mathematics

MDPI

Article

Context-Guided Traversal with Deep Reinforcement Learning for Suspicious Subgraph Detection in Cryptocurrency Transaction Graphs

Seok-Jun Buu [1] and Hae-Jung Kim [2,*]

[1] Department of Computer Science and Engineering, Gyeongsang National University, Jinju 52828, Republic of Korea; sj.buu@gnu.ac.kr
[2] Department of Computer Science, Kyungil University, Gyeongsan 38428, Republic of Korea
* Correspondence: hjkim325@kiu.kr

Abstract

Detecting suspicious activity in cryptocurrency transaction networks is important for identifying illicit financial flows. When funds pass through several participants, the relationships among their transactions can reveal patterns that individual account attributes alone do not explain. Subgraph classification provides a way to evaluate these connected transactions as a group, using participant composition and local transaction paths. However, aggregating participant attributes does not preserve connection order, whereas observing only a few paths can overlook relevant information elsewhere in the group. Moreover, subgraph labels do not specify which paths are useful for detection. In this paper, we propose context-guided traversal with deep reinforcement learning to select informative paths according to the transaction group being examined. A role-wise context representation captures the composition of the group and its funding and receiving participants, enabling path selection to reflect the surrounding transaction relationships. This context is combined with the accumulated path representation to assess neighboring candidates and adapt subsequent observations. The traversal policy learns from subgraph classification performance without requiring annotated paths. Predictions from the selected paths are then combined with global transaction information to incorporate evidence beyond the visited nodes. In ten-fold cross-validation on the Elliptic2 Bitcoin dataset, the proposed method achieves average precision of 0.9657 and an F1 score of 0.9191, exceeding the corresponding mean scores of all nine evaluated comparison methods. A separate entity-disjoint test gives the highest three-seed mean AP of 0.6734 among four complete detectors. Transaction case analyses further show how the combined detector corrects missed suspicious subgraphs and false alarms from classification based on global information alone.

Keywords: suspicious subgraph detection; cryptocurrency transaction graphs; context-guided traversal; deep reinforcement learning

MSC: 68T07

Check for updates

Academic Editors: Vasilios N. Katsikis, Xinwei Cao, Shuai Li, Dimitris Balios, Theodore E. Simos and Anatoliy Swishchuk

Received: 26 September 2026
Revised: 23 October 2026
Accepted: 26 October 2026
Published: 27 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Cryptocurrency transaction networks provide public records of transfers while separating these records from directly identifiable account holders. This combination supports financial analysis but also allows illicit funds to move among multiple participants before reaching legitimate services. The FBI continues to report substantial cryptocurrency-related

losses [1]. In the Colonial Pipeline investigation, the U.S. Department of Justice traced ransom proceeds through Bitcoin transfers and recovered approximately 63.7 BTC from a payment of approximately 75 BTC [2]. The relationships along a transfer route therefore provide evidence that cannot be reduced to an isolated account description.

Figure 1 illustrates the transfer sequence documented in the seizure affidavit [2]. Its purpose is to show why connected observations matter: the investigative object is a flow of funds across participants. A recipient can belong to a legitimate service even when the preceding flow is suspicious. Consequently, assigning a suspicious label to a transaction group and identifying every participating account as criminal are different tasks.

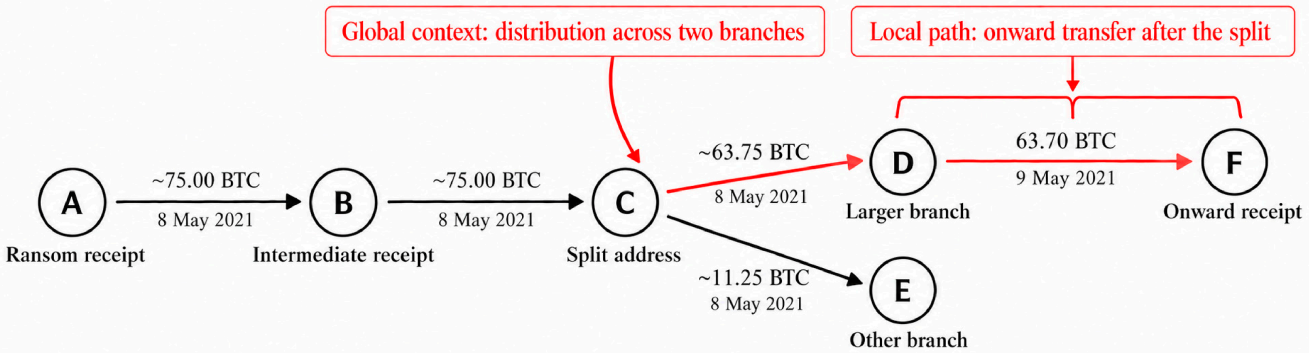


Figure 1. Illustrative Bitcoin transfer flow in the Colonial Pipeline investigation. Connected transfers motivate evaluating a flow rather than assigning the same ownership label to all participants. Letters identify addresses, arrows indicate transfers, and amounts are in BTC. Red highlights the larger branch and its onward transfer.

This study addresses suspicious subgraph classification. Given a target transaction subgraph and the attributes and relationships of its surrounding funding and receiving participants, the task is to estimate whether the target subgraph represents a suspicious flow. A node represents a transaction participant, and a directed edge represents an observed transfer relationship. The output is one suspiciousness score and one decision for the target subgraph. By evaluating connected transfers as a group, subgraph classification helps prioritize suspicious fund flows for investigation across multiple participants.

Two complementary forms of evidence are relevant to this task. The composition of a transaction group indicates which kinds of participants supply, transfer, and receive funds. An ordered path describes how selected participants are connected. Aggregating node attributes offers a broad view of the group but does not preserve the order in which connected participants are observed. Conversely, a small collection of paths retains local relationships while omitting other participants. Effective subgraph classification therefore requires a connection between the broad transaction context and the particular relationships examined within it.

This connection also creates a learning problem. The available labels specify whether a subgraph is suspicious but do not identify an optimal observation path. The usefulness of visiting a neighbor depends on the group being examined and the observations already collected. A path-selection mechanism should consequently adapt to both the transaction context and its own observation history. Reinforcement learning provides a way to learn these discrete sequential choices from their eventual classification outcome, without requiring annotated paths.

In this paper, we propose context-guided traversal with deep reinforcement learning for suspicious subgraph detection. A role-wise context encoder summarizes internal members and external funding and receiving participants. A recurrent path encoder records the visited sequence. The selection policy combines these representations with each neighboring

candidate's attributes to decide where to observe next. After traversal, the selected paths produce a subgraph prediction that is combined with predictions from global transaction information. The resulting detector uses transaction context to guide observation while retaining evidence outside the visited paths.

The contributions of this study are summarized as follows.

- We introduce role-conditioned observation selection for cryptocurrency subgraph classification. Separating funding, member, and receiving information makes the next-node decision depend on the transaction group's composition as well as its previously observed path, connecting broad context to local relational evidence.
- We formulate path selection as a classification-guided sequential decision process. A baseline-adjusted policy-gradient objective learns which observations help distinguish suspicious and licit subgraphs using subgraph labels alone, removing the need for manually annotated observation paths.
- We demonstrate the effectiveness of the complete detector on Elliptic2. Across ten evaluation folds, it achieves mean AP of 0.9657 and F1 of 0.9191, exceeding the corresponding mean scores of nine evaluated comparison methods. Representation and transaction-case analyses connect these results to improved separation of difficult subgraphs and concrete corrections of classification errors.

The remainder of this paper reviews transaction-graph representations in Section 2, presents the proposed method in Section 3, and evaluates detection performance and its underlying evidence in Section 4. Section 5 concludes the paper and discusses future research.

2. Related Work

2.1. Cryptocurrency Transaction Analysis

Cryptocurrency detection studies differ in the object they classify and the transaction evidence they represent. Node-level methods associate a label with an account or transaction, whereas subgraph methods judge a connected group within a larger network. This distinction determines how predictions should be interpreted. A model that labels an individual account does not directly solve the problem of assessing a flow whose participants can have different roles and ownership labels.

Account-level studies provide evidence that transaction relationships are useful for detecting malicious participants. Chen et al. combine graph-based cascade features with a LightGBM-based dual-sampling ensemble to identify phishing accounts on Ethereum [3]. Buu and Kim use disentangled prototypical graph convolution to learn discriminative representations for phishing detection [4]. More recently, IGT4ETH incorporates transaction topology and role embeddings into a pretrained graph transformer for Ethereum account classification [5]. These studies improve representations for account prediction. Assessing a connected flow requires a subgraph-level target that combines evidence from multiple participants.

Graph-based anti-money-laundering research has sought to preserve both transaction attributes and their relationships. LB-GLAT uses forward and reverse transaction graphs together with layer attention to represent transaction topology and address information propagation across layers [6]. Ouyang et al. use subgraph contrastive learning for Bitcoin money-laundering detection and include Random Forest as a domain baseline [7]. These studies motivate relational modeling while emphasizing the need to compare neural representations with capable attribute-based classifiers.

Elliptic2 shifts the prediction target to labeled Bitcoin subgraphs [8]. Song et al. further investigate the surrounding funding and receiving entities through RevClassify, using set-based and bipartite representations [9]. Their work establishes that participants around a target group can provide valuable evidence. The present study builds on this subgraph-level perspective but uses transaction context to condition a sequence of local observations. Context therefore has an operational role in deciding which connected information to read, in addition to its role in final classification.

2.2. Attribute, Relational, and Traversal Representations

The representations most relevant to this problem can be organized by the information retained before prediction: participant attributes and sets, neighborhood relationships, and ordered graph observations. This organization clarifies why strong performance from one family does not eliminate the need to consider the others. Table 1 summarizes the approaches and their training objectives.

Attribute aggregation describes a transaction group through statistics of its participants. A Random Forest or multilayer perceptron can classify these summaries without explicitly representing traversal order. Deep Sets applies learned element-wise transformations and permutation-invariant pooling to variable-sized sets [10]. RevClassify's set-based formulation preserves the distinction between funding and receiving sets [9]. These representations are suited to broad participant composition, but an explicit sequence of connected observations is absent from the pooled representation.

Graph representation learning also supports the classification of security objects beyond transaction accounts. Bu and Cho combine control-flow graphs, a graph transformer, and triplet learning for few-shot malware classification [11]. This approach learns discriminative representations of graph-structured security objects. The present study additionally learns which connected observations to select for transaction-subgraph classification.

Message-passing methods update node representations using adjacent nodes. GCN applies normalized neighborhood convolution [12], GraphSAGE learns neighborhood aggregation [13], and GAT assigns attention weights to neighboring messages [14]. GIN studies the discriminative capacity of aggregation functions and uses sum-based updates [15]. Directed-multigraph models, including Multi-GIN, extend this perspective to graph structures in which direction and repeated relations matter [16]. These methods encode relationships through propagation. Their local attention or aggregation should be distinguished from a policy that repeatedly chooses one next observation and carries its accumulated history forward.

Recent graph-learning studies also use reinforcement learning to refine community assignments. MaGNet-BN [17] combines Bayesian graph representations and Markov transitions with policy-based community refinement, while TIDE-MARK [18] applies temporal graph representations and reinforcement-based refinement to evolving communities in fake-news cascades. These methods learn changes to community membership. Our policy instead selects which connected participants to observe for classifying a given transaction group, using the group context and accumulated observation history.

Walk-based methods express graph structure as ordered observations. Recurrent Attention Walk learns graph traversal for node classification with reinforcement learning [19], demonstrating that classification feedback can train discrete movements. AgentNet learns graph-level representations through agents that move over a graph [20]. More recent random-walk neural networks revisit the expressive information contained in walk records [21], while NeuralWalker combines sequence-based walk representations with message passing [22]. Learn to jump further studies of adaptive walks across graph hierarchies [23]. These studies establish graph traversal as a meaningful representation strategy

and provide the context for distinguishing how a path is selected from how its observations are encoded.

RAW establishes that classification feedback can train discrete graph traversal. Building on this capability, our design makes the target transaction group the reference for both observation selection and final prediction. Funding/member/receiving composition, recurrent observation history, and candidate attributes jointly determine the next observation for a suspicious-subgraph label. A uniform-reference observation of the same group supplies the baseline-adjusted return, so the learning criterion compares observations within that classification task. Global attribute and set predictions preserve evidence outside the selected paths. The contribution is this connection between group-conditioned selection and complete-group prediction. Section 4 compares the complete detector with representative alternatives, then tests learned selection and context separately. Section 4.7 additionally compares the complete detector with RAW adapted to the same group-classification target.

Table 1. Graph representation and observation selection in related approaches. The table summarizes the original training objectives. Section 4.1 describes their adaptations for subgraph classification.

Approach	Method	Graph Representation	Observation Selection	Training Objective
Attribute/set aggregation	Deep Sets [10], 2017	Permutation-invariant sets	No traversal	Set classification
Attribute/set aggregation	RF baseline [7], 2024	Aggregate attributes	No traversal	Impurity reduction
Attribute/set aggregation	RevClassify (DS) [9], 2024	Funding/receiving sets	No traversal	Subgraph classification
Message passing	GCN [12], 2017	Normalized neighbor aggregation	No traversal	Node classification
Message passing	GraphSAGE [13], 2017	Sampled neighbor aggregation	Fixed neighbor sampling	Task prediction/ proximity
Message passing	GAT [14], 2018	Attention-weighted neighbors	No traversal	Node classification
Message passing	GIN [15], 2019	Sum aggregation	No traversal	Graph classification
Message passing	RevClassify (BP) [9], 2024	Funding/receiving bipartite graph	No traversal	Subgraph classification
Message passing	Multi-GIN [16], 2024	Multiple directed relations	No traversal	Transaction/account classification
Walk representation/ learned traversal	RAW [19], 2020	Recurrent path state	Learned policy	Classification + reward
Walk representation/ learned traversal	AgentNet [20], 2023	Agent and node states	Learned attention	Task prediction
Walk representation/ learned traversal	NeuralWalker [22], 2025	Walk sequences + message passing	Fixed walk rule	Task prediction
Walk representation/ learned traversal	RWNN [21], 2025	Walk and adjacency records	Fixed walk rule	Task prediction
Proposed method	Ours	Global context + recurrent paths	Context-guided policy	Subgraph classification + relative loss reduction

3. Proposed Method

3.1. System Overview

A suspicious transaction flow may be characterized by both the participants involved and the relationships linking them. Observing every participant through an unordered summary obscures the selected connection sequence, whereas relying only on a short path leaves part of the group unexamined. The specific design question is which connected participant to examine next within a finite path budget. Recurrent observation history lets this choice depend on information already collected, while role-wise group composition supplies a wider reference for assessing each candidate. The proposed architecture connects these local choices to complete-group prediction by combining the selected-path score with global evidence.

Figure 2 presents the overall structure. The node encoder first transforms participant attributes and role indicators into shared node embeddings. The role-wise context encoder then summarizes the observed group's composition. In parallel, a gated recurrent unit (GRU) accumulates the selected node sequence. At each step, the policy combines the context vector, the current GRU state, and candidate information to produce next-node probabilities. A selected node updates the same GRU, and the selection-encoding cycle continues until the observation budget is exhausted or no valid neighbor remains. The completed path states are pooled for subgraph classification.

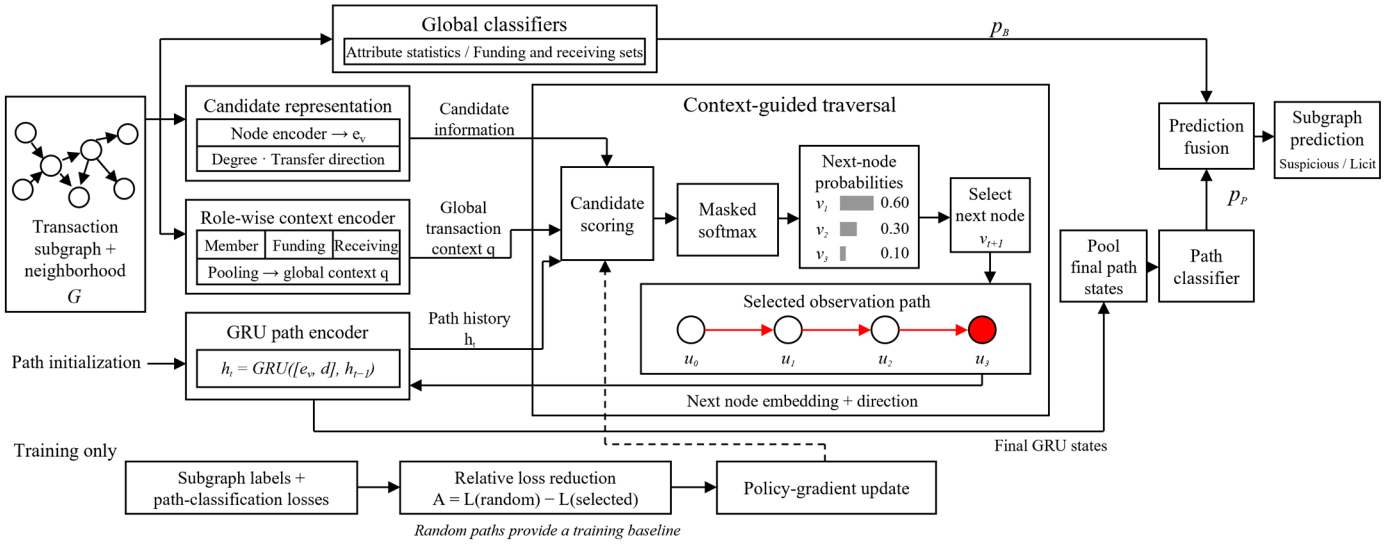


Figure 2. Overview of context-guided traversal for suspicious subgraph detection. Group context, recurrent path history, and candidate information guide next-node selection. Selected observations support a path prediction, which is combined with global transaction predictions. Classification-guided policy updates are used during training. Solid arrows show information flow, the dashed arrow shows the training update, and red marks the selected observation sequence and its current endpoint. G, q, and h denote the graph, context, and recurrent state, respectively.

Let $G = (V, E, X)$ denote the observed graph associated with a target subgraph. The node set contains target members and the available external funding and receiving participants. The roles describe their relationship to the target group, not their class labels. The task is to learn a score for the target subgraph from G. During training, the known subgraph label evaluates the selected observations. During inference, the detector uses attributes, relationships, and the learned policy without accessing the true label or a reference path.

3.2. Role-Wise Transaction Context

A funding participant and a receiving participant can have similar attributes but different meanings for a flow. Pooling their features together would remove this distinction before the policy chooses its next observation. The context encoder preserves the internal, incoming, and outgoing composition of the target group, allowing the same neighboring attributes to be interpreted in relation to different transaction roles.

For a node v, let x_v denote its attribute vector and r_v its three role indicators. A shared multilayer perceptron produces the embedding e_v used by both the context and path components.

$$e_v = \mathrm{MLP}_{\mathrm{enc}}([x_v; r_v]). \tag{1}$$

For each role r in the set of member, funding, and receiving roles, the encoder computes the mean and maximum node embeddings and a logarithmic count. The mean represents the typical attributes of the role, the maximum retains prominent feature responses, and the count represents group size. Their concatenation is projected into the context vector q.

$$u_r = \left[mean_{v \in V_r} e_v; max_{v \in V_r} e_v; \log(1+|V_r|)\right]. \tag{2}$$

$$q = \tanh(W_q[u_{\mathrm{member}}; u_{\mathrm{funding}}; u_{\mathrm{receiving}}] + b_q). \tag{3}$$

Here V_r is the node set for role r. An external participant that both sends and receives funds contributes to both corresponding summaries. Empty sets have zero mean and maximum summaries. The learned context summarizes the observed transaction group. Figure 3 shows how the role partitions are maintained before projection.

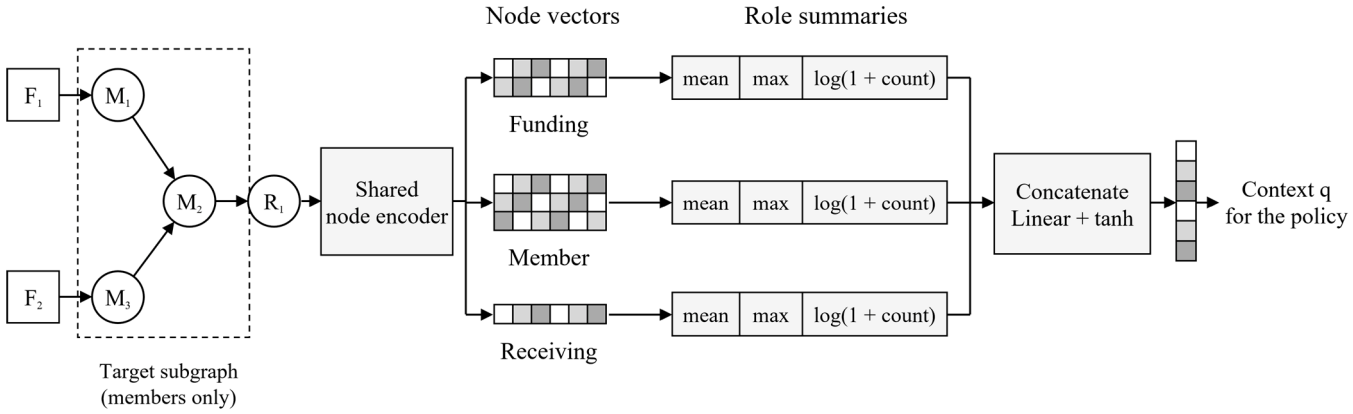


Figure 3. Role-wise context encoding. Node embeddings are grouped by funding, member, and receiving roles, preserving participant composition before projection into the context vector used for selection. Circles denote participants, arrows show transfers or feature transformations, and shaded vector cells illustrate feature values. The dashed box encloses target members.

The context vector guides the selection policy. It is not directly supplied to the path classification head. This separation requires the path prediction to be supported by the observations actually collected, while allowing the policy to choose those observations with knowledge of the broader group. The independent global classifiers described in Section 3.5 provide a separate route for broad information to affect the final decision.

3.3. Context-Conditioned Traversal and Recurrent Path Encoding

The next useful observation depends on what the detector has already seen. For example, observing a funding participant changes which adjacent internal participant may provide additional evidence. The policy therefore evaluates a candidate jointly with the

group context and the recurrent path state, rather than assigning a context-free importance score to the candidate.

At step t, let u be the current node, h_t the accumulated path state, and C_t the valid candidate set. Candidate information comprises the embedding e_v, logarithmic degree, and two indicators describing the original transfer direction between u and v. One shared scoring network is applied to every candidate.

$$s_t(v) = \mathrm{MLP}_{\mathrm{sel}}([q; h_t; e_v; \log(1 + \deg(v)); d_{u,v}]). \tag{4}$$

$$\pi_\theta(v \mid s_t) = \frac{\exp(s_t(v))}{\sum_{w \in C_t} \exp(s_t(w))}, v \in C_t. \tag{5}$$

The softmax is evaluated only over valid neighbors. Thus, the network produces one scalar per candidate and does not require a fixed number of meaningful actions. A padded implementation can mask nonexistent or already visited candidates.

The recurrent path encoder is implemented with a gated recurrent unit (GRU). After the policy samples the next node, this encoder reads its embedding and transfer-direction indicators. The updated state carries the observation order to the next policy decision. GRUs provide a recurrent mechanism for accumulating sequential information [24]. In the proposed method, the state update and the final path prediction are defined by

$$h_{t+1} = \mathrm{GRU}\left(\left[e_{v_{t+1}}; d_{u,v_{t+1}}\right], h_t\right). \tag{6}$$

$$p_P = \sigma\left(\mathrm{MLP}_{\mathrm{path}}\left(\frac{1}{K}\sum_{k=1}^{K} h_{T_k}^{(k)}\right)\right). \tag{7}$$

The start node initializes the GRU from a zero state with zero direction indicators. K denotes the number of paths, T_k the last step of path k, and σ the sigmoid function. Figure 4 separates the three policy inputs and shows the feedback from a selected node to the recurrent state.

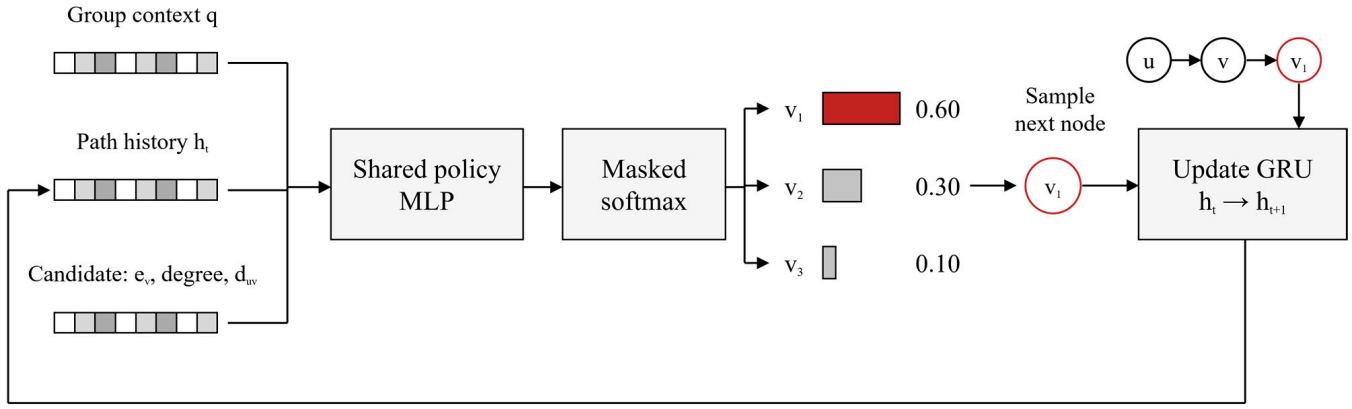


Figure 4. Context-conditioned next-node selection. The policy evaluates neighboring candidates using group context and recurrent path history. A sampled next node updates the GRU for the subsequent decision. The displayed probabilities are illustrative. Arrows show information flow. Red highlights the sampled node and its probability, and shaded vector cells illustrate features.

The default detector selects two paths with at most four moves per path. Starts are sampled uniformly from internal members without replacement when at least two members are available. A node cannot be revisited within the same path, but different paths can subsequently overlap. Traversal can follow either direction of an observed transfer relationship, with the original direction retained as an input. The learned path is therefore an observation sequence over the transaction graph, not a reconstruction of the chronological movement of an individual unit of currency. Algorithm 1 specifies this procedure.

Algorithm 1. Path selection and encoding (WALK)

Input: observed graph G, model parameters θ, policy mode, random draw ξ. Output: logit z, path log-probability ℓ, entropy H, selected paths τ.
Symbols: X, node attributes; q, role-wise context; e, node embeddings; h, GRU state; k, path index; C, valid neighbors; s, candidate logits; π, selection probabilities; ℓ_k, H_k, and τ_k, log-probability, entropy, and sequence of path k. θ denotes trainable parameters.

```
e ← Encode_θ(X, roles); q ← Context_θ(role-wise mean, max, size)
two starts ← sample uniformly without replacement from member nodes (reuse only if one node exists)
for each start u: visited ← {u}
    h ← GRU_θ([e[u], 0, 0], 0); ℓ_k ← 0; H_k ← 0; τ_k ← [u]
    for t = 1, ..., 4:
        C ← fixed adjacent candidates[u]\visited //Up to 16 candidates
        if C = ∅: break                          //Retain the final state
        s[v] ← MLP_θ(q,h,e[v], log(1 + degree[v]), direction[u,v])   for v ∈ C
        if mode = uniform: s[v] ← 0              for v ∈ C
        π ← softmax(s); v ← argmax(log π + Gumbel(ξ))
        h ← GRU_θ([e[v], direction[u,v]], h)
        ℓ_k ← ℓ_k + log π[v]; H_k ← H_k − Σ_j π[j] log π[j]
        u ← v; visited ← visited ∪ {v}; append v to τ_k
    final_state[k] ← h
z ← Head_θ(mean(final_state[1], final_state[2]))
return z, Σ_k ℓ_k, Σ_k H_k, {τ_1,τ_2}
```

3.4. Classification-Guided Policy Learning

Subgraph labels indicate the desired classification but do not prescribe the route that should be observed. Moreover, a next-node choice affects both the current observation and the neighbors available afterward. We therefore learn traversal as a sequential decision process whose terminal outcome is the classification loss of the completed observations. This formulation directly connects path usefulness to the detection task without introducing a target path that the policy must imitate.

The reward asks whether a selected observation helps classify the transaction group. It encourages the policy to collect relationships that the path reader can use to distinguish suspicious and licit groups. Section 4.3 tests this predictive function by changing the observation policy, and the case studies connect selected relationships to particular classification decisions. A high reward is therefore evidence of classifier usefulness, not a forensic attribution to the participants on a path.

The environment is the observed transaction graph. The agent is the selection policy. Its available actions are the unvisited neighbors allowed by the candidate limit, and its state representation consists of the role-wise context, recurrent path history, and candidate features. Position, visited nodes, and the remaining move budget determine the action set and stopping condition. An action moves the observation location to one neighbor and updates the recurrent state. An episode completes the prescribed paths for one subgraph and evaluates their pooled representation against that subgraph's label.

Let τ denote observations sampled by the policy and y the subgraph label, with 0 for Licit and 1 for Suspicious. The path classifier is trained with a class-weighted binary cross-entropy loss. The weight w_y is determined from the training class distribution.

$$L_{\text{cls}}(\tau, y) = -w_y[y \log p_\theta(\tau) + (1-y)\log(1 - p_\theta(\tau))]. \tag{8}$$

A useful path reduces this loss for either class. Consequently, the objective rewards observations that support correct licit decisions as well as correct suspicious decisions.

To evaluate a selected path relative to the difficulty of its transaction group, we independently sample uniform-reference observations from the same graph. Both observation

sets use the same path count, move limit, revisit constraint, encoder, and classifier. As shown in Figure 5, the reference supplies a classification baseline.

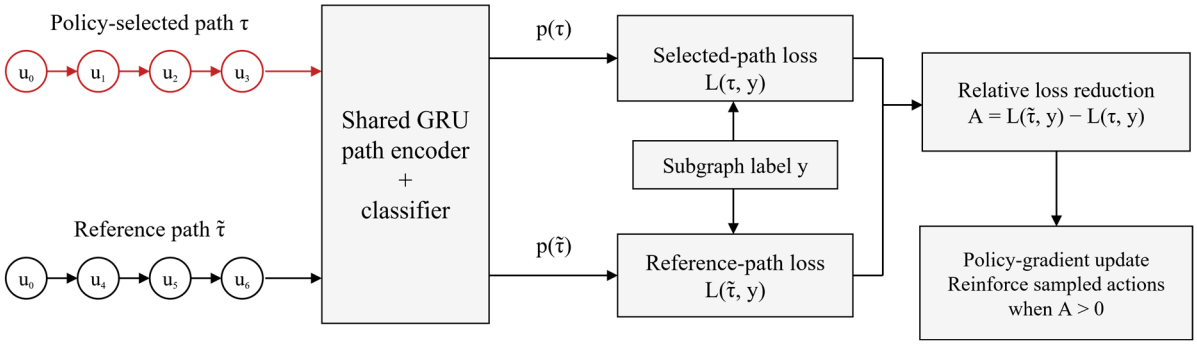


Figure 5. Classification-guided policy learning. Selected observations and independent uniform-reference observations are evaluated against the same subgraph label. Their loss difference supplies a baseline-adjusted return for the policy. One path per set is drawn for clarity. The detector pools two paths. Arrows show computation flow. Red highlights policy-selected observations, and the tilde distinguishes independent reference observations.

The terminal reward is the negative classification loss. Subtracting the reference reward gives a baseline-adjusted return A, which measures the relative usefulness of the selected observations. A tilde identifies the independently sampled reference observations.

$$R = -L_{\text{cls}}(\tau, y), b = -L_{\text{cls}}(\tilde{\tau}, y), A = R - b. \tag{9}$$

$$A = L_{\text{cls}}(\tilde{\tau}, y) - L_{\text{cls}}(\tau, y). \tag{10}$$

For example, losses of 0.4 for the selected observations and 0.6 for the reference give $A = 0.2$. The selected observations support the known subgraph label more accurately, so their action probabilities receive a reinforcing update. A negative A provides the opposite signal. Comparing observations of the same graph reduces the influence of between-graph classification difficulty on this interpretation.

Discrete node sampling prevents ordinary backpropagation through a selected node index from directly training the choice itself. A REINFORCE estimator instead connects the terminal return to the log-probabilities of the sampled actions [25]. Let ℓ_θ sum the valid action log-probabilities across all paths. The policy and entropy terms are

$$\ell_\theta(\tau) = \sum_{k,t} \log \pi_\theta(a_{k,t} \mid s_{k,t}). \tag{11}$$

$$\mathcal{L}_{\text{policy}} = -\mathit{stopgrad}(A)\ell_\theta(\tau), \mathcal{L}_{\text{explore}} = -\lambda \sum_{k,t} H(\pi_\theta(\cdot \mid s_{k,t})). \tag{12}$$

$$\hat{\mathcal{L}} = L_{\text{cls}}(\tau, y) + \mathcal{L}_{\text{policy}} + \mathcal{L}_{\text{explore}}. \tag{13}$$

The three terms have distinct functions. Classification loss trains the representations and classification head to predict the subgraph label. The policy term increases the log-probability of actions with positive relative return and decreases it for negative return. Entropy regularization discourages premature concentration on a small number of candidates, allowing the policy to explore alternative connections.

Both losses used to form A are detached from differentiation within the policy term. The reference is sampled independently of the policy's realized actions. An action-independent baseline can be subtracted because its expected score-function contribution is zero [25]. No learned value critic or reference-path imitation objective is introduced. The same terminal signal supervises the choices contributing to the pooled path prediction. Algorithm 2 summarizes training and validation.

Algorithm 2. Policy-gradient training, validation, and inference

Input: training set T, validation set V, 6000 update budget, path parameters θ, separately fitted global classifiers.
Output: selected parameters θ^*, fusion weights α and β, threshold c, and final score.
Symbols: p_{RF}, Random Forest score; p_{DS}, Deep Sets score; p_P, path-classifier score; p_B, global fused score; p, final score.
Suffix V denotes validation predictions; y_B and y_V denote labels. α and β are fusion weights; c is the decision threshold.
WALK is Algorithm 1; BCE is binary cross-entropy; AP is average precision.

Initialize θ; Adam(learning rate 0.001); best validation AP $\leftarrow -\infty$
for step = 1, ..., 6000:
    B $\leftarrow$ sample 128 graphs with replacement from training set T
    mode $\leftarrow$ uniform if step $\leq$ 200 else learned policy
    $(z, \ell, H, \tau) \leftarrow \mathrm{WALK}_\theta(B, \text{mode}, \xi)$
    L $\leftarrow$ training-prevalence-weighted BCE(z, y_B)
    J $\leftarrow$ mean(L)
    if step > 200:
        $(\tilde{z}, _, _, _) \leftarrow \mathrm{WALK}_{\text{stopgrad}(\theta)}$(B,uniform,independent random draw)
        $\Delta L \leftarrow$ stopgrad(L − weighted BCE$(\tilde{z}, y_B)$)
        J $\leftarrow$ J + mean$(\Delta L \times \ell)$ − 0.01 × mean(H)
    $\theta \leftarrow$ Adam(θ, clip to global norm 5: ∇_θ J)
    if step mod 400 = 0:
        validation scores $\leftarrow$ mean sigmoid(z) over three walk draws
        if AP(V) > best validation AP: $\theta^* \leftarrow \theta$; best validation AP $\leftarrow$ AP(V)
$p_{RF,V}$, $p_{DS,V}$, $p_{P,V} \leftarrow$ validation predictions of the fitted classifiers
$(\alpha^*, \beta^*) \leftarrow \mathrm{SelectFusionWeights}(p_{RF,V}, p_{DS,V}, p_{P,V}, y_V)$
$p_V \leftarrow (1-\beta^*)[(1-\alpha^*)p_{RF,V} + \alpha^* p_{DS,V}] + \beta^* p_{P,V}$
$c^* \leftarrow \mathrm{argmax}_c\, F1(y_V, 1[p_V \geq c])$
Inference(G): $p_P \leftarrow$ mean sigmoid scores over three independent $\mathrm{WALK}_\theta{}^*$ draws
    $p_B \leftarrow (1-\alpha^*)p_{RF} + \alpha^* p_{DS}$; $p \leftarrow (1-\beta^*)p_B + \beta^* p_P$
    return p, $1[p \geq c^*]$

3.5. Combining Selected Paths with Global Evidence

A short observation sequence cannot include every participant. The final detector therefore combines its path prediction with global classifiers that retain broader transaction information. A Random Forest uses role-wise attribute statistics, while a Deep Sets classifier encodes the external funding and receiving sets following the set-based RevClassify formulation [9]. The two global predictions are first combined, after which the path prediction supplies complementary ordered evidence.

$$p_B = (1-\alpha)p_{\mathrm{RF}} + \alpha p_{\mathrm{DS}}. \tag{14}$$

$$p_{\mathrm{Ours}} = (1-\beta)p_B + \beta p_P. \tag{15}$$

Here p_{RF} and p_{DS} denote Random Forest and Deep Sets scores, p_{B} denotes their global combination, and p_{P} denotes the selected-path score. The component models are fitted on training data. The two combination coefficients are selected using validation AP, and a decision threshold is selected using validation F1. These quantities are fixed when evaluating the corresponding outer fold. This separation makes the contributions of broad composition and selected observations explicit, while reserving final decisions for the complete detector.

4. Experimental Results

4.1. Dataset and Experimental Settings

We evaluate the proposed method on the Bitcoin Elliptic2 dataset [8]. Its public release contains 121,810 labeled subgraphs within a background graph of 49,299,864 address clusters and 196,215,606 transfer edges. Each node has 43 attributes. The prediction target is the subgraph's Licit or Suspicious label, rather than a label assigned to every participating

address cluster. Address-cluster identifiers are used to reconstruct and inspect graphs, not as input features.

The evaluation pool contains 97,448 subgraphs, including 95,249 Licit and 2199 Suspicious examples. Table 2 describes their observed composition. Funding and receiving participants extend the target's internal members with the available surrounding transfer relationships. The similar average sizes of the two classes indicate that the task cannot be explained simply as separating large groups from small groups. Attributes, roles, and relationships must instead be considered together.

Table 2. Composition of the evaluation pool. Node and link counts are means ± standard deviations. Links connect distinct ordered cluster pairs.

Statistic	Licit	Suspicious
Subgraphs	95,249	2199
Share	97.74%	2.26%
Member nodes	3.65 ± 3.17	3.80 ± 2.65
Funding nodes	1.59 ± 2.01	1.54 ± 1.70
Receiving nodes	1.43 ± 2.17	1.65 ± 2.43
All observed nodes	6.51 ± 4.77	6.82 ± 4.34
Directed links	6.00 ± 5.75	6.24 ± 4.75

Ten outer folds are used for evaluation. In each round, the remaining subgraphs are divided into training and validation sets, yielding 78,932–78,933 training examples, 8771 validation examples, and 9744–9745 evaluation examples. Node standardization is fitted using unique observed entities associated with training subgraphs. Checkpoints, combination weights, and classification thresholds are selected on the inner validation set. Each outer fold is evaluated once with those choices fixed. This protocol evaluates held-out subgraphs within the same network.

Across the ten folds, 97.16–97.63% of evaluation groups contain at least one observed participant also present in fitting, including internal, funding, and receiving participants. The entity-disjoint evaluation described below tests groups whose internal, funding, and receiving participants are all excluded from fitting and validation.

Both the outer partition and inner validation sampling preserve the severe class imbalance. Suspicious examples constitute approximately 2.26% of the evaluation pool. Validation alone selects operating thresholds. F1 provides a common reference operating point. Additional thresholds selected for specified validation false-positive rates quantify the trade-off between missed suspicious groups and false alarms in Section 4.2.

Parallel transfers between an ordered cluster pair are collapsed into a directed relationship. The primary comparison uses participant attributes and directed relationships. The additional edge-summary comparison is described below. Ours uses 32-dimensional representations, two paths, at most four moves per path, and up to 16 fixed adjacent candidates before visited-node masking. Optimization uses Adam with learning rate 0.001, batch size 128, gradient-norm clipping at 5, and 6000 updates. The first 200 updates train the path classifier using uniform observations before policy-gradient updates begin. Validation AP is measured every 400 updates. Prediction averages three independent path draws, retaining stochastic policy selection during evaluation. The entropy coefficient is 0.01. Combination weights are selected from 0 to 1 in increments of 0.1.

Internal starting members are sampled uniformly without replacement until all members have been used. If a group has fewer members than paths, sampling restarts from its internal members. Thus, increasing the path count broadens starting positions whenever distinct members are available. The joint path-count and move-limit analysis in Section 4.4 examines the resulting observation coverage.

Collapsing parallel transfers defines a relationship for selecting the next participant. To evaluate the information omitted by this interface, Section 4.3 adds group-level means, standard deviations, and maxima of all 95 public edge attributes to the global Random Forest input. Four further features describe transfer count, distinct ordered-pair count, and mean and maximum pair multiplicity, with $\log(1 + x)$ applied to the two counts. The 289 additional features use the original observed graph scope and fit-only preprocessing. Anonymous attribute indices are not assigned undocumented amount or time meanings. This comparison enriches the global summary while keeping the traversal policy unchanged.

We additionally evaluate nonempty funding/receiving configurations not seen in fitting or inner validation using one fixed, non-temporal partition. A configuration is the ordered pair of sorted unique funding and receiving entity sets. Groups sharing a configuration remain in the same partition. The assignment balances group counts, positive counts, and size-by-boundary-count distributions, retaining all 97,448 groups and their original relationships. The empty configuration is assigned to fitting. The fit, inner, and evaluation sets contain 68,704, 14,372, and 14,372 groups, including 1545, 327, and 327 suspicious groups. This evaluates new surrounding combinations under the original group-classification target. The ten-fold and unseen-configuration evaluations use the same Elliptic2 pool used during method design.

The fixed boundary-signature comparison trains the global Random Forest for 200 trees and Deep Sets, the learned path model, and NeuralWalker for 6000 updates each under the original settings. The partition is predetermined, and fit data determine preprocessing. Inner AP selects neural checkpoints and combination weights, and inner F1 selects thresholds before evaluation. The independently selected Deep Sets model is shared by its standalone baseline and the global combination. Section 4.7 reports this single fixed comparison separately from the ten-fold primary evaluation.

RAW is additionally implemented for the same fixed partition as a learned-traversal comparator. Its sigmoid neighbor relevance, normalized categorical moves, sum of neighbors with relevance above 0.5, recurrent update, and terminal correct/incorrect reward are retained. To adapt node classification to the group target, walks start at every internal member, and their final histories are mean pooled for one group prediction per rollout. The input projection uses the same standardized node attributes and role indicators, with transfer direction as the edge input. RAW uses 128 hidden units, ten observation steps, a learning rate of 0.0001, five training rollouts, and ten evaluation rollouts. We use unweighted binary cross-entropy, REINFORCE with discount 1, classifier L2 coefficient 0.00001, and the same 6000-update budget and inner-selection protocol. Its two-layer score and classification networks are specified in the implementation. It uses the full observed neighborhood and permits revisits, without our role-wise context vector, uniform-reference reward, or global-score combination.

For the entity-disjoint evaluation, the fitting, validation, and test sets contain 18,302, 4575, and 4742 groups, including 413, 103, and 107 suspicious groups, respectively. The original labels, participant sets, role assignments, attributes, and directed relationships are preserved for every selected group. No internal, funding, or receiving entity in the test set occurs in either fitting or validation. Fitting and validation share 556 participants, all disjoint from testing. Whole groups are selected once with class proportions of approximately 2.26% and stratification by group size and surrounding-participant count. This evaluates unseen participants under the same group-classification target.

The proposed detector, RevClassify (Deep Sets), NeuralWalker, and group-adapted RAW are each trained from scratch under three independent seeds on this fixed partition. All preprocessing uses fitting entities only. Each method uses the same fitting-derived positive weight of 17,889/413, with a unit of negative weight. This weight enters neural classification

losses and RAW policy rewards, and the Random Forest uses the same class weights. Every neural component receives 6000 updates with batch size 128, and validation AP is measured every 400 updates, providing 15 checkpoint-selection opportunities. Validation alone selects thresholds and the original combination weights. Hidden dimensions and observation strategies remain those specified in this section, preserving each model structure. Section 4.7 reports mean and sample standard deviation across training seeds, separately from the ten-fold and repeated-inference analyses.

The nine comparison methods cover four forms of transaction representation. Random Forest and MLP use the same role-wise aggregated attributes. GCN, GraphSAGE, GAT, and GIN use neighborhood message passing. The two RevClassify variants model funding/receiving sets or their bipartite relationships, whereas NeuralWalker represents sampled walks. These implementations are adapted to the same subgraph prediction task. NeuralWalker starts a walk at each node, whereas Ours uses two internal starts and retains global predictions. The comparison evaluates complete detectors with their respective observation strategies.

Random Forest uses 200 trees. RevClassify (Deep Sets) uses 128-dimensional hidden representations. MLP and GraphSAGE use two 32-dimensional layers, a learning rate of 0.001, unweighted binary cross-entropy, and up to 6000 updates. GraphSAGE concatenates self and mean-neighbor representations with layer-wise L2 normalization and uses all neighbors in the observed graph. GCN and GAT use two 32-dimensional layers and mean graph pooling. GCN adds a 32-dimensional ReLU classification head, and GAT uses four attention heads. Their update limit is 12,000 with validation every 400 updates. GCN's graph-level nonlinear head was selected on a development validation split and then fixed across the evaluation folds. GIN uses four message-passing layers and layer-wise sum readout. NeuralWalker uses two 32-dimensional layers, a walk length of 8, and a structural window of 4. Evaluation batch order is fixed for the batch-dependent normalization in the bipartite RevClassify implementation.

Average precision (AP) is the primary metric because suspicious subgraphs form a small fraction of the data and should be ranked ahead of licit examples. Precision measures the reliability of suspicious decisions, recall measures the fraction of suspicious subgraphs detected, and F1 balances the two. Decision thresholds maximize validation F1 and remain fixed for evaluation. The primary ten-fold comparison reports means and sample standard deviations across outer folds. Additional evaluations use the repetition units specified in their captions.

4.2. Comparison with Existing Methods

Table 3 evaluates whether context-guided observations, combined with global information, improve complete-detector performance. Ours achieves the highest mean AP and F1 among the nine evaluated alternatives. Relative to the strongest individual comparator, set-based RevClassify, AP increases from 0.9495 to 0.9657 and F1 from 0.9161 to 0.9191. The higher AP indicates more effective ranking of suspicious subgraphs, while the higher F1 reflects improved classification at the validation-selected operating points.

The performance distribution also explains why broad participant information remains part of the proposed detector. Random Forest and set-based RevClassify perform strongly, showing that transaction attributes and funding/receiving composition are informative. NeuralWalker reaches AP 0.9348, supporting the usefulness of walk representations. Ours combines broad evidence with context-conditioned ordered observations and improves mean AP over each of these individual approaches.

Table 3. Detection performance on ten evaluation folds. Entries are means ± sample standard deviations. All scores use the stated task adaptations. Decision thresholds are selected on validation data.

Approach	Method	AP	Precision	Recall	F1
Attribute aggregation	Random Forest [7]	0.9420 ± 0.0145	0.9194 ± 0.0265	0.8681 ± 0.0269	0.8924 ± 0.0129
	MLP	0.9004 ± 0.0163	0.8590 ± 0.0292	0.8245 ± 0.0297	0.8409 ± 0.0206
Message passing	GCN [12]	0.6822 ± 0.0217	0.6753 ± 0.0516	0.5684 ± 0.0280	0.6155 ± 0.0188
	GraphSAGE [13]	0.7150 ± 0.0454	0.7049 ± 0.0340	0.6994 ± 0.0523	0.7008 ± 0.0296
	GAT [14]	0.6958 ± 0.0372	0.6333 ± 0.0580	0.6199 ± 0.0547	0.6232 ± 0.0283
	GIN [15]	0.8194 ± 0.0171	0.8026 ± 0.0521	0.7171 ± 0.0473	0.7551 ± 0.0202
Funding/receiving structure	RevClassify (Deep Sets) [9]	0.9495 ± 0.0135	0.9423 ± 0.0228	0.8922 ± 0.0268	0.9161 ± 0.0128
	RevClassify (bipartite) [9]	0.9167 ± 0.0193	0.8986 ± 0.0257	0.8359 ± 0.0265	0.8655 ± 0.0098
Walk representation	NeuralWalker [22]	0.9348 ± 0.0129	0.9032 ± 0.0234	0.8677 ± 0.0229	0.8848 ± 0.0147
Global + learned paths	Ours	0.9657 ± 0.0120	0.9383 ± 0.0179	0.9018 ± 0.0319	0.9191 ± 0.0119

With the task adaptations and training budgets specified in Section 4.1, the message-passing models obtain AP ranging from 0.6822 for GCN to 0.8194 for GIN. The proposed detector combines participant summaries with selected relational observations to achieve higher mean AP than these message-passing alternatives.

Relative to Global, paired fold differences average 0.001684 AP and 0.003719 F1, with paired standard deviations of 0.001664 and 0.006863. Standardized paired differences are 1.012 and 0.542, respectively. For an exploratory sensitivity analysis, we approximate fold-size dependence using a standard error obtained by multiplying the paired standard deviation by the square root of (1/10 + evaluation-size/fit-size), with nine degrees of freedom [26]. This approximation uses fold sizes, while recurring participants and prior development introduce additional dependence. The approximate 95% intervals are [−0.000096, 0.003463] for AP and [−0.003620, 0.011057] for F1, with two-sided p-values of 0.0610 and 0.2812.

At the original validation-selected thresholds, the 97,448 pooled predictions contain 1983 true positives and 216 false negatives for both Global and the complete detector. False positives decrease from 150 to 132, and true negatives increase from 95,099 to 95,117. The corresponding false-positive rates are 0.1575% and 0.1386%. At these operating points, combining selected observations with global evidence reduces the false-alarm burden while preserving aggregate recall.

Varying the validation operating target makes the alert burden explicit. At inner false-positive-rate targets of 0.1%, 0.5%, and 1%, the complete detector obtains mean evaluation precision/recall of 0.9473/0.8936, 0.8064/0.9518, and 0.6768/0.9663. Achieved evaluation false-positive rates are 0.1155%, 0.5302%, and 1.0677%, respectively. The respective mean false-negative counts per fold are 23.4, 10.6, and 7.4. At these validation-selected thresholds, increasing recall reduces missed suspicious groups while lowering precision and increasing false alarms, making the investigation burden of each operating choice explicit.

4.3. Contribution of Transaction Context

The preceding comparison establishes the performance of the complete detector. Table 4 examines how participant composition and selected relational observations contribute to suspicious-subgraph detection. The first condition retains the Random Forest/Deep Sets global combination but omits the path prediction. The second classifies path observations without the graph-context input. Ours uses context-guided paths together with the global

combination. These conditions distinguish broad attribute/set information from localized sequential observations and their complete integration.

Table 4. Contribution of broad attribute/set information and local path observations. Entries are means ± sample standard deviations over ten evaluation folds.

Method	AP	Precision	Recall	F1
Attribute/set summaries	0.9641 ± 0.0117	0.9313 ± 0.0295	0.9018 ± 0.0341	0.9154 ± 0.0120
Path observations without graph context	0.9148 ± 0.0165	0.8546 ± 0.0222	0.8686 ± 0.0333	0.8610 ± 0.0168
Ours	0.9657 ± 0.0120	0.9383 ± 0.0179	0.9018 ± 0.0319	0.9191 ± 0.0119

Global attribute/set summaries achieve AP 0.9641 and F1 0.9154. Adding the context-guided path prediction increases these values to 0.9657 and 0.9191. Mean recall remains 0.9018, while precision increases from 0.9313 to 0.9383, consistent with a reduction in false alarms at the selected operating points.

Path observations without graph context reach AP 0.9148 and F1 0.8610. Local observation alone therefore leaves information that is recovered by the complete design. The complete detector combines broad transaction composition with ordered observations, allowing the final decision to use evidence from both information sources. Section 4.4 examines the path component more closely by varying how much it observes and how its policy explores.

A path encoder can exploit relational evidence only if it observes useful participants. We therefore test whether the learned choices contribute beyond simply following valid connections. With the learned encoder and classifier fixed, replacing the policy with uniform random or degree-weighted selection preserves the starts, valid-neighbor masks, stopping rules, global scores, and final combination weights. Across ten folds and three draws, path AP falls from 0.9100 with learned selection to 0.6178 and 0.6175, respectively. The final AP changes from 0.9657 to 0.9623 and 0.9626. The trained reader thus uses information supplied by its learned observations. To allow adaptation to random observations, we also retrain the uniform-policy model. Its path AP reaches 0.7709 and final AP 0.9629, remaining below the learned selection. Learned selection achieves higher path and final AP than both fixed-policy replacements and the retrained uniform model. Combining these observations with global summaries allows the detector to use both ordered relationships and broad participant information.

We next examine the role of group context in policy decisions. Masking the group-context vector under fixed weights changes path AP from 0.9100 to 0.9078, while the final AP is 0.9659 and F1 remains 0.9191. Retraining without this vector yields path AP 0.9148, final AP 0.9660, and F1 0.9181. Role context supplies group-composition information alongside recurrent history and candidate information for next-node decisions. The policy comparisons demonstrate the value of learning which observations to collect, while the combined detector uses both the selected evidence and whole-group information.

The incremental effect also depends on the evaluation setting. In the fixed unseen-configuration evaluation described in Section 4.1, Global obtains AP 0.9744 and F1 0.9325, compared with 0.9733 and 0.9331 for the complete detector. The inner-selected path weight of 0.1 recovers three missed suspicious groups and adds three false alarms.

Transfer attributes provide another possible source of evidence beyond participant composition. On three fixed folds, adding summaries of all 95 public edge attributes and transfer multiplicity changes Global AP/F1 from 0.9632/0.9184 to 0.9601/0.9191 and complete-detector AP/F1 from 0.9642/0.9206 to 0.9625/0.9219. On these same three folds, the complete detector has higher mean AP and F1 than the corresponding Global condition both with and without the added edge summaries. Adding edge summaries increases F1

and lowers AP in these comparisons. The edge attributes enter the global summary, while traversal uses the directed relationships.

4.4. Sensitivity to Observation Range and Exploration

We examine observation breadth and depth using K in {1, 2, 4} and T in {2, 4, 8} on three fixed evaluation folds, retaining the original training budget and remaining settings. Figure 6 reports means and sample standard deviations. A separate, single-fold analysis varies the entropy coefficient over {0, 0.001, 0.01, 0.1} with two paths and four moves. The ten-fold primary evaluation retains its fixed default setting.

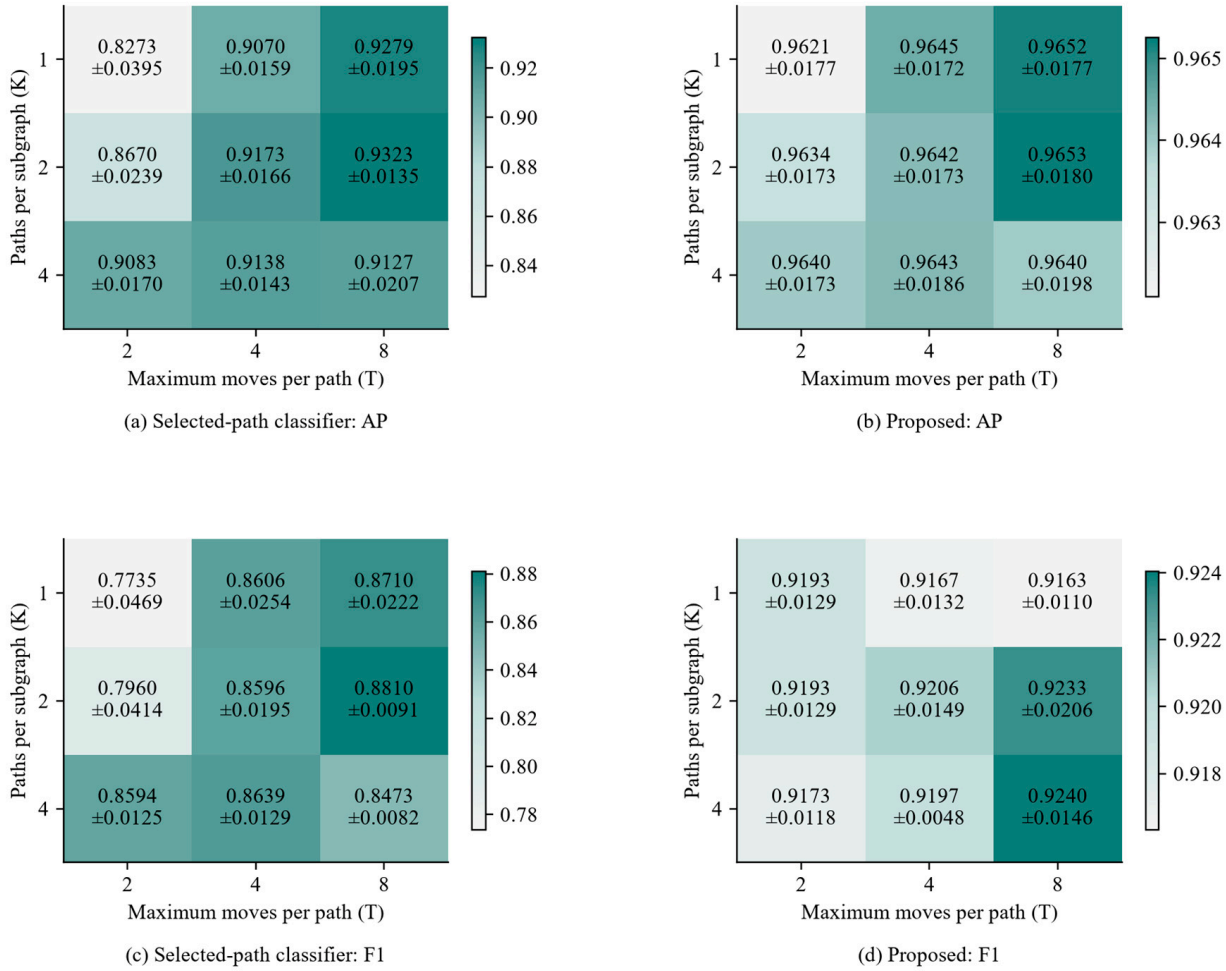
Figures 6 and 7 report two predictions from each configuration. The selected-path prediction is produced by the classifier inside Ours after reading only the selected observations. The final prediction combines that score with the global classifiers.

Across the three evaluation folds, increasing the move limit from two to four and eight raises the mean path AP from 0.8273 to 0.9070 and 0.9279 with one path, whereas the corresponding values with four paths are 0.9083, 0.9138, and 0.9127. Short observations can omit useful relational evidence, and deeper traversal improves ranking more clearly with one path than with four. For two paths, extending four moves to eight raises final AP from 0.9642 to 0.9653 and F1 from 0.9206 to 0.9233. The smaller changes in the complete detector show how global evidence moderates changes in path quality. These results support considering observation width and depth together.

Entropy regularization acts on the next-node distribution. A larger coefficient encourages a more dispersed policy during training and increases the incentive to explore alternative neighbors. With a zero coefficient, the model learns through classification and policy gradients without entropy regularization. Changing it tests whether the detector relies on a narrowly concentrated exploration setting. Path-classifier AP varies from approximately 0.9251 to 0.9409 across the tested coefficients. Final AP remains between approximately 0.9810 and 0.9818, indicating that global evidence moderates these variations. The default coefficient of 0.01 gives the highest final F1 among these four settings, although the best path-only AP occurs at 0.001.

The candidate cap controls how many neighboring entities are available to the policy at each step. Under fixed learned weights on the same three folds, caps of 8, 16, and 32 yield path APs of 0.9133, 0.9173, and 0.9174. The final AP is 0.964170, 0.964236, and 0.964242, with an unchanged F1 of 0.920637. Expanding the candidate set from 16 to 32 produces little additional ranking benefit, and F1 remains unchanged across the three caps. The inference comparison covers the observed degree distribution, which contains few high-degree cases.

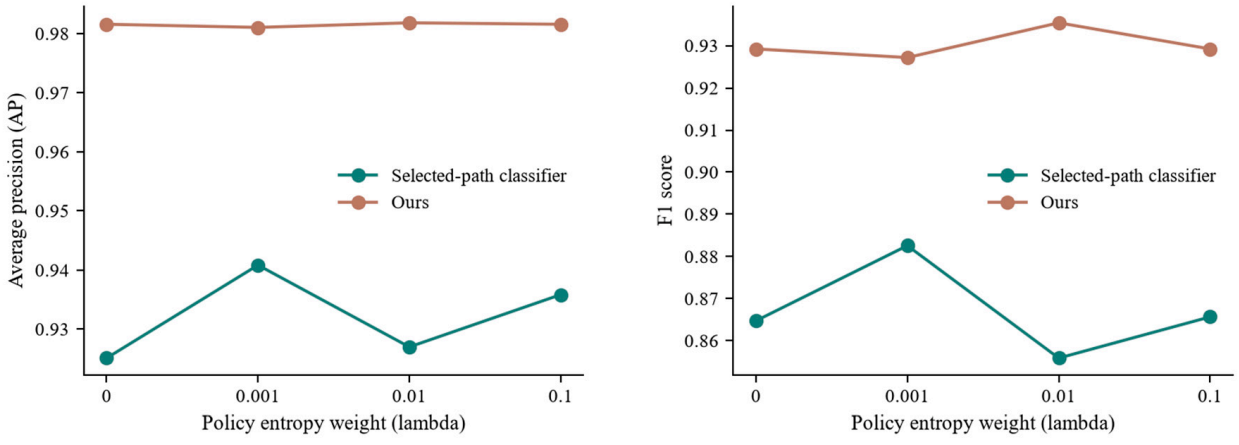
Repeated sampling reduces variation caused by the policy choosing different observations of the same group. With model weights, combination weights, and thresholds frozen, nested sets of 1, 3, 5, 10, and 20 stochastic draws produce mean final AP of 0.9647, 0.9657, 0.9660, 0.9662, and 0.9663. Final F1 remains between 0.9186 and 0.9192, and mean per-case path-score variance is approximately 0.0041 for multiple draws. The narrow F1 range shows that aggregate classification performance is maintained across these observation budgets. Increasing the number of draws from 3 to 20 adds 0.0006 AP. Thus, three draws capture most of the measured ranking benefit while using substantially fewer repeated observations, supporting the default inference budget within this comparison.



(a) Selected-path classifier: AP (b) Proposed: AP

(c) Selected-path classifier: F1 (d) Proposed: F1

Figure 6. Sensitivity to path count and maximum moves on three fixed evaluation folds. Entries summarize mean and sample standard deviation. Panels (**a**,**c**) report the selected-path classifier, and panels (**b**,**d**) the complete detector. AP and F1 use separately labeled color scales.



(a) Ranking performance (b) Classification performance

Figure 7. Sensitivity to policy entropy regularization on one fixed evaluation fold. The coefficient controls the incentive to explore alternative neighbors. Both selected-path and complete-detector predictions are evaluated with two paths and at most four moves. Panels (**a**,**b**) show AP and F1, respectively.

4.5. Representation of Difficult Subgraphs

A subgraph classifier should separate suspicious transaction groups even when their aggregate attributes resemble those of licit groups. Figure 8 investigates this property by comparing input attributes, random-walk representations, and learned-path representations for the same evaluation examples. Every point denotes one labeled subgraph, not a transaction node. Thus, the visual question is whether a representation places difficult, suspicious groups among more relevant examples of their own class.

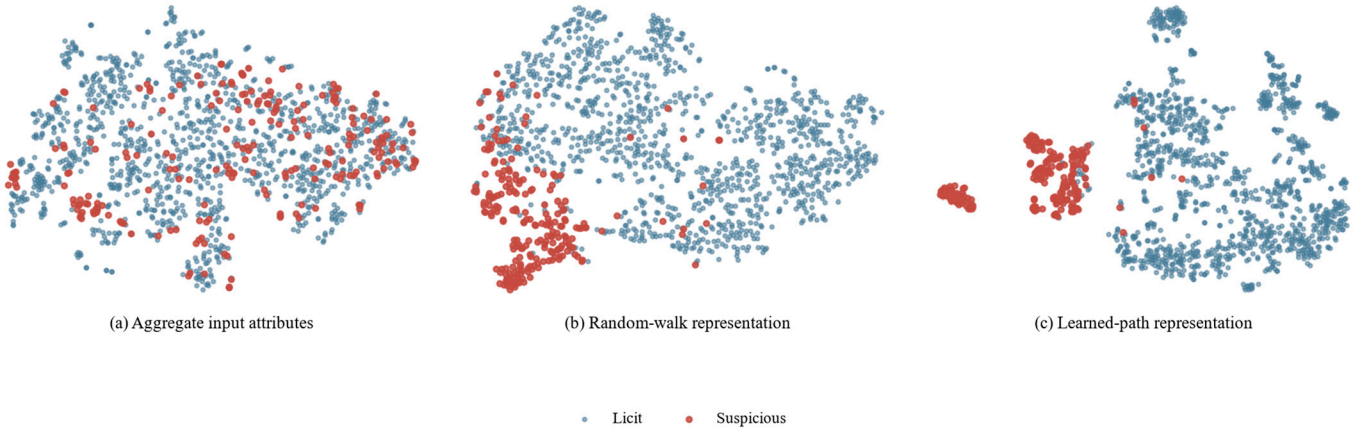


Figure 8. Representations of the same held-out transaction subgraphs. Blue points denote Licit subgraphs, and red points Suspicious subgraphs. Panels compare input attributes, random-walk representations, and learned-path representations. Each t-SNE projection is fitted independently.

The projections use t-SNE [27] with perplexity 30, 1000 iterations, PCA initialization, and a learning rate of 200. They contain all 220 suspicious examples from the selected fold and a fixed sample of 1200 licit examples to make both classes visible. Each representation is projected separately. The figure supports a comparison of neighborhood organization within each panel, rather than direct comparison of coordinates between panels or estimation of class prevalence from plotted densities.

We complement the projection with an analysis in the original representation spaces. A fixed cohort of 75 suspicious subgraphs has at least eight licit examples among its ten nearest neighbors in the input-attribute space. Table 5 follows this same cohort through the three representations. Its mean licit-neighbor share decreases from 88.8% in the input space to 29.7% in the random-walk representation and 11.1% in the learned-path representation.

Table 5. Licit-neighbor share for 75 suspicious subgraphs initially surrounded by licit examples. Neighbors are identified in each original representation space.

Representation	Licit Share Among 10 Nearest Samples (%)
Input attributes	88.8
Random-walk representation	29.7
Learned-path representation	11.1

The reduction shows that suspicious groups initially embedded among licit-looking examples become more distinctly organized after path encoding, with the learned-path representation providing the clearest separation for this cohort. This is directly relevant to subgraph classification: similar aggregate descriptions can conceal differences in connected observations.

4.6. Interpretation of Subgraph Decisions

Aggregate performance does not show which errors are corrected or how a selected observation relates to the target flow. Figures 9 and 10 examine complementary outcomes: recovering a missed suspicious subgraph and rejecting a false alarm on a licit subgraph. The accompanying tables identify the actual anonymized address clusters and show each component score relative to its validation-selected threshold. Letter labels map the displayed nodes to the recorded address clusters.

Figure 9 shows a suspicious subgraph missed by the global classifiers. Its members connect external funding participants to a receiving cluster. The selected observations examine connected participants within this arrangement, providing sequential evidence in addition to the global attribute summary. Table 6 shows that the global score of 0.6468 falls below its threshold of 0.6770, whereas the path classifier assigns 0.9999. The combined detector produces 0.6821 and classifies the group as suspicious at its threshold of 0.5944.

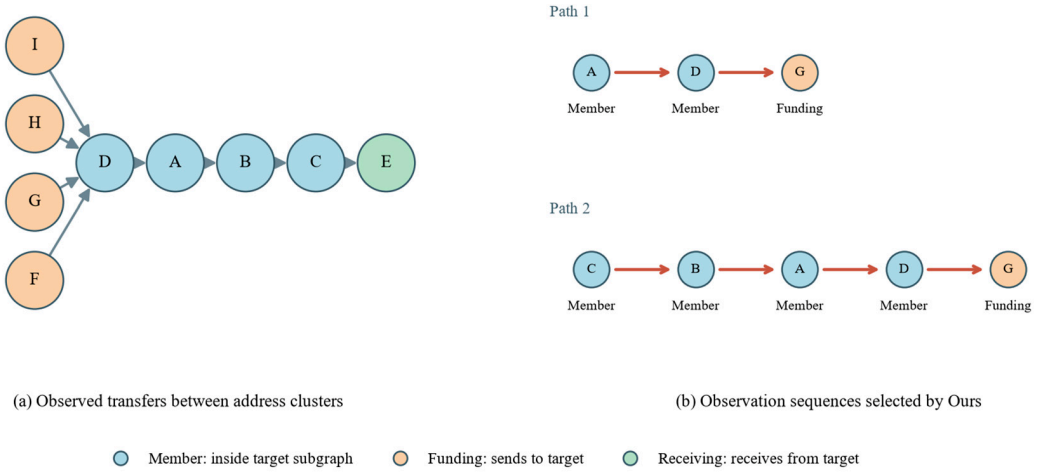


Figure 9. Recovery of a missed suspicious subgraph. Panel (**a**) shows transfers, and panel (**b**) shows the observation order, not chronological money flow. Tables 6 and 7 give scores and address-cluster identities.

Table 6. Component scores and decisions for the suspicious subgraph in Figure 9. The path prediction has weight 0.1 in the final score. Each threshold maximizes F1 on the corresponding model's inner validation set and is fixed for all evaluation subgraphs in that fold.

Model	Score	Threshold	Prediction
Global classifiers	0.6468	0.6770	Licit
Selected-path classifier	0.9999	0.9881	Suspicious
Ours	0.6821	0.5944	Suspicious

This case illustrates how a group with an inconclusive global description can become detectable when selected local observations support its suspicious label. A path weight of 0.1 combines the selected-observation score with global evidence, and the resulting score exceeds the complete detector's validation-selected threshold.

Figure 10 examines the complementary problem. The global classifiers flag a licit group whose members receive funds from several external participants and connect to a receiver. The path score is lower than the global score, reducing the combined score from 0.2937 to 0.2666 with a path weight of 0.2. At the respective validation-selected operating points, the global condition produces a false alarm, while Ours correctly predicts Licit, as reported in Table 8.

Table 7. Address-cluster identities and roles for Figure 9. Counts refer to distinct incoming and outgoing neighbors in the displayed graph.

Figure Label	Transaction Role	Address-Cluster ID	Incoming Neighbors	Outgoing Neighbors
A	Member	351849049	1	1
B	Member	376969900	1	1
C	Member	495971720	1	1
D	Member	528600837	4	1
E	Receiving	96159393	1	0
F	Funding	446726285	0	1
G	Funding	497832657	0	1
H	Funding	527191416	0	1
I	Funding	557738128	0	1

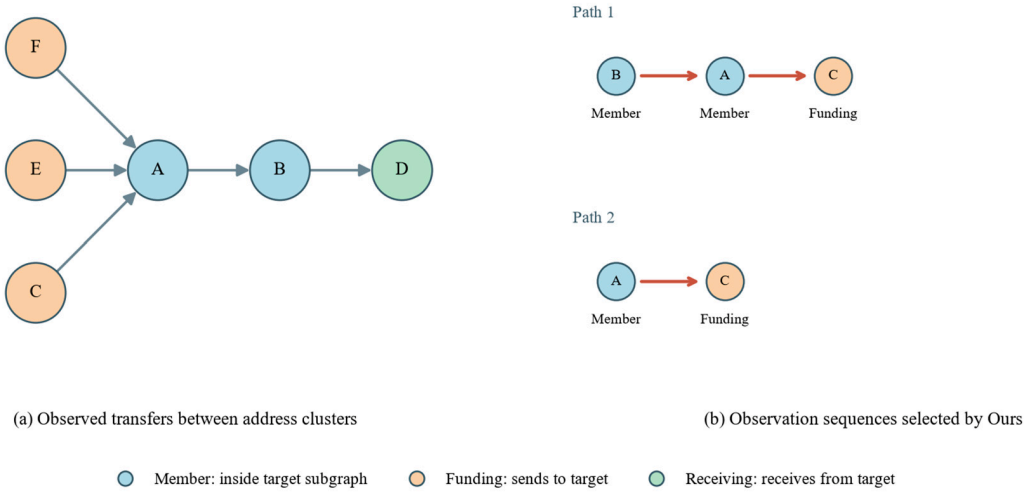


Figure 10. Correction of a false alarm. Panel (**a**) shows transfers, and panel (**b**) shows the observation order, not chronological money flow. Tables 8 and 9 give scores and address-cluster identities.

Table 8. Component scores and decisions for the licit subgraph in Figure 10. The path prediction has weight 0.2 in the final score. Each threshold maximizes F1 on the corresponding model's inner validation set and is fixed for all evaluation subgraphs in that fold.

Model	Score	Threshold	Prediction
Global classifiers	0.2937	0.2284	Suspicious
Selected-path classifier	0.1580	0.8791	Licit
Ours	0.2666	0.5245	Licit

Table 9. Address-cluster identities and roles for Figure 10. Counts refer to distinct incoming and outgoing neighbors in the displayed graph.

Figure Label	Transaction Role	Address-Cluster ID	Incoming Neighbors	Outgoing Neighbors
A	Member	764514838	3	1
B	Member	768759436	1	1
C	Funding	6242266	0	1
D	Receiving	9496379	1	0
E	Funding	428371784	0	1
F	Funding	764514841	0	1

The two cases show why local observation should support both classes. A useful policy does not merely seek observations that increase suspicion. It can also produce evidence consistent with a licit flow and help avoid unnecessary investigation. The cluster identities in Tables 7 and 9 allow the illustrated topology to be connected to the dataset records without equating a cluster identifier with a public wallet address.

Figure 11 extends the case analysis to all out-of-fold decisions. Relative to the global combination, Ours corrects nine missed suspicious subgraphs and introduces nine new misses. It also corrects 23 false alarms while introducing five, giving a net reduction of 18 false alarms. This pattern explains why mean recall is unchanged while precision and F1 improve in Table 4.

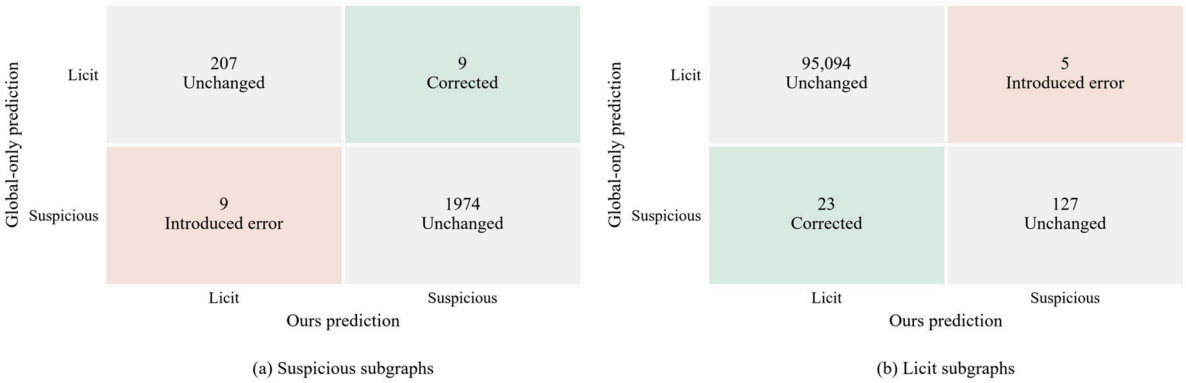


Figure 11. Changes in out-of-fold decisions after adding learned-path predictions. Rows denote the global-only decision and columns the complete-detector decision, separately for each true class. Corrected and newly introduced errors are distinguished. Green cells mark corrected errors, tan cells mark introduced errors, and gray cells mark unchanged decisions.

The selected-path score lowers the final score in ten of the 23 corrected false alarms. In the remaining thirteen, the score rises but falls below the final detector's validation-selected threshold. These corrections reflect the combined effect of the path predictions and the validation-selected decision thresholds. Representative cases were selected within each error-transition category by subgraph size, and the figures show one of the three sampled observations used for the averaged predictions.

4.7. Discussion

The complete detector benefits from selected paths, but its combination rule can also suppress a useful local prediction. Figure 12 shows a suspicious subgraph for which the path classifier predicts Suspicious with a score of 0.9994, while the global combination produces only 0.0558. With a path weight of 0.1, the final score becomes 0.1502 and remains below the final threshold of 0.6919. Tables 10 and 11 give the scores and recorded cluster identities.

Table 10. Component scores and decisions for the suspicious subgraph in Figure 12. The path prediction has weight 0.1 in the final score. Each threshold maximizes F1 on the corresponding model's inner validation set and is fixed for all evaluation subgraphs in that fold.

Model	Score	Threshold	Prediction
Global classifiers	0.0558	0.6597	Licit
Selected-path classifier	0.9994	0.9287	Suspicious
Ours	0.1502	0.6919	Licit

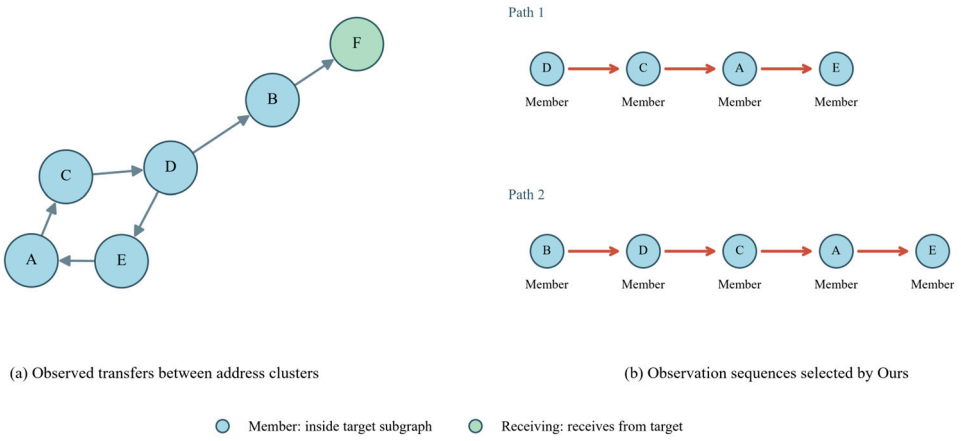


Figure 12. A persistent missed suspicious subgraph despite a positive path prediction. Panel (**a**) shows transfer relationships, and panel (**b**) shows selected observations. Tables 10 and 11 provide the predictions and recorded address clusters. The selected order is an observation path and does not reconstruct chronological money flow. Blue nodes are members and green nodes are receiving participants. Gray arrows show transfers, and red arrows show observation order.

Table 11. Address-cluster identities and roles for Figure 12. Counts refer to distinct incoming and outgoing neighbors in the displayed graph.

Figure Label	Transaction Role	Address-Cluster ID	Incoming Neighbors	Outgoing Neighbors
A	Member	480657902	1	1
B	Member	483740057	1	1
C	Member	514866775	1	1
D	Member	576718535	1	2
E	Member	831031366	1	1
F	Receiving	96159393	1	0

In this case, the fixed combination suppresses a strongly positive path prediction. Learning the interaction between traversal and the complete detector's residual errors offers a way to adapt the contribution of local evidence when it conflicts with broad summaries.

A useful score combination must both express reliable probabilities and preserve evidence that changes a group decision. We test a regularized logistic calibrator fitted to global and path probabilities using inner validation predictions only. Five-way inner cross-fitting selects its F1 threshold, after which it is fitted on the full inner set. Across ten folds, calibration reduces the mean Brier score from 0.003156 to 0.002929 and the logarithmic loss from 0.015668 to 0.012966, while mean AP changes from 0.9657 to 0.9639 and F1 from 0.9191 to 0.9172. The improved probability calibration accompanies lower AP and F1, and the illustrated group remains misclassified. These results motivate a group-dependent combination of selected observations and broad evidence when the two predictions disagree.

Calibration and ranking also differ in the original detectors. Pooling the out-of-fold scores gives AP of 0.9608 for Global and 0.9598 for the complete detector, whereas their mean fold AP values are 0.9641 and 0.9657. Pooling mixes fold-specific score scales. Pooled Brier score is 0.002947 for Global and 0.003156 for the complete detector, and the logarithmic loss is 0.013051 and 0.015668. The complete detector improves mean within-fold ranking, whereas Global provides better calibrated probabilities in this comparison. These measures distinguish prioritizing suspicious groups from estimating their absolute probabilities. Table 12 reports the corresponding confusion counts, and Figure 13 shows the precision–recall and calibration curves.

Table 12. Pooled out-of-fold confusion counts at the original inner-validation-selected thresholds. TP and FN refer to Suspicious cases, FP and TN to Licit cases.

Model	TP	FP	FN	TN
Global	1983	150	216	95,099
Proposed	1983	132	216	95,117

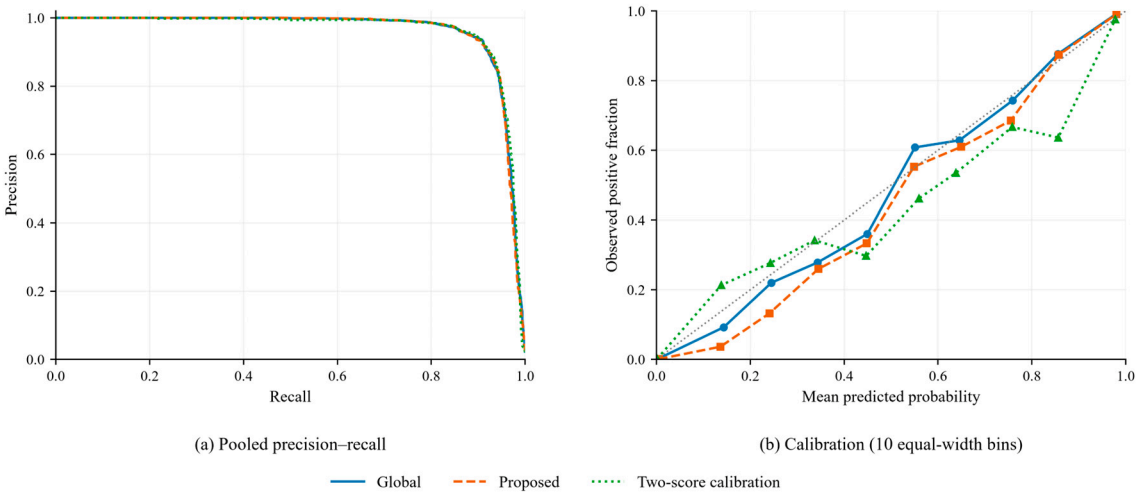


Figure 13. Pooled out-of-fold precision-recall curves and calibration for Global, Proposed, and two-score calibration. Calibration uses ten equal-width probability bins. Curves pool fold-specific scores and therefore differ from the mean fold AP used in the primary comparison. The dotted diagonal indicates perfect calibration. Line styles and markers distinguish the methods, with the legend outside the plotting areas.

The ten-fold evaluation compares detection performance across subgraph partitions. Table 13 examines recognition of new surrounding combinations: can the complete detector distinguish suspicious groups when their surrounding funding/receiving combination was absent from fitting and inner validation? It compares the complete detector with role-based Deep Sets, NeuralWalker, and RAW adapted to group classification on the single non-temporal partition described in Section 4.1. No exact funding/receiving configuration crosses its partitions. Individual participants recur, with 14,357 of the 14,372 evaluation groups containing an entity seen in fitting.

Table 13. Complete-method comparison on one fixed split of nonempty unseen funding/receiving configurations. All model selection uses the inner partition.

Model	AP	F1	Precision	Recall
Proposed	0.9733	0.9331	0.9734	0.8960
RevClassify (Deep Sets)	0.9645	0.9327	0.9798	0.8899
NeuralWalker	0.9267	0.8614	0.9355	0.7982
RAW (group adaptation)	0.9213	0.8692	0.8857	0.8532

On unseen funding/receiving configurations, the proposed detector achieves the highest AP (0.9733) and F1 (0.9331) among the four methods. RevClassify (Deep Sets), NeuralWalker, and group-adapted RAW obtain AP 0.9645, 0.9267, and 0.9213, with F1 0.9327, 0.8614, and 0.8692, respectively. Ranking is central to this imbalanced task because suspicious groups must be prioritized among many licit groups. Although Deep Sets reaches a similar F1 at its selected threshold, the higher AP supports the proposed detector's ability

to prioritize suspicious groups across operating thresholds. The complete detector combines transaction-group-conditioned observation selection with whole-group evidence and achieves higher AP and F1 than group-adapted RAW. The policy comparisons in Section 4.3 separately show that learned selection improves prediction over uniform and degree-weighted observations.

Table 14 tests whether the complete detector can prioritize suspicious groups when none of their observed participants occurred in fitting or validation. All methods are trained independently under the three fixed seeds on the same entity-disjoint partition, using the common class weighting and model-selection opportunities described in Section 4.1.

Table 14. Complete-method comparison on the fixed entity-disjoint split. Entries are means ± sample standard deviations over three independently trained seeds. All model selection uses validation data.

Model	AP	F1	Precision	Recall
Proposed	0.6734 ± 0.1073	0.5356 ± 0.2196	0.5251 ± 0.3914	0.6667 ± 0.0216
RevClassify	0.4854 ± 0.3164	0.5088 ± 0.3081	0.5948 ± 0.4398	0.5919 ± 0.0515
NeuralWalker	0.4566 ± 0.1316	0.4950 ± 0.0883	0.4517 ± 0.0896	0.5483 ± 0.0868
RAW	0.1314 ± 0.0486	0.1653 ± 0.0155	0.0972 ± 0.0097	0.5545 ± 0.0540

The proposed detector achieves the highest mean AP (0.6734) and F1 (0.5356) across the three training runs. RevClassify, NeuralWalker, and RAW obtain mean APs of 0.4854, 0.4566, and 0.1314, respectively. The proposed detector exceeds the strongest alternative by 0.1880 AP on groups whose participants are all excluded from fitting and validation. The proposed detector leads in AP in two training runs, while RevClassify leads in the third. At the validation-selected operating points, the proposed detector achieves higher mean recall, while RevClassify provides higher mean precision. The higher mean AP demonstrates more effective prioritization of suspicious groups involving previously unseen participants.

Learning which connected observations to collect improves path prediction over uniform observations. Combining the selected paths with whole-group evidence achieves the highest mean AP and F1 among the nine alternatives in the primary ten-fold comparison. At the validation-selected operating points, the complete detector reduces false alarms by 18 while retaining the same total number of detected suspicious groups. These results support using learned local observations alongside broad transaction information for suspicious-subgraph detection. Future work will examine how to preserve useful local evidence when it conflicts with the global prediction.

5. Conclusions

This paper proposed context-guided traversal with deep reinforcement learning for suspicious subgraph detection in cryptocurrency transaction graphs. The method uses role-wise transaction composition and recurrent observation history to select neighboring participants, learning from subgraph classification outcomes without annotated paths. Combining the selected-path prediction with global transaction evidence achieved mean AP of 0.9657 and F1 of 0.9191 in ten-fold evaluation on Elliptic2, exceeding the corresponding means of nine comparison methods. The representation analysis showed clearer separation of difficult suspicious subgraphs, and case analysis connected the combined detector to corrected misses and false alarms. On the entity-disjoint test, the detector also achieves the highest mean AP (0.6734) and F1 (0.5356) over three independent training seeds. These results demonstrate the value of linking transaction context to the selection of local relational evidence.

Future work will coordinate traversal learning with the errors of the complete detector, allowing local and global evidence to interact more adaptively.

Author Contributions: Conceptualization, S.-J.B. and H.-J.K. Methodology, software, formal analysis, visualization, and writing—original draft preparation, S.-J.B. Writing—review and editing, S.-J.B. and H.-J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Technology Development Program (RS-2025-02220156) funded by the Ministry of SMEs and Startups (MSS, Korea), and by the Glocal University 30 Project Fund of Gyeongsang National University in 2026.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Acknowledgments: During preparation of this study, the authors used OpenAI Codex (GPT-6 Astra) for assistance with code implementation, manuscript drafting and revision, and figure preparation. The authors reviewed and edited the outputs and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Federal Bureau of Investigation. Cryptocurrency and AI Scams Bilk Americans of Billions. 2026. Available online: https://www.fbi.gov/news/press-releases/cryptocurrency-and-ai-scams-bilk-americans-of-billions (accessed on 26 September 2026).
2. Federal Bureau of Investigation. Affidavit in Support of an Application for a Seizure Warrant. Case 3:21-mj-70945-LB, Document 1, 7 June 2021. 2021. Available online: https://www.justice.gov/opa/press-release/file/1402056/download (accessed on 25 September 2026).
3. Chen, W.; Guo, X.; Chen, Z.; Zheng, Z.; Lu, Y. Phishing scam detection on Ethereum: Towards financial security for blockchain ecosystem. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Virtual Event, 7–15 January 2021; pp. 4506–4512.
4. Buu, S.J.; Kim, H.J. Disentangled prototypical graph convolutional network for phishing scam detection in cryptocurrency transactions. *Electronics* **2023**, *12*, 4390. [CrossRef]
5. Liu, A.; Zhang, Y.; Wang, Y.; Duan, Q. IGT4ETH: An Isotropic Pre-trained Graph Transformer for Ethereum Account Classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Singapore, 20–27 January 2026; Volume 40, pp. 23631–23639.
6. Guo, C.; Zhang, S.; Zhang, P.; Alkubati, M.; Song, J. LB-GLAT: Long-term bi-graph layer attention convolutional network for anti-money laundering in transactional blockchain. *Mathematics* **2023**, *11*, 3927. [CrossRef]
7. Ouyang, S.; Bai, Q.; Feng, H.; Hu, B. Bitcoin money laundering detection via subgraph contrastive learning. *Entropy* **2024**, *26*, 211. [CrossRef] [PubMed]
8. Bellei, C.; Xu, M.; Phillips, R.; Robinson, T.; Weber, M.; Kaler, T.; Leiserson, C.E.; Chen, J. The shape of money laundering: Subgraph representation learning on the blockchain with the elliptic2 dataset. *arXiv* **2024**, arXiv:2404.19109.
9. Song, K.; Dhraief, M.A.; Xu, M.; Cai, L.; Chen, X.; Mithal, A.; Chen, J. Identifying money laundering subgraphs on the blockchain. In Proceedings of the 5th ACM International Conference on AI in Finance, Brooklyn, NY, USA, 14–17 November 2024; pp. 195–203.
10. Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.R.; Smola, A. Deep sets. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 3391–3401.
11. Bu, S.J.; Cho, S.B. Triplet-trained graph transformer with control flow graph for few-shot malware classification. *Inf. Sci.* **2023**, *649*, 119598. [CrossRef]
12. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017; pp. 2713–2726.
13. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 1024–1034.
14. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
15. Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How powerful are graph neural networks? In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
16. Egressy, B.; Von Niederhäusern, L.; Blanuša, J.; Altman, E.; Wattenhofer, R.; Atasu, K. Provably powerful graph neural networks for directed multigraphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 11838–11846.
17. Qu, D.; Ma, Y. Magnet-bn: Markov-guided Bayesian neural networks for calibrated long-horizon sequence forecasting and community tracking. *Mathematics* **2025**, *13*, 2740. [CrossRef]

18. Ma, Y.; Qu, D.; Wang, Y. Tracking evolving communities in fake news cascades using temporal graphs. *Sci. Rep.* **2026**, *16*, 4952. [CrossRef] [PubMed]
19. Akujuobi, U.; Zhang, Q.; Yufei, H.; Zhang, X. Recurrent attention walk for semi-supervised classification. In Proceedings of the 13th International Conference on Web Search and Data Mining, Houston, TX, USA, 3–7 February 2020; pp. 16–24.
20. Martinkus, K.; Papp, P.A.; Schesch, B.; Wattenhofer, R. Agent-based graph neural networks. In Proceedings of the International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
21. Kim, J.; Zaghen, O.; Suleymanzade, A.; Ryou, Y.; Hong, S. Revisiting random walks for learning on graphs. In Proceedings of the International Conference on Learning Representations, Singapore, 24–28 April 2025; pp. 82497–82547.
22. Chen, D.; Schulz, T.; Borgwardt, K. Learning long range dependencies on graphs via random walks. In Proceedings of the International Conference on Learning Representations, Singapore, 24–28 April 2025; pp. 96443–96469.
23. Mathys, J.; Errica, F. Learn to Jump: Adaptive Random Walks for Long-Range Propagation through Graph Hierarchies. *arXiv* **2025**, arXiv:2509.01381.
24. Cho, K.; Van Merriënboer, B.; Gulçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
25. Williams, R.J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **1992**, *8*, 229–256. [CrossRef]
26. Nadeau, C.; Bengio, Y. Inference for the generalization error. *Adv. Neural Inf. Process. Syst.* **1999**, *12*, 307–313.
27. Van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.