that illicit sub-networks are not more sparsely or densely connected with the wider network than licit ones, which shows a high level of integration within the ecosystem. The fact that approximately half of all Bitcoin transactions are within two hops of an illicit one outlines the high level of risk Bitcoin users expose themselves to, willingly or otherwise.

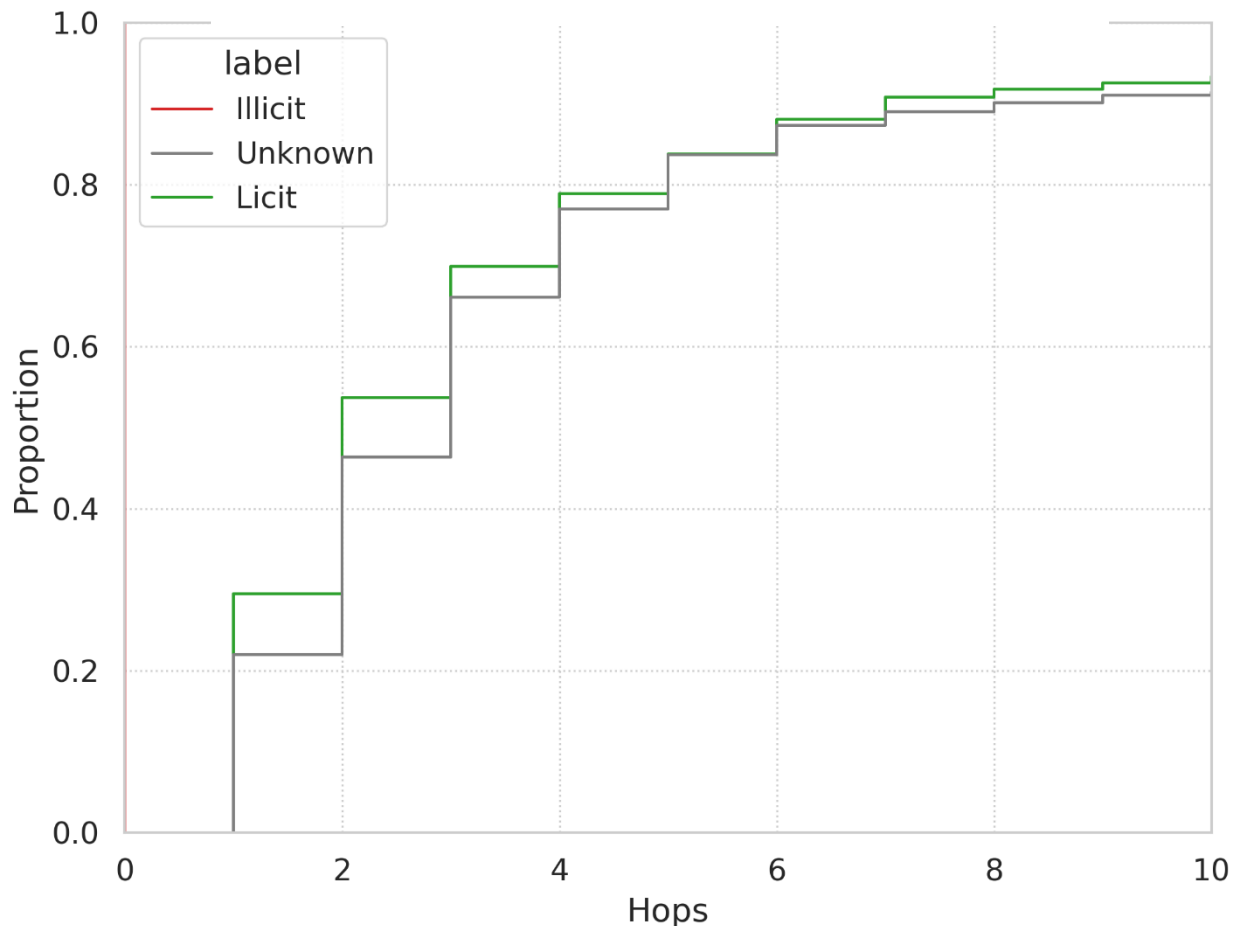


Figure 2. Empirical Cumulative Distribution Function (ECDF) of the proportion of hops/steps needed to reach the closest illicit wallet

Source: author's own elaboration using the Elliptic++ dataset.

Inference stability

The measurement of stability as described in our methodology can be explored in Figure 3. In this visualization we explore model stability through plotting the J-Score against the rate of illicit wallets in the blocks (non-propagated) as this visualization additionally reveals the unsupervised capability of SBMs to capture illicit wallets in the highly unbalanced Bitcoin network. We note that the y-axis starts at 0.6 for readability.

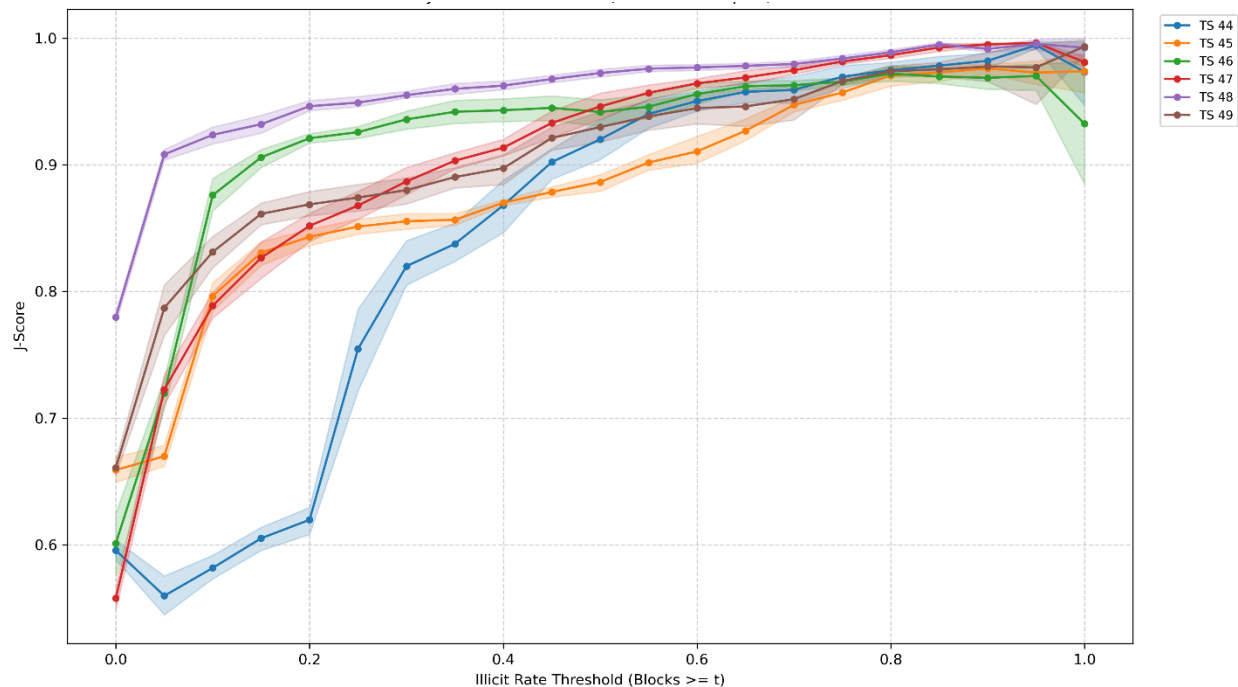


Figure 3. J-Scores averaged over 5 runs, TS 44 - 49, stabilized with ensemble consensus clustering (30 SBMs), plotted against intra-block illicit rates (pre-propagation)

Source: author's own elaboration and results using stochastic block modeling on the Elliptic++ dataset.

Starting from the left side of the chart, we note that the majority of partitioned snapshots perform well or very well even at low illicit rates, meaning that with the exception of TS 44 the model is highly stable even when it comes to blocks composed of just over 10% illicit wallets. If we are to consider a score of 0.8 stable, TS 44 stabilizes around the 30% mark, before ultimately joining the rest of the TS and reaching near complete stability at high illicit rates. This indicates that when the model partitions these highly illicit wallets together, it does so with high confidence, while the vastly outnumbering “Unknown” or “Licit” wallets introduce noise and cause some destabilization the lower the illicit rate is. On the other hand, even in the first quarter of illicit rates (0-25%) we note a J-Score of 0.6 to 0.75 which means that if unknown wallets are grouped with illicit wallets even at this level, they are suspicious at best, or the illicit wallets are highly “blended into the crowd” at worst. Regardless of the case, these results indicate high stability across illicit rates and provide valuable insight into the choice of the illicit rate to be used for propagation, depending on the risk appetite of the organization or individual that aims to implement our framework. The bootstrapped confidence intervals are widest for the most unstable snapshot, TS 44, followed by TS 49 toward the higher thresholds. We must note that these results are especially revealed given that, as shown in Figure 1, TS 44 through 49 contain some of the lowest illicit ratios across the dataset, making the SBM deal with a highly imbalanced, noise dataset.

Partitioning the entire network

We observe that per-snapshot SBM inference yields reproducible partitions (supported by ensemble-consensus stability analysis using the J-score) and produces blocks that separate dense nuclei and sparser, role-similar actors; these block assignments then serve as the sandbox for thresholded label propagation in the training period and downstream surrogate evaluation on the unpropagated labeled test set.

Illicit networks can be both sparse and dense at the same time. Figure 4 displays a wide-reaching (likely illicit) sub-network within time step 25 that comprises a dense core to the bottom of the plot, long-winding chains of wallets toward the edges, as well as more complex meshes slightly above the core of the plot. In such cases, subgraph visualizations are a powerful tool in understanding the underlying structure of the subnetwork.

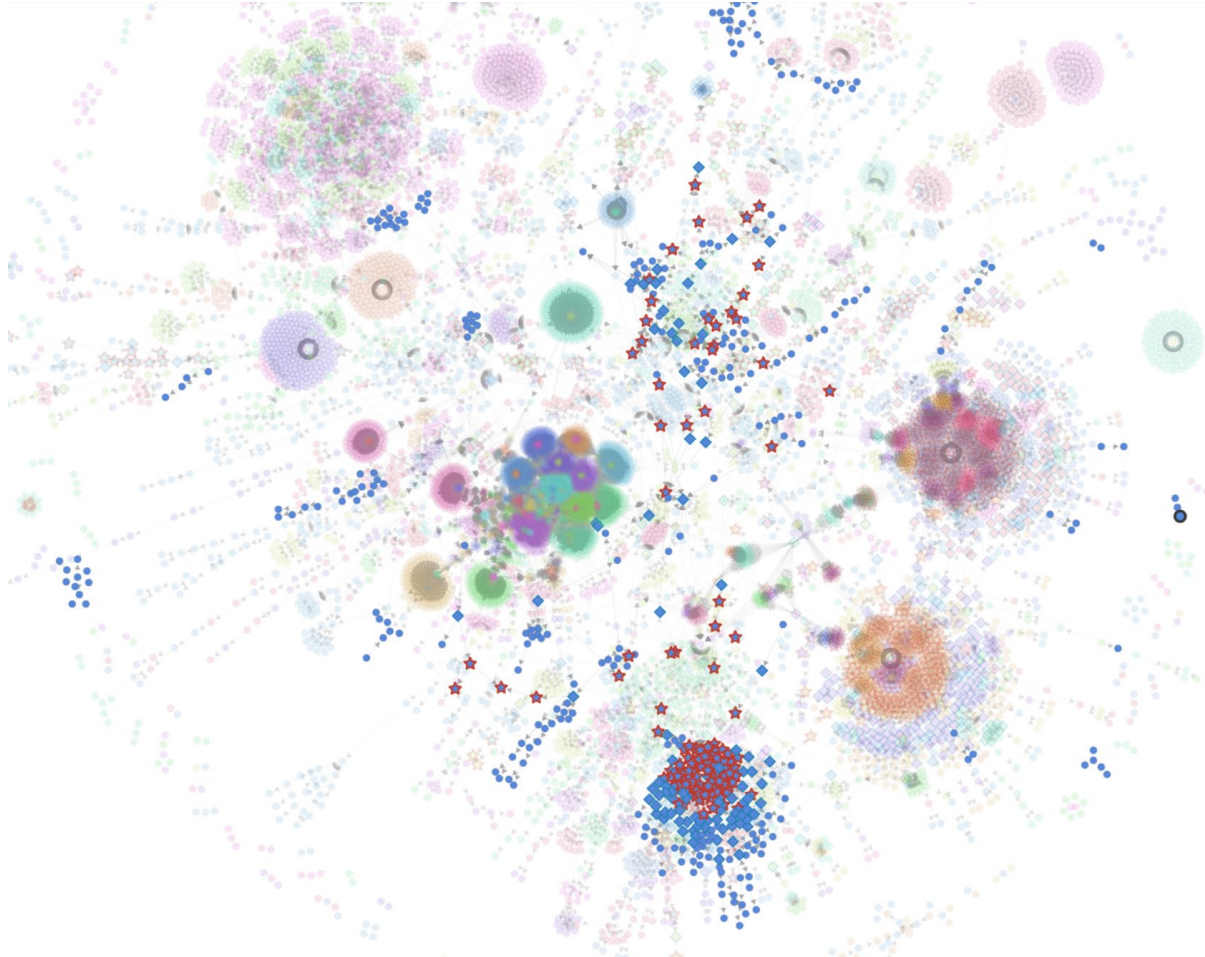


Figure 4. Selected view of a wide block in time step 25, drawn as stars for illicit wallets, diamonds for infected wallets

Source: author's own elaboration and results using stochastic block modeling on the Elliptic++ dataset.

Figure 5 displays the subgraph of the above-mentioned block, where we note a wallet at the root of the dense core, and some of the sparser chains that manage to connect directly to the wallets within the same block, in red. The gray nodes are nodes not marked as illicit that are grouped as part of this relatively high-risk block, based on the wallet behavior.

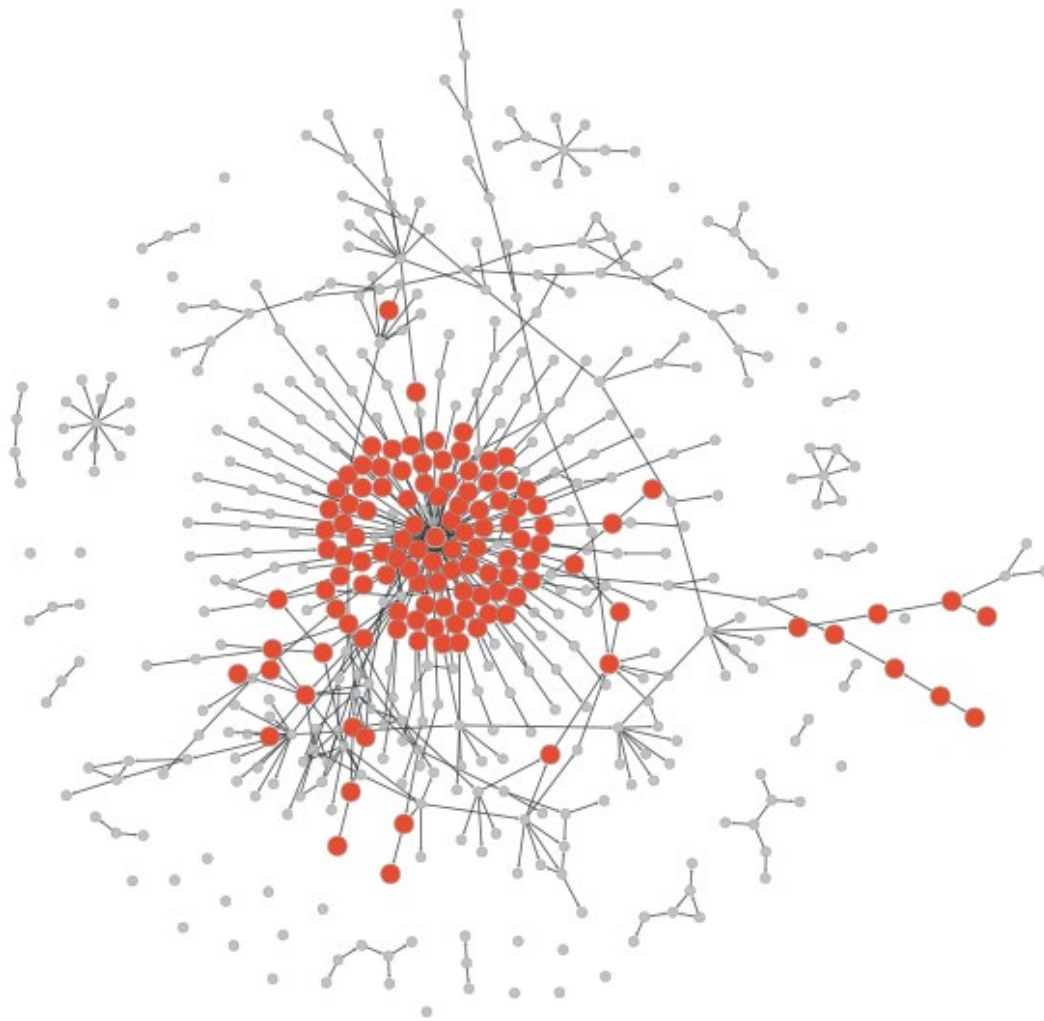


Figure 5. Subgraph of previously selected block in time step 25, illicit-labeled wallets in red

Source: author's own elaboration and results using stochastic block modeling on the Elliptic++ dataset.

Modi operandi in the Bitcoin network

Modi operandi are methods of committing financial crime and obfuscating the flow of money. In the transaction graph sense, modi operandi are the way in which multiple members of a network coordinate the transfer of funds between themselves, ultimately aiming to get the money out of the network after either fabricating a legitimate history of activity or complicating the investigation process enough that recovery is either impossible or at least more costly than accepting the loss. The Bitcoin network contains several known modi operandi by structural composition in the network, which we will highlight and combine with the block assignments.

The subnetwork displayed in Figure 6 is a textbook example of a “stacked” subnetwork where a party of multiple actors coordinate to create two or more layers of abstraction between the target wallet (bottom, marked as illicit) and the true source of funds. The money here is spread by the 7 diamonds to the outer ring of actors, made up of 8 unlabeled wallets and 1 illicit wallet. In this particular case, it is entirely possible that the unlabeled wallets are either all or in part illicit.

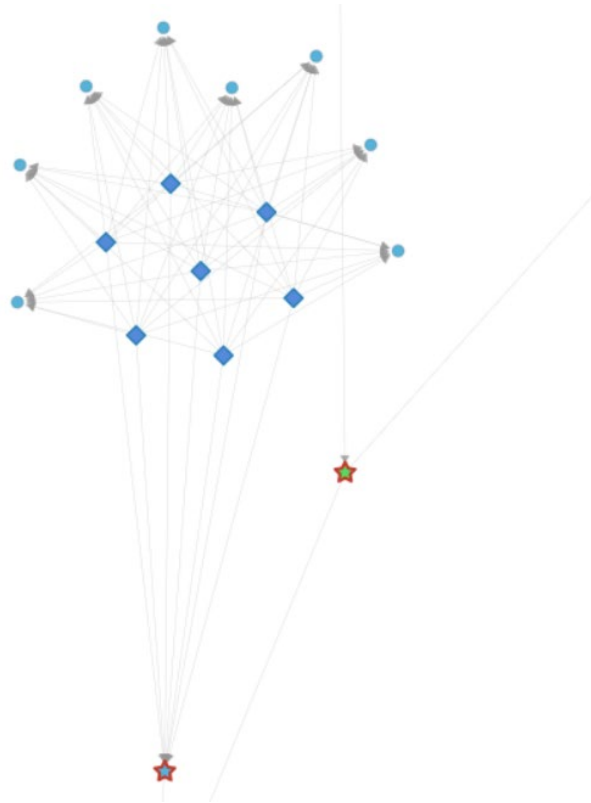


Figure 6. Close view of a stacked sub-network formed of 3 main blocks

Source: author's own elaboration based on stochastic block modeling results on the Elliptic++ dataset.

The same pattern, with small variations, is also found with other wallets and points in time. The stochastic block modeling performs the same partitioning each time, assigning the receivers (circles, star) together while the senders (diamonds) are grouped separately, to their own block. By following the recipients of funds transferred by the diamond block at any point, we may further discover and label illicit wallets.

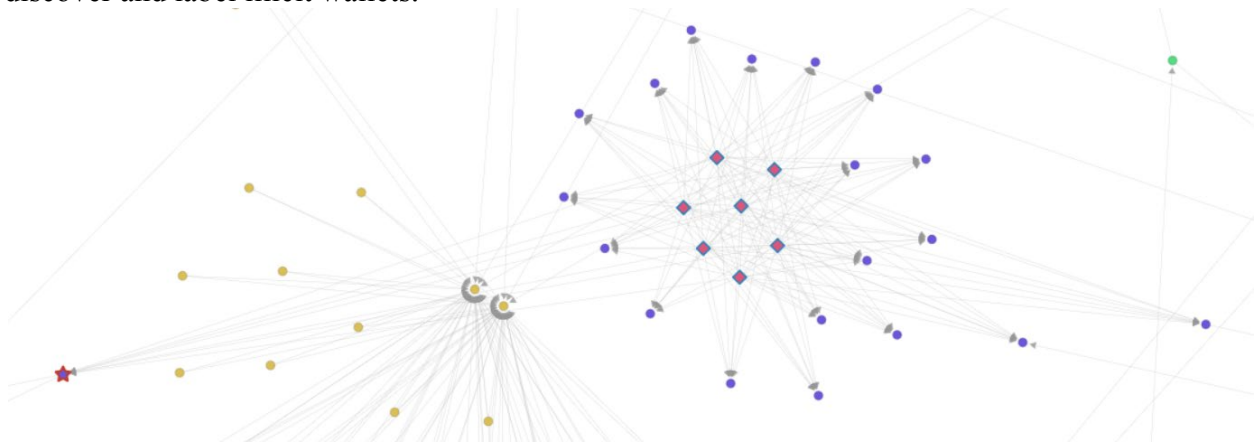


Figure 7. Close view of a second independent stacked sub-network formed of 3 main blocks

Source: author's own elaboration based on stochastic block modeling results on the Elliptic++ dataset.

Thresholded label propagation

Figure 8 displays the best achievable F1 Score based, across thresholds (5% to 100%) per each propagation scenario. The chosen threshold controls the minimum level of prevalence a class must have within the block or between blocks, allowing whichever label is higher to cross over and “infect” unknown label wallets. Incorporating inter-propagation (inter, intra+inter, inter+intra) achieve high and stable F1 across a broad low-to-mid threshold range, while intra propagation alone is less stable, providing only second-order information to the surrogate model(s).

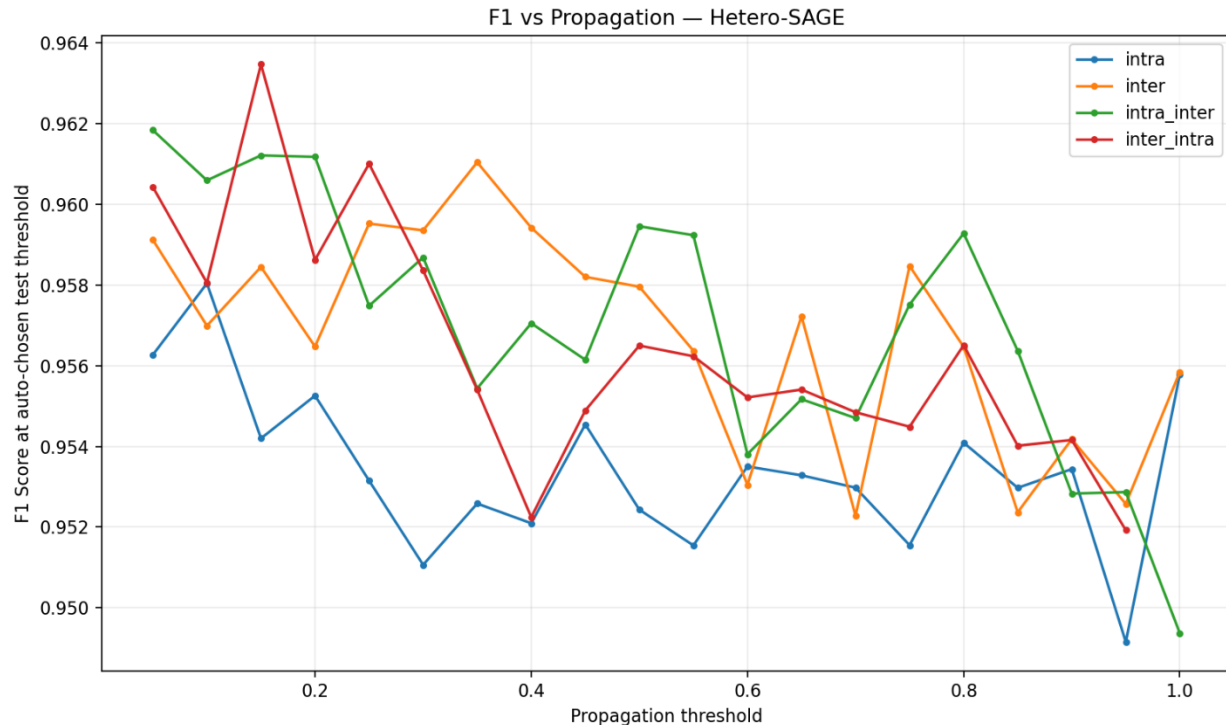


Figure 8. F1 Score per label propagation threshold, mean performance per propagation scenario, Hetero-SAGE, all wallets

Source: author's own elaboration based on trained (time steps 1-34) classification model results on test set (time steps 35-49).

Figure 9 displays the effect of the chained label propagation logic, propagating the dominant label exceeding 20% of the group prevalence throughout the block, as well as extending it to neighboring blocks when highly interconnected. In this case, the illicit class gets propagated throughout unknowns, while keeping known licit wallets untouched (note a small number of green diamonds).

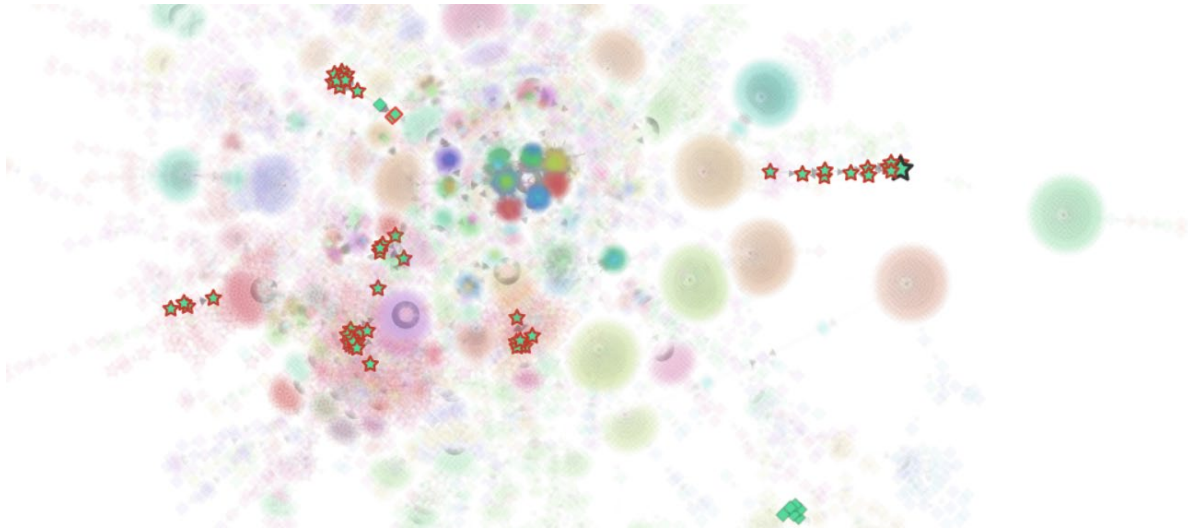


Figure 9. Snapshot view of label-propagated network at time step 49, intra+inter scenario, 0.2 label threshold

Source: author's own elaboration based on stochastic block modeling results on the Elliptic++ dataset.

In comparison, Figure 10 shows how this block looks when the threshold is set at 1.0, effectively disabling propagation, leading to many more observable diamonds (unknowns). Note that discrepancies in the rotation and setup of the nodes in the figures are expected, as they are re-computed and arranged independently, but the blocks, wallets, and edges are identical.

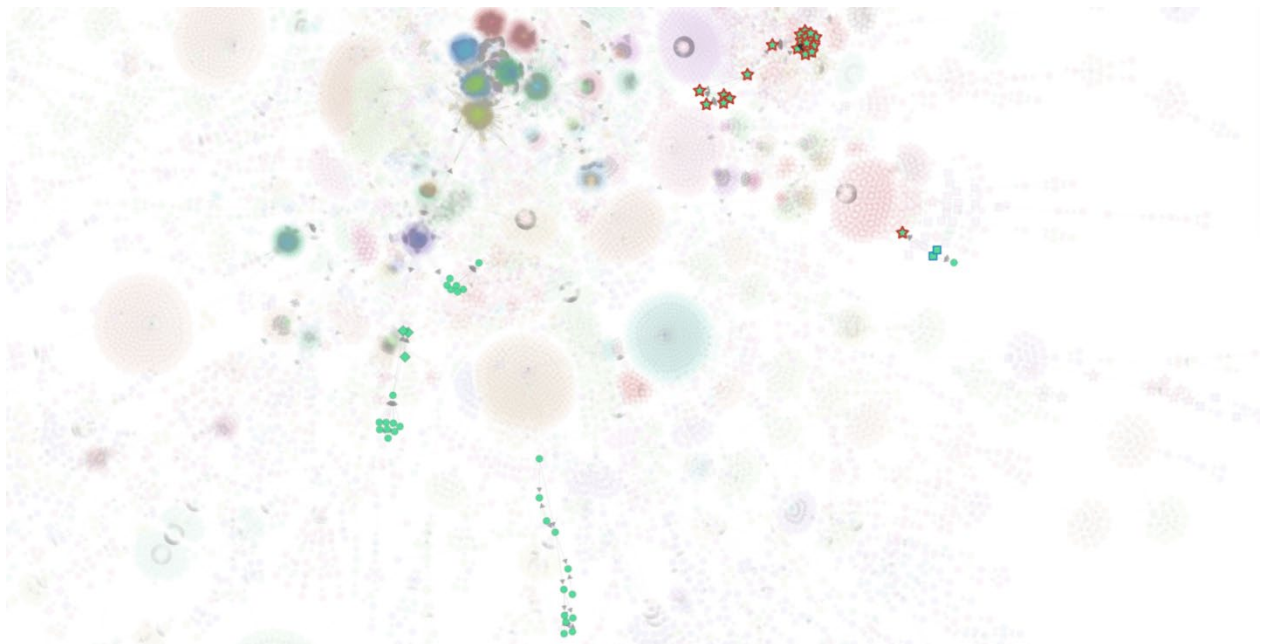


Figure 10. Snapshot view of label-propagated network at time step 49, intra+inter scenario, 1.0 label threshold (no propagation)

Source: author's own elaboration based on stochastic block modeling results on the Elliptic++ dataset.

Regarding cost-saving for cryptocurrency investigations, we leverage block label propagation by applying higher thresholds for extending the “licit” label in order to reduce risk of

mis-legitimizing unknown illicit wallets. As we see in Figure 11, the selected block is made up of approximately 80% licit wallets. While this is not a guarantee for legitimacy, we also do not have any illicit wallet present in this block, nor do any of the wallets have a direct connection to illicit wallets from another block.



Figure 11. Snapshot view of largely licit network at time step 49, intra+inter scenario, 1.0 label threshold (no propagation)

Source: author's own elaboration based on stochastic block modeling results on the Elliptic++ dataset.

Surrogate model testing

We benchmark the SBM block assignments using two surrogate model pipelines, a multi-model wallet/block surrogate with XGBoost, Random Forest, MLP, GCN and GraphSAGE, as well as a HeteroSAGE pipeline reflecting the modeling decisions taken by (Ferretti et al., 2024). Our methodology produces results that are competitive with the best reported wallet-level models in Table 3. While we do not see improvements over the champion benchmarks, our results are notable because top performers include XGBoost, MLP, Random Forest, meaning that the bulk of the information is captured by the stochastic block modeling such that even traditional architectures yield high performance, and testing more complex architectures such as the heterogenous attention network will likely yield considerably higher results. We note especially the considerable rise of Heter SAGE from its performance in Table 3 (F1: 0.922, F1 (micro): 0.860).

Table 4. Challenger benchmark results on the Elliptic++ dataset (labeled wallets)

Rank	Model	Precision	Recall	F1	F1-micro
1	Hetero-SAGE	0.935	0.977	0.956	0.917
2	XGBoost	0.922	0.995	0.957	0.915
3	GCN	0.914	0.976	0.944	0.894
4	MLP	0.922	0.995	0.957	0.886
5	GraphSage	0.916	0.990	0.952	0.867

Source: author's own results by applying the described methodology, averaged between thresholded runs' performance and rounded to 3 decimal places.

In the following we plot a precision-recall trade-off map for the heterogenous SAGE outputs, also known as a Pareto Frontier plot, which shows us the set of optimal scenarios to use

in case that we might favor recall over precision, vice-versa, or a balanced approach. We note that the intra-block scenario maximizes precision marginally, followed by inter and then the combined scenarios, which maximize recall. In particular, we note that almost every scenario/threshold improves over the base features slightly, proving the effectiveness of incorporating SBM partitioning to graph data sets for the binary classification task.

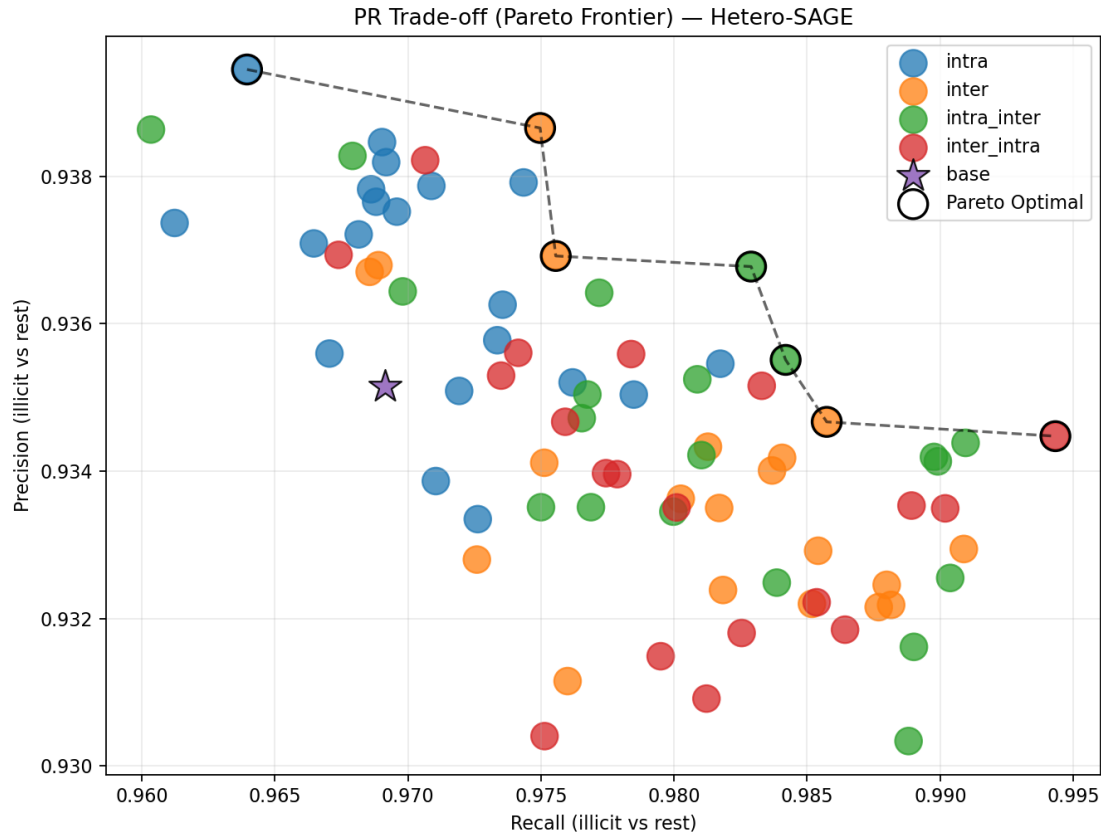


Figure 12. Precision-Recall Tradeoff Map (Pareto Frontier) for the Hetero-SAGE model

Source: author's own elaboration based on Hetero-SAGE classification results, as described by methodology.

In reference to the Pareto frontier above, the table below shows the best scenario/threshold pairs by F1-micro performance, where we see that depending on risk appetite, we can achieve even higher performance than is shown in Table 4.

Table 5. Granular challenger benchmark results on the Elliptic++ dataset (labeled wallets)

Rank	Model	Scenario	Threshold	Precision	Recall	F1	F1-micro
1	Hetero-SAGE	inter>intra	0.15	0.935	0.994	0.964	0.931
2	Hetero-SAGE	intra>inter	0.05	0.934	0.991	0.962	0.928
3	Hetero-SAGE	intra>inter	0.15	0.934	0.990	0.961	0.927
4	Hetero-SAGE	intra>inter	0.20	0.934	0.990	0.961	0.927
5	Hetero-SAGE	inter	0.35	0.933	0.991	0.961	0.927

Source: author's own results by applying the described methodology, rounded to 3 decimal places

Conclusion

In a real investigative process, finding a direct link between the two clusters would be very unlikely by using simple clustering algorithms or manual data exploration – by leveraging unsupervised methods such as SBMs for partitioning the Bitcoin network into blocks based on wallet behavior, the investigative process can automatically unveil separate sub-networks forming a whole, improving identifiability of wallets through peers, and by connecting multiple pieces of the same puzzle, ultimately dismantling such illicit networks and recovering lost funds faster and at lower cost by revealing these disconnected pieces.

In our study we successfully determine stochastic block modeling as an effective method for separating wallets into behavior groups (blocks), enabling automated discovery of wider sub-networks starting from a few labeled illicit wallets. The degree-corrected models, even without extensive tuning, perform on par with more advanced algorithms. More broadly, SBM implementations provide value where relationships are not obvious, for instance with networks that operate through separate pockets of actors or at various times in the network, switching parts of the illicit wallets on and off to avoid pattern detection through time. Our thresholding approach indicates the use of chained inter+intra or intra+inter label propagation as the most robust, with intra-block propagation being uniquely suited for maximizing precision. The general logic flow in Figure 13 describes the abstract process and can be used as a blueprint for reproducing our study on any network-based dataset such as other cryptocurrency data, standard payments or telecommunications, to name a few.

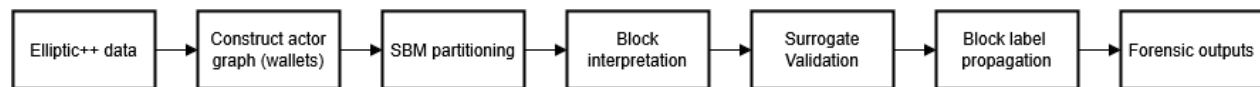


Figure 13. Logical flow of implementing, validating and contributing from SBM partitioning in a cryptocurrency network

Source: authors' own elaboration using draw.io.

A key limitation to such implementations is that while the models perform well in grouping together illicit actors and thus provide actionable blocks to target, this requires having at least a few known illicit wallets to start from, and the main contribution is with regard to significantly reducing the scope of required investigations and costs with recovering misappropriated funds or dismantling organized crime networks, through label propagation or block profiling. Moreover, a fair comparison between the SBM pipeline/experiment and non-propensity-based algorithms would require calibrating the block partitioning for optimizing on density, for instance, which adds a dimension of complexity to benchmarking the first stage model as described in the literature, with other partitioning algorithms.

Acknowledgments: The costs with the conference attendance and article publication are funded by the Bucharest University of Economic Studies.

References

- Ahmadinejad, N., & Liu, L. (2021). *J-Score: A robust measure of clustering accuracy* (arXiv:2109.01306). arXiv. <https://arxiv.org/abs/2109.01306>
- Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). *Optuna: A next-generation hyperparameter optimization framework*. arXiv. <https://arxiv.org/abs/1907.10902>
- Alves, L. G., Sigaki, H. Y., Perc, M., & Ribeiro, H. V. (2020). Collective dynamics of stock market efficiency. *Scientific Reports*, 10(1), 21992.

- Elmougy, Y., & Liu, L. (2023). Demystifying fraudulent transactions and illicit nodes in the Bitcoin network for financial forensics. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)* (pp. 3979–3990). Association for Computing Machinery. <https://doi.org/10.1145/3580305.3599803>
- Færch, C. J., Sen, P. S., & Yunqian, T. (2021). Information Flow and Social Organization in a Bitcoin Discussion Network on Twitter. *Proceedings of the Association for Information Science and Technology*, 58(1), 713-715.
- Ferretti, S., D'Angelo, G., & Ghini, V. (2024). On the use of heterogeneous graph neural networks for detecting malicious activities: A case study with cryptocurrencies. In *Proceedings of the 4th International Workshop on Open Challenges in Online Social Networks (OASIS '24)* (pp. 33–40). Association for Computing Machinery. <https://doi.org/10.1145/3677117.3685009>
- Funke, T., & Becker, T. (2019). Stochastic block models: A comparison of variants and inference methods. *PLOS ONE*, 14(4), e0215296. <https://doi.org/10.1371/journal.pone.0215296>
- Held, P., Krause, B., & Kruse, R. (2016). Dynamic clustering in social networks using Louvain and Infomap method (arXiv:1603.02413) [cs.SI]. arXiv. <https://arxiv.org/abs/1603.02413>
- Kang, H., Chang, X., Mišić, J., Mišić, V. B., Yao, Y., & Chen, Z. (2020). *Stochastic modeling approaches for analyzing blockchain: A survey*. arXiv. <https://arxiv.org/abs/2009.05945>
- Lee, C., & Wilkinson, D. J. (2019). A review of stochastic block models and extensions for graph clustering. *Applied Network Science*, 4, Article 122. <https://doi.org/10.1007/s41109-019-0232-2>
- Lo, W. W., Kulatilleke, G. K., Sarhan, M., Layeghy, S., & Portmann, M. (2022). *Inspection-L: Self-supervised GNN node embeddings for money laundering detection in Bitcoin* (arXiv:2203.10465). arXiv. <https://arxiv.org/abs/2203.10465>
- Milgram, S. (1967). The small world problem. *Psychology Today*, 1(1), 61-67.
- Peixoto, T. P. (2017). Nonparametric Bayesian inference of the microcanonical stochastic block model. *Physical Review E*, 95(1), 012317. <https://doi.org/10.1103/PhysRevE.95.012317>
- Peixoto, T. P. (2021). Revealing consensus and dissensus between network partitions. *Physical Review X*, 11(2), 021003. <https://doi.org/10.1103/PhysRevX.11.021003>
- Peixoto, T. P. (2023). *Descriptive vs. inferential community detection in networks: Pitfalls, myths and half-truths* (Elements in the Structure and Dynamics of Complex Networks). Cambridge University Press. <https://doi.org/10.1017/9781009118897>
- Qiu, L., Huang, Y., & Dong, G. (2026). Exploring crypto-stock risk contagion via directed complex network analytics. *Physica A: Statistical Mechanics and Its Applications*, 681, 131111. <https://doi.org/10.1016/j.physa.2025.131111>
- Takahashi, K., Yamamoto, K., Kuchiba, A., & Koyama, T. (2022). Confidence interval for micro-averaged F1 and macro-averaged F1 scores. *Applied Intelligence*, 52(5), 4961–4972. <https://doi.org/10.1007/s10489-021-02635-5>
- Vassallo, D., Vella, V., & Ellul, J. (2021). Application of gradient boosting algorithms for anti-money laundering in cryptocurrencies. *SN Computer Science*, 2(3), Article 143. <https://doi.org/10.1007/s42979-021-00558-z>
- Vu-Le, T.-A., Park, M., Chen, I., & Warnow, T. (2025). Using stochastic block models for community detection. *Applied Network Science*. Advance online publication. <https://doi.org/10.1007/s41109-025-00747-2>

Zheng, B., Zhu, L., Shen, M., Gao, F., Zhang, X., & Guizani, M. (2020). Identifying the vulnerabilities of bitcoin anonymous mechanism based on address clustering. *Science China Information Sciences*, 63, Article 132101. <https://doi.org/10.1007/s11432-019-9900-9>

Annex

Table 6. Selected best hyperparameters per surrogate model tested (threshold=0.2)

Model	Best Parameters
GCN	dropout=0.30571724727322863; epochs=117; h=256; l=3; lr=0.0075069210830028404; weight_decay=1.483181008313561e-05
Hetero-SAGE*	dropout=0.2844776724266701; h=128; l=3; weight_decay=1.4322754354742029e-05
MLP	dropout=0.2929559354096461; epochs=101; h=256; l=4; lr=0.0024167388720332046; weight_decay=7.051959661730181e-05
Random Forest	max_depth=20; min_samples_leaf=4; min_samples_split=20; n_estimators=95
GraphSage	dropout=0.153189296658792; epochs=160; h=256; l=4; lr=0.0006958006652474989; weight_decay=0.0029630454452071985
XGBoost	alpha=0.00011474764860769192; colsample_bytree=0.562951688976695; gamma=0.101494343132602; lambda=2.923609674456517; learning_rate=0.25339779313364075; max_depth=8; n_estimators=132; subsample=0.5972321837010757

Source: author's own results by applying the described methodology

*Hyperparameters fixed for epochs (1000) and learning rate (0.001)