NATIONAL SECURITY DATA AND POLICY INSTITUTE

PRODUCT NO. 00008

EFFICIENT DETECTION OF ILLICIT FINANCIAL ACTIVITY

MARCH 20, 2025

Efficient detection of illicit financial activity

Executive Summary

This study addresses the challenge of detecting illicit financial transactions on cryptocurrency exchange networks. Identifying activities such as money laundering and narcotics trafficking is difficult due to the anonymity of cryptocurrency networks and the strategic deception employed by illicit actors. The vast scale and complexity of financial transactions further complicate detection, necessitating advanced analytical tools. These tools, in turn, rely on data labeling, which is the process of annotating data with meaningful tags or categories to train machine learning models. The study offers early-stage research on practical approaches that minimize the cost and intelligence resources required for data labeling by integrating machine learning, active learning, and graph-based anomaly detection.

Cost-Sensitive Labeling Strategies

- Performance gains from additional labeling initially rise exponentially but plateau after about a third of the data is labeled. This indicates points of diminishing returns that can guide analytical investment.
- Findings indicate that identifying (labeling) only a small fraction ($< 4\%$) of illicit nodes is sufficient for supervised models to outperform unsupervised models.

Active Learning in Labeling Optimization

- The study explores the utility of active learning, a machine learning approach that prioritizes the most uncertain or valuable data points for expert labeling, in the context of illicit finance.
- The study demonstrates mechanisms for directing labeling that will most contribute to model performance based on observable node features.

Implications and Applications

- The results suggest that, while purely unsupervised methods are insufficient on their own, they can complement active learning techniques.
- The integration of active learning with cost-sensitive adjustments to labeling strategies can optimize the allocation of intelligence resources in cryptocurrency monitoring.
- These approaches point to scalable intelligence solutions that address the analytical burden posed by the increasing use of digital currencies in illicit activities and may generalize to other contexts.

Introduction

Identifying illicit transactions, such as those linked to narcotics trafficking and money laundering, within financial networks presents a significant intelligence challenge. Analysts must sift through vast amounts of transactional data, often with incomplete or ambiguous information, to detect patterns indicative of bad behavior by strategically deceptive actors. Addressing these challenges requires specialized collection and analytical expertise. Determining when and where to invest these finite resources is critical to optimizing intelligence capabilities.

Machine learning methods offer the potential to help combat the challenge of scale. Algorithms can automatically survey and categorize nodes at a speed humans cannot match, enabling analysts to focus on understanding the system rather than processing data. However, the utility of the results depends on the quality and quantity of labeled training data, which must be representative of the patterns and features of interest.

Labeling data is the process of taking a subset of data and identifying occurrences of the output that a model is trying to predict. For example, in cryptocurrency exchange data, this can mean coding actors and transactions as licit or illicit. In more routine applications where scale is the primary challenge, labeling is often simply a function of financial investment in human effort and computing power. In contrast, generating high-quality labeled data in intelligence and law enforcement contexts requires substantial analytic effort, domain expertise, and targeted collection. Without accurate labels, reliance on machine learning risks misidentifying illicit activity or failing to detect it altogether.

Intelligence analysts seeking to improve labeling in cryptocurrency exchange data face a particularly labor-intensive task. They must use multiple sources of information to triangulate profiles of anonymous licit and illicit actors. Doing so requires deep subject matter expertise, the painstaking assemblage of information, and the capacity to grapple with a nonstop flow of new data. Thus, optimizing the balance between automated detection and expert-driven analysis is essential to improve intelligence capabilities in financial monitoring.

This report offers early-stage work on potential solutions to the problem of optimal allocation of scarce intelligence resources in the context of illicit actors in financial networks. We focus on a Bitcoin network (released by [4]) to illustrate our results. Our work builds on a substantial body of research on the use of machine learning methods to identify illicit blockchain transactions and wallets, e.g., [4, 15, 1, 6, 13, 5, 8, 14, 16]. This literature consistently indicates that when sufficient information about illicit transactions or wallets is available, simple supervised methods using a large set of engineered features can adequately identify illicit activity, e.g. [4]. However, identifying the required labels, particularly in a context characterized by clandestine actors seeking anonymity, is challenging.

We investigate active learning as a strategy for labeling nodes for models that identify illicit actors in Bitcoin networks. This approach prioritizes the most informative nodes, enabling analysts to invest more efficiently and effectively. Work in this area is limited, see, e.g., [2, 3, 11, 7]. We develop a systematic understanding of the relative returns to investment in labeling incremental data and how those investments could be targeted for the best effect. To our knowledge, no previous research has addressed these questions in this context. We further explore strategies to enhance analytical capabilities, including the potential use of unsupervised methods as a warm start for active learning, their integration into subsequent iterations, and the incorporation of heterogeneous labeling costs.

Faster convergence of these methods will lead to lower overhead for the analyst.

While we focus on a Bitcoin network here, our findings generalize broadly. The supervised and unsupervised machine learning methods we use here rely on local and global network features, such as degree, properties of the ego network, and centrality scores. The unsupervised machine learning method we use is based on a Graph Neural Network architecture, which is based on the network structure. As such, the findings could be applied to any networked dataset.

Impediments to labeling

Manually identifying fraudulent or malicious actors in Bitcoin transactions requires analysts to grapple with vast amounts of pseudonymous data, taxing scarce time and resources. Because blockchain addresses eschew personal information by design, specialized investigators must rely on complex on-chain analytics and off-chain intelligence to build insight.

On-chain techniques–including address clustering, transaction graph analysis, and pattern recognition–demand substantial expertise and investment to generate even a partial understanding of who may be behind suspicious transactions. These techniques become more cumbersome as cybercriminals adopt sophisticated obfuscation methods such as mixers, peel chains, and chain-hopping, which demand time and computational resources to unravel.

The challenge is exacerbated by the need to merge on-chain analytics with off-chain intelligence, such as "Know Your Customer / Anti-Money Laundering" (KYC/AML) data from cryptocurrency exchanges, blacklists maintained by law enforcement agencies, and open-source information gleaned from data breaches or social networks. Data are siloed in different organizations, often across multiple jurisdictions, complicating collaboration. Analysts must continuously sift through and reconcile disparate datasets, requiring advanced tools and specialized expertise in data science, criminal investigation, and blockchain technology.

Ultimately, the fragmented and adaptive nature of illicit finance on the Bitcoin network underscores how resource-intensive it is for compliance and law enforcement teams to stay ahead of ever-evolving bad actors. Frequent updates to analytic methodologies and sustained interagency cooperation are necessary merely to keep pace. Given these challenges, any efficiency gains are valuable.

Quantifying the efficiency gains from additional information

Background

We represent the Bitcoin data as directed, temporal and weighted network $G_t = (V, E_t)$ with V denoting the set of nodes of the network and E_t denoting the set of edges at time stamp t; we assume the set of nodes is fixed, even if many of them do not have any incident edges at some times. For a node $v \in V$, let $N_{out}(v,t)$ and $N_{in}(v,t)$ denote the set of out-neighbors and in-neighbors of v, respectively; let $c_{out}(v,t) = |N_{out}(v,t)|$ and $c_{in}(v,t) = |N_{in}(v,t)|$. We drop the time dependence to simplify the notation. Most of our analysis uses the structure of the networks aggregated over the whole time period. The supervised machine learning methods we explore use simple features for each node v defined over $\bigcup_t N_{out}(v,t)$ and $\bigcup_t N_{in}(v,t)$. We use an unsupervised machine learning method that is based on Graph Neural Networks. Figure 1 gives an overview of the methods studied here.

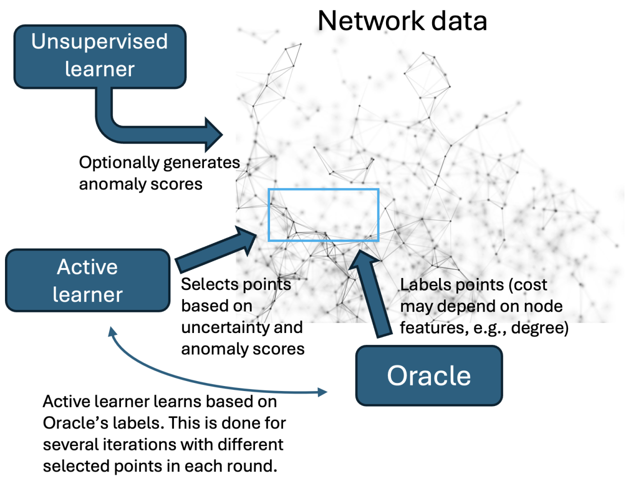

Network data
Unsupervised learner
Optionally generates anomaly scores
Active learner
Selects points based on uncertainty and anomaly scores
Labels points (cost may depend on node features, e.g., degree)
Oracle
Active learner learns based on Oracle's labels. This is done for several iterations with different selected points in each round.

Figure 1: An overview of the methods. An unsupervised method can provide optimal anomaly scores for the active learner, which then goes through multiple rounds of node selection for the oracle (human) to label, then updating its learned model using the cumulative set of nodes that have been labeled.

Assessing data labeling requirements

We begin by examining how the fraction of labeled data affect the performance of supervised and unsupervised methods. This analysis clarifies the amount and type of labeled data needed to gain an advantage over unsupervised anomaly detection approaches in cryptocurrency and other networks. We explore the following questions by evaluating the performance of machine learning models when:

- x_1 fraction of randomly chosen illicit nodes are correctly labeled;
- x_0 fraction of randomly chosen non-illicit nodes are correctly labeled;
- Nodes are ordered based on the ith feature value, and for k_i, the illicit nodes among the top k_i nodes with respect to the feature value are labeled correctly.

Active Learning to Improve Performance

We begin by exploring active learning methods to improve performance while reducing the amount of labeling required by experts. In this stylized description, the "learner" is the algorithm and the "oracle" is the human expert (intelligence analyst).

Suppose that we have a set of points $\mathcal{X} = \{X_i\}_{i=1}^{N}$. Each point has a corresponding true label Y_i, drawn from $\mathcal{Y} = \{-1, 1\}$, but these labels are initially unknown. The learner has a parameterized family of hypotheses $\mathcal{F} = \{f_\theta\}$, where $f_\theta(\cdot) : \mathcal{X} \to \mathcal{Y}$. The goal of the learner is to find a $\overline{\theta}$ that optimizes a loss function $l(f_\theta(X), Y)$. In the standard active learning setting, the learner sequentially chooses points from $\mathcal{X}$, for which the true labels can be revealed by an oracle. The active learner then has the additional goal of minimizing the number of queries to the oracle. A standard approach to this problem is known as uncertainty sampling [9], where the active learner has an "uncertainty function", $U : (\theta, X) \to [0, 1]$. The interpretation of the output of this function is that the closer the value is to 0, the more certain the learner is about the label of X. The closer the value is to 1, the less certain the learner is about the label. To choose the next point to query the oracle, the active learner samples from $\mathcal{X}$ by treating U as a probability distribution.

The active learning procedure is shown in Algorithm 1. The sets C and R represent the set of candidate nodes whose labels have not been revealed yet, and the set of nodes whose labels have been revealed, respectively. The function "Sample(D, S, N)" samples N nodes from the set S according to the distribution D over S. In the first iteration of active learning, a set of B random nodes are drawn from C (Line 5) and their labels are revealed (Line 11). Then, at each subsequent iteration, a model is trained using all the revealed samples (Line 7), and used to calculate uncertainty scores for all the candidate nodes (Line 8). Finally, a set of nodes is drawn from the candidate set according to the uncertainty scores (Line 9) and their labels are revealed (Line 11). Finally, after the last iteration of active learning, a model is trained using all the revealed node labels (Line 13).

Algorithm 1 Active learning strategy

```
Require: Input features of the training dataset $\mathcal{X}$ and their true labels $\mathcal{Y}$. Hypothesis family $\mathcal{F}$.
    Batch size $B$. Number of samples in the training set $n$.
 1: $C = \{i \in \mathbb{N} \mid i \leq n\}$
 2: $R = \emptyset$
 3: for $t$ from 1 to $T$ do
 4:     if $t = 1$ then
 5:         $R = \text{Sample}(\text{Uniform}, C, B)$
 6:     else
 7:         $\theta \leftarrow \arg\min_\theta \sum_{i \in R} l(f_\theta(x_i), y_i)$
 8:         $U \leftarrow \text{UncertaintyFunction}(f_\theta, \{x_i \in \mathcal{X} \mid i \in C\})$
 9:         $R = R \cup \text{Sample}(U, C, B)$
10:     end if
11:     $C = C - R$
12: end for
13: $\overline{\theta} \leftarrow \arg\min_\theta \sum_{i \in R} l(f_\theta(x_i), y_i)$
14: return $f_{\overline{\theta}}$
```

The total number of points queried by the active learning method is $B \cdot T$. The costs of queries can vary with these parameters; the number of iterations T can have a bigger impact on the total cost.

Strategies for reducing labeling cost. During active learning, responses to queries are expensive to obtain because they need expert analysts. Here, we examine whether anomaly scores from unsupervised ML methods could help in reducing the number of labels needed. We experiment with a GNN-based unsupervised anomaly detection process (described later), which gives us an anomaly score, $S_i \in [0, \infty)$ for each X_i. We assume that the anomaly scores are correlated with the probability that the label for that point is 1. We convert these anomaly scores into a prior probability distribution over the labels as

$$P_0(Y_i = 1) = \frac{S_i - S_{min}}{S_{max} - S_{min}}, \tag{1}$$

where $S_{max} = \max_i S_i$ and $S_{min} = \min_i S_i$. We assign an initial labeling to all points by sampling from P_0.

We proceed as follows. At time t, we train the active learner on the labeling at time $t-1$. We then let it choose a set $\mathcal{Q}_t$ from $\mathcal{X} \setminus \cup_{j=0}^{t-1} \mathcal{Q}_j$ (using uncertainty sampling) to query the oracle. The oracle provides the true labels for $\mathcal{Q}_t$, which replace the labels from time step $t-1$ for those points, and the process repeats.

Integrating labeling cost into active learning. So far, we have assumed a uniform labeling cost model, i.e., labeling a sample requires the same amount of work, or cost, from the oracle, regardless of its properties. This is clearly not true, since labeling requires a variety of expensive expertise, groundwork, and investigations. We assume a cost c_v for acquiring a model for a node v, to make the active learning process cost sensitive. This approach has been explored to a limited extent for special cost functions, e.g., [18], but not in the context of illicit finance. Efficient active learning methods for general cost functions remain open. Here, we study a heuristic that modifies the sampling based on uncertainty score (Line 9). Specifically, we scale the uncertainty of every node v by $1/c_v$, and then run the sampling step to select the B nodes with the highest scaled uncertainty scores to be queried by the oracle. In future work, we will explore more sophisticated strategies for selecting nodes to query with a bound on the total cost.

Anomaly detection using Graph Neural Networks (GNNs)

We employ Graph Neural Networks (GNNs) as an unsupervised method to detect anomalies (see Figure 1). GNNs take the graph structure and node features as input. For the initial analysis, we rely on subsets of the node centralities. We use CoLA [10] because of its high performance and scalability to large networks. CoLA is a self-supervised method that attempts to learn neighborhood patterns of nodes in a manner that incorporates node features and uses this information to assign anomaly scores to nodes. A node is ranked as highly anomalous if the way it fits within its neighborhood does not align with other nodes in the graph (i.e., is out of distribution.) Its operation is explained in Figure 2. In subsequent analyses we employ additional node features; for many tasks, additional features are known to increase the power of GNNs.

We use a Graph Autoencoder (GAE) for anomaly detection. A GAE encodes nodes into a latent representation and then attempts to reconstruct the original node attributes. The anomaly score of a node becomes attribute vector's reconstruction error.

Since both the training and inference steps only require sampled subgraphs, the memory cost of this approach is extremely low. This contrasts with auto-encoder methods that require reconstructing entire graphs.

Data and Evaluation

Dataset. We use the labeled Bitcoin network datasets (Elliptic++) released in 2023 by [4]. The dataset is obtained by crawling the Blockchain, building on an earlier dataset released by [17]. The data is comprised of two networks: transactions and wallets.

1. *Transaction* network: This temporal network conceptualizes financial transactions as a network of transaction nodes connected by edges capturing value flows. The network has 203K nodes, which represent individual transactions. Edges in this network represent payment flows between them. Each node is associated with 183 features, which include local information about the transactions and metadata derived from transaction attributes. The nodes

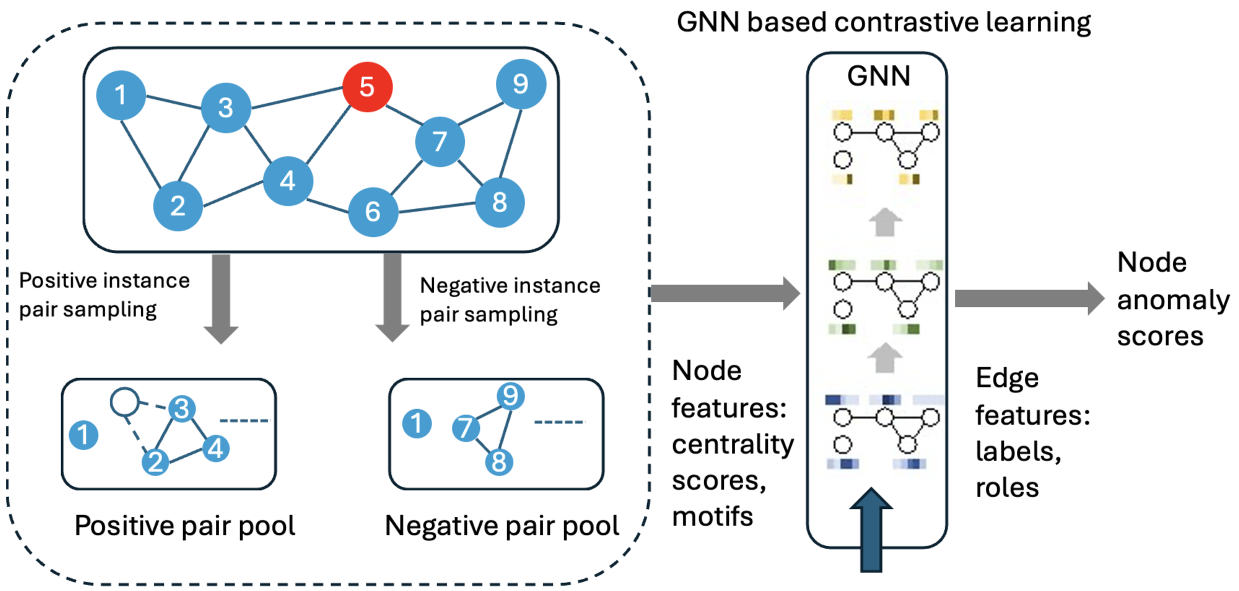

GNN based contrastive learning
GNN
1
2
3
4
5
6
7
8
9
Positive instance pair sampling
Negative instance pair sampling
Positive pair pool
Negative pair pool
Node features: centrality scores, motifs
Edge features: labels, roles
Node anomaly scores

Figure 2: GNN based anomaly detection, building on CoLA [10]. This is a three-stage pipeline: sampling, a trainable GNN, and a ranking. **Sampling:** First, for each node, referred to as a target node, a positive subgraph, and a negative subgraph are sampled. The positive subgraph is sampled using a Random Walk with Restart (RWR) starting at the target node, and the negative subgraph is sampled by doing an RWR beginning at a different node from the target node. **GNN:** Both sampled subgraphs are fed into a GNN and aggregated into a single embedding vector each. A "fitness" score of a target node with its subgraph is calculated using a single discriminator layer, a bilinear trainable function. The model is trained using an objective function that maximizes the fitness score of the target node in its positive sample and minimizes the fitness score of the target node in its negative sample. **Ranking:** Finally, to arrive at an anomaly score for a target node, the premise is that a regular, non-anomalous node will fit much better with its positive-sample subgraph than its negative-sampled subgraph, while an anomalous node will have "similar" fitting in its positive and negative subgraphs. Based on that, they calculate the anomaly ranking of a node as the difference between the fitness score of a target node with its negative sample, and the target node with its positive sample. The lower this value is, the less likely it is that a node is anomalous. They average this difference over many positive-negative sample pairs. Crucially, diverse kinds of additional node features are used; for many tasks, such additional features are known to increase the power of GNNs.

(transactions) are labeled illicit (about 2%), non-illicit (about 21%), or unknown using prior information on illicit transactions.

2. *Wallet* network: This consists of 822K nodes, where nodes are wallets and edges indicate that a transaction pulled money from a wallet and then deposited into another wallet. The edges in this network correspond to the nodes in the transaction network. Each wallet node is associated with 56 features, which are properties of the associated transactions. Wallets are also labeled illicit, non-illicit, or unknown.

   The active learning algorithm applied to the wallet network uses features to classify action as likely illicit or likely non-illicit. At the wallet level, these classifications are associated with their respective edges, and are therefore derived from the transaction classifications. Wallet nodes are labeled illicit if associated with at least one illicit transaction and non-illicit if the fraction of non-illicit transactions associated with it is larger than a threshold.

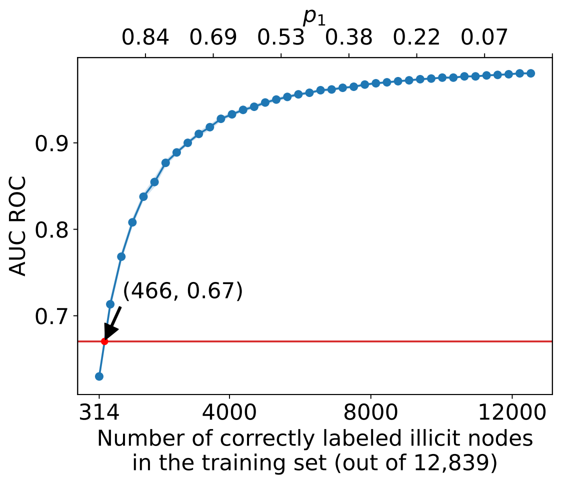
An additional user-user network is constructed after clustering. Our initial analysis here is on the wallet network.

Machine learning methods. We consider both supervised and unsupervised methods. Supervised methods approach the data in a tabular form. These methods include logistic regression, random forests, and SVMs. Every entity is represented using a vector of features; the network structure is not considered in the training. Note, however, that a node's features can result from a network aggregation operation (e.g., centrality calculation). The unsupervised methods take all the data as input (both the training and test sets), but none of the samples have any labels (anomaly/non-anomaly). We use message passing based GNNs, including graph auto-encoders and CoLA [10]. These methods produce, for each element in the test set, an anomaly score value $\in [0, 1]$ where the higher the value, the more likely an element is to be an anomaly. The ranking of nodes is based on node features, such as eigenvector centrality. A higher ranking on the target feature represents a higher anomaly score and, hence, a higher likelihood that the node is anomalous. Choosing the appropriate threshold at which elements are deemed anomalous typically requires external knowledge about the expected number (or fraction) of anomalous data. Assuming that k nodes in the test sets are anomalous, the top$-k$ elements in the anomaly score are chosen and classified as anomalous.

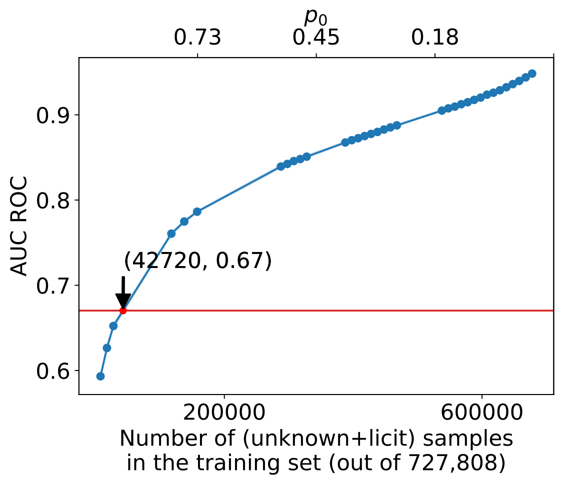
Evaluation. We evaluate all of our methods (supervised and unsupervised) on 10% of the data, using ten different random subsets of the data. Our evaluation metric is the Area Under the ROC curve (AUC ROC). It measures the area under the True Positive Rate versus False Positive Rate curve. A score of p is interpreted as the probability that an anomalous node will have a higher anomaly score than a non-anomalous node.

Results

Label value with full information. We first study the effect of the number of correctly labeled illicit nodes and non-illicit nodes (chosen uniformly at random from all known licit or illicit nodes) on the performance of a supervised anomaly detection model. Figure 3a shows the performance of the supervised model as the number of illicit nodes labeled increases. In other words, the leftmost point on the x-axis corresponds to only 314 out of the 12,839 illicit nodes having an "illicit" label. All other nodes in the graph are considered "non-illicit", corresponding to the combination of illicit



(a) The x-axes show the number of correctly labeled illicit nodes (bottom) and the corresponding flipping probability p_1 (top).



(b) The x-axes show the number of correctly labeled non-illicit nodes (bottom) and the corresponding flipping probability p_0 (top).

Figure 3: For all data points, the same train/test data split was used. At each point, we train a model with a perturbed training set in which x samples are flipped. We then use the trained model to rank the test set and calculate the resulting AUC ROC score. Each experiment is repeated 20 times with random perturbation. The red horizontal lines indicate the performance of an unsupervised model.

nodes and those without a label. The red horizontal line shows the performance of an unsupervised model that does not consume any label information. Note that the unsupervised model was tuned for the node features used. Note that the train/test split is the same for all the points, but each point is the average of 20 replicates, each randomly flipping the labels of a different set of k nodes.

Figure 3b shows the opposite process; all the nodes start with the illicit label, and then as the x-value increases more nodes are correctly labeled as "non-illicit."

In both instances, supervised methods outperform unsupervised ones after correctly labeling a small percentage of nodes (466 for illicit nodes and 42,720 for licit). Similarly, both demonstrate rapid returns to scale for approximately the first third of the data, leveling out thereafter. However, while these results clarify the context of the problem, they are of limited use for economizing actual analysis because they indicate the value of each newly labeled node chosen randomly but do not direct the analysis toward specific nodes of high value.

Label importance: ordering based on different feature values. Next, instead of randomly choosing correctly labeled illicit or non-illicit nodes, we select nodes based on feature values. This corresponds to settings where analysts could obtain information on correct labels based on easily observable intelligence; such orderings are commonly considered when evaluating feature importance using methods such as SHAP [12].

In this setting (shown in Figure 4), data points are considered in sorted order of a particular feature value; for any k, let S_k denote the set of points consisting of the top k values associated with a feature. For each k, the labels of illicit nodes within the set S_k are provided correctly; the labels of illicit nodes not in S_k are not provided. For example, for the blue line in Figure 4, the leftmost

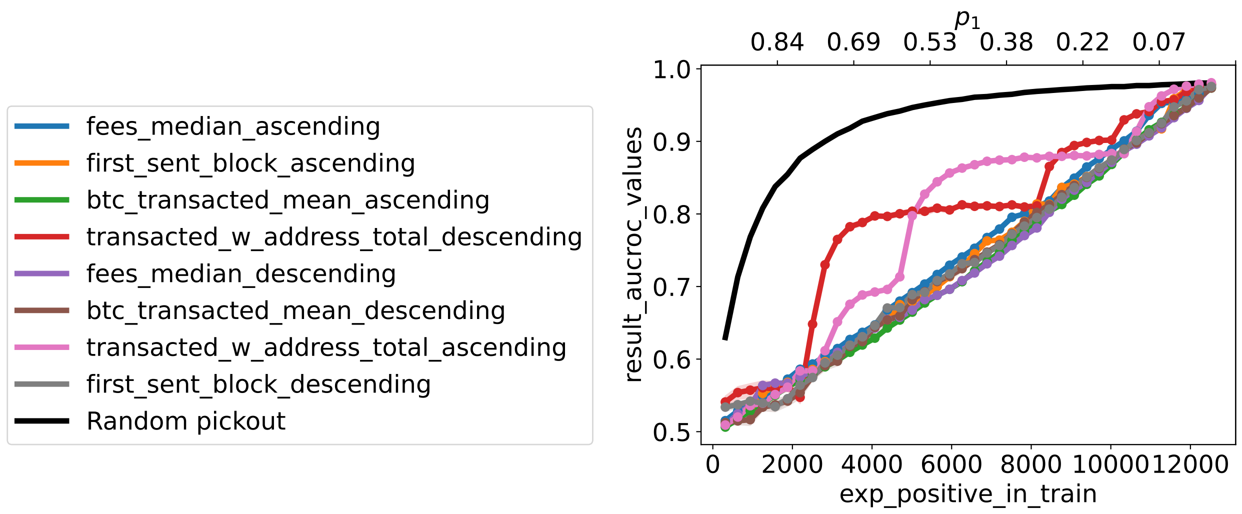
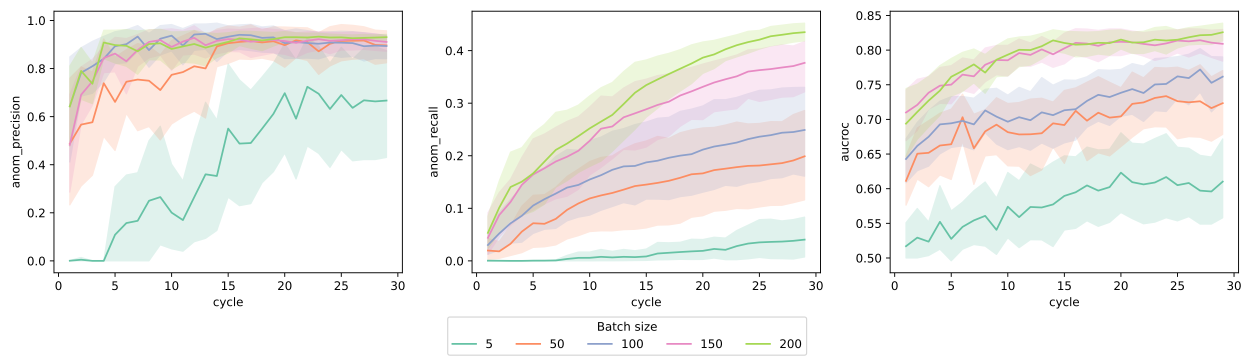


Figure 4: Evaluating label importance using ordering based on different feature values: Data points are considered in sorted order of a particular feature value. For any k and any feature (shown as different lines in the plot), the labels of illicit nodes within the set S_k consisting of the data points with the top k values associated with a feature are made available, and the corresponding line shows the AUCROC for that ordering.

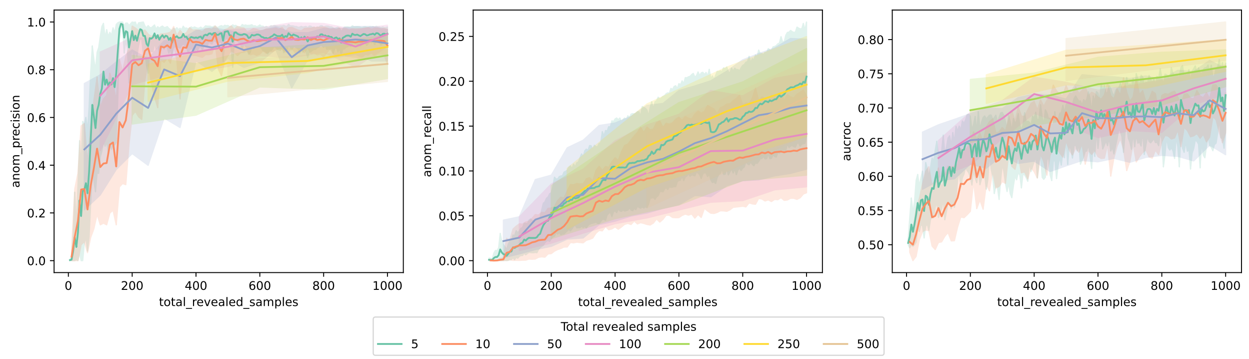
point is the performance of a model that is trained on data in which the only illicit nodes are those among the set S_{314} of points with the highest values of the node feature `fees_median`. The next point on the line shows model performance after the next highest 314 illicit nodes on that feature. Similarly, the red and pink lines in Figure 4 correspond to data points sorted based on the features `transacted_w_address_total` and `transacted_w_address`, respectively. The specific shape of these lines arises from the fact that the number of illicit nodes in the set S_k varies in a monotone step-wise manner, which contributes to a similar behavior in the AUCROC values. The results in Figure 4 imply that picking points to query labels along individual feature values does not lead to good performance. A natural question is whether a combination of features could help; our initial analysis suggests that simple ways of combining (e.g., an average after normalization) does not help, and more sophisticated methods must be explored.

Active learning. We first show the benefit of the active learning method (Algorithm 1). For all of our experiments, the data is split into a 90/10 train-test split. Each experiment is repeated ten times with a different train-test split. The test data is not available to the model at training time. A single experiment proceeds over T time steps, where each time step corresponds to the active learning model proposing B samples for labeling and an oracle (i.e., expert) assigning correct labels to these samples.

Figure 5 shows the precision, recall, and AUC ROC scores of models trained using different batch sizes. Figure 5a shows the metrics with respect to active learning cycles; each cycle represents an additional round of the oracle revealing the labels of B additional samples, and the model being trained by all the revealed datapoints up to this cycle. Figure 5b shows the performance metrics compared against the total number of revealed samples to a model, irrespective of cycle. The recall is low in these plots because the active learning method was not run long enough; in separate analysis, we observe that running active learning for more iterations (until about 10% of the data



(a) The $x-$axis is the cycle of the training process of the model. In other words, cycle x indicates that the oracle revealed x batches so far and the model trained on all $x \cdot B$ samples.



(b) The $x-$axis is the total number of samples that the model is trained on. All models are left to iterate for as many cycles as needed to reach these counts. In other words, the model with $B = 5$ runs a total of 500 time cycles, while the one with $B = 500$ runs for two time cycles only.

Figure 5: Performance of active learning model using different batch sizes. Each line represents ten replicates with ten different train/test splits. The highlighted area represents the 95% confidence interval.

is seen) ensures a performance similar to that of the supervised learning method. *Our results show that the larger the number of samples that is revealed per step, the better the model. This suggests a potential performance gain from carefully staggering the expert work across time-steps.*

Active learning augmented with unsupervised ML methods. Next, we try to incorporate the GNN-based unsupervised learning model's information into the active learning procedure. With this experiment, rather than the first iteration of active learning picking a random set of B samples to label, we run an unsupervised learning method on the data to produce an anomaly score for each sample. Then, we label the top τ samples in terms of anomaly score as anomalous. We use these τ labeled samples as the initial data set from which the model trains to choose its first B samples for which it will query the oracle. Note that the labels of the initially chosen nodes (according to the unsupervised model) may not be correct, making the model train with potentially incorrect labels. Our results show that this approach fails to improve the model, at least with this method of incorporating anomaly scores into the active learning phase. This makes sense as the unsupervised anomaly detection does not successfully label anomalies. Figure 6 shows the model's performance on the final training cycle using different batch sizes for different τ values. Performance only deteriorates as τ increases. We note that this lack of utility holds even for the best-performing

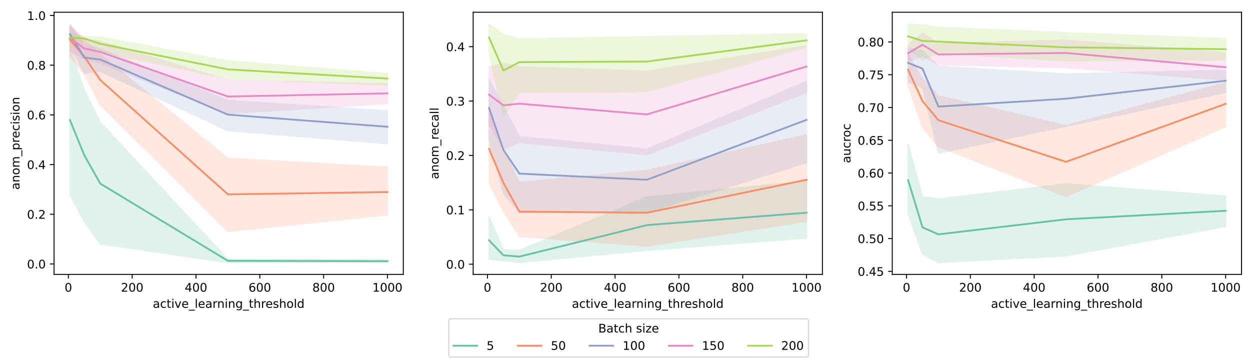


Figure 6: Performance using an initial labeling strategy in which labels are given based on an unsupervised learning method to a subset of samples.

threshold value, $\tau = 5$.

Integrating labeling cost into active learning. We now consider the setting where the cost c_v of labeling a node v is not uniform, but depends on its properties. Here, we assume $c_v \propto d(v)$, where $d(v)$ is the degree of node v. As described earlier, we modify the Sample step (Line 9) in Algorithm 1 in the following manner: We scale the uncertainty score of every node by the inverse of the degree of the node and select the B nodes with the highest scaled uncertainty scores to be queried by the oracle. Figure 7 shows the results of applying this approach with batch size $B = 5$ for 60 cycles. Scaling uncertainties in this manner has a clear impact on performance. On average, the total cost of labeling the samples without using cost scaling was 2882, and with using cost scaling it was 694.

Implications

In Bitcoin networks and other intelligence environments where labeling is costly and relies on specialist knowledge, unsupervised methods can be attractive because they do not require any labeled examples, thereby sidestepping the steep “upfront” cost of data annotation. However, purely unsupervised approaches often produce limited gains when the underlying patterns of illicit behavior are nuanced or sparse, and they may generate more false positives in complex financial networks. Active learning offers a more targeted strategy: Rather than attempting to label a random subset of data—or none at all—it is possible to identify the most uncertain or potentially informative instances for expert review. By focusing the scarce labeling budget on data points likely to refine the model’s decision boundary, active learning can achieve significantly better accuracy than unsupervised methods with far fewer annotated samples. This is particularly valuable in cryptocurrency and other financial networks, where selectively labeling a small fraction of highly suspicious nodes often suffices to outperform a label-free anomaly detector, all while minimizing analytic overhead. The cost of labeling nodes is typically highly non-uniform, and adapting active learning to incorporate these costs can provide significant benefits.

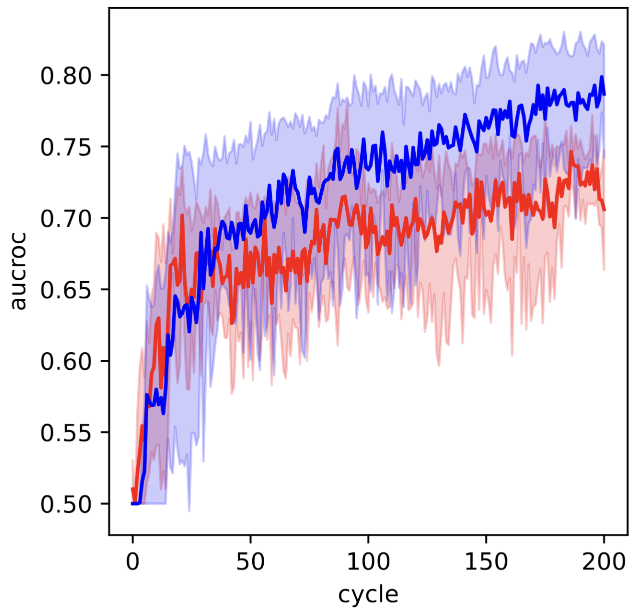


Figure 7: The performance of the model when uncertainty scores are not scaled by cost (red) and when they are scaled by cost (blue) using batch size $B = 5$.

References

[1] Massimo Bartoletti, Barbara Pes, and Sergio Serusi. Data mining for detecting bitcoin ponzi schemes. In *2018 crypto valley conference on blockchain technology (CVCBT)*, pages 75–84. IEEE, 2018.

[2] Leandro L Cunha, Miguel A Brito, Domingos F Oliveira, and Ana P Martins. Active learning in the detection of anomalies in cryptocurrency transactions. *Machine Learning and Knowledge Extraction*, 5(4):1717–1745, 2023.

[3] Leandro Lopes Cunha. *Anomaly detection in cryptocurrency transactions with active learning.* PhD thesis, 2024.

[4] Youssef Elmougy and Ling Liu. 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*, pages 3979–3990, 2023.

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

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

[7] Danilo Labanca, Luca Primerano, Marcus Markland-Montgomery, Mario Polino, Michele Carminati, and Stefano Zanero. Amaretto: An active learning framework for money laundering detection. *IEEE Access*, 10:41720–41739, 2022.

[8] Chaehyeon Lee, Sajan Maharjan, Kyungchan Ko, and James Won-Ki Hong. Toward detecting illegal transactions on bitcoin using machine-learning methods. In *Blockchain and Trustworthy Systems: First International Conference, BlockSys 2019, Guangzhou, China, December 7–8, 2019, Proceedings 1*, pages 520–533. Springer, 2020.

[9] Shang Liu and Xiaocheng Li. Understanding uncertainty sampling. *arXiv:2307.02719 [cs.LG]*, July 2023.

[10] Yixin Liu, Zhao Li, Shirui Pan, Chen Gong, Chuan Zhou, and George Karypis. Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2378–2392, June 2022. arXiv:2103.00113 [cs].

[11] Joana Lorenz, Maria Inês Silva, David Aparício, João Tiago Ascensão, and Pedro Bizarro. 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*, pages 1–8, 2020.

[12] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

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

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

[15] Xiaobing Sun, Wenjie Feng, Shenghua Liu, Yuyang Xie, Siddharth Bhatia, Bryan Hooi, Wenhan Wang, and Xueqi Cheng. Monlad: Money laundering agents detection in transaction streams. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 976–986, 2022.

[16] Adam Turner and Angela Samantha Maitland Irwin. Bitcoin transactions: a digital discovery of illicit activity on the blockchain. *Journal of Financial Crime*, 25(1):109–130, 2018.

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

[18] Kuoliang Wu, Deng Cai, and Xiaofei He. Multi-label active learning based on submodular functions. *Neurocomputing*, 313:436–442, 2018.