PERISCOPE: A SINGLE BANK'S VIEW INTO CROSS-INSTITUTIONAL LAUNDERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph neural networks (GNN) have become the dominant approach to anti-money laundering (AML), yet progress is held back by the scarcity of public data that reflect how banking systems actually operate. Existing datasets typically compress banking records into account-to-account transfer graphs, discarding the ownership ties and cross-entity relations on which practical risk analysis relies. This mismatch restricts both the credibility of public evaluation and the development of models able to exploit richer financial structure. We present PERISCOPE, a simulation-based AML testbed grounded in real banking data. Unlike prior public datasets, PERISCOPE retains the complete customer, account, and transaction schema of a single-bank operational setting, including ownership relations over both internal and external accounts, together with fine-grained transaction-level timestamps. We document the construction pipeline, assess the fidelity of the simulation, and provide a detailed characterization of anomalous users and accounts. Our analysis shows that associations among external entities reveal additional structural links between suspicious multi-account users that remain invisible in typical AML scenarios. Building on PERISCOPE, we conduct a systematic empirical study at both the user and the account level. The results show that retaining the richer structure consistently improves AML performance over simplified transfer-graph constructions. Meanwhile, current leading specialized AML models bring only limited gains over generic graph baselines and scale poorly on our dense data. These findings indicate that the main bottleneck of public AML research lies not only in data scarcity but also in the overly simplified graph abstractions adopted by existing datasets. Overall, PERISCOPE offers a grounded single-bank AML testbed, together with concrete evidence on how customer, account, and transaction structure and external-entity associations advance graph-based AML evaluation and modeling.

1 INTRODUCTION

Money laundering hides illicit proceeds through chains of seemingly legitimate transactions, making detection increasingly difficult in large, complex, and fast evolving financial systems (Deprez et al., 2025). Graph neural networks (GNNs) have become a promising approach for anti-money laundering (AML), since financial transactions can be naturally modeled as graphs over accounts and other entities (Johannessen & Jullum, 2025). Yet progress is still limited by the lack of realistic public benchmarks: due to privacy constraints, many studies rely on private data, while existing public AML datasets typically reduce banking records to simple account-to-account transfer graphs and omit the rich auxiliary information available in real systems. This gap weakens openness, reproducibility, and fair comparison, and may lead current methods to underestimate the complexity of real-world AML detection (Chen et al., 2026).

Moreover, existing public benchmarks remain misaligned with how AML is conducted in practice. A single bank observes two fundamentally different types of accounts: internal accounts, for which it has rich know your customer (KYC) information and ownership links, and external accounts from other institutions, whose associations are largely hidden behind clearing systems. This asymmetry is crucial, since recovering relations among external accounts can provide strong signals for cross-institution laundering. We explicitly demonstrate the significance in Figure 1. However, current datasets and methods typically assume an overly simplified or globally observable setting (Tian et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

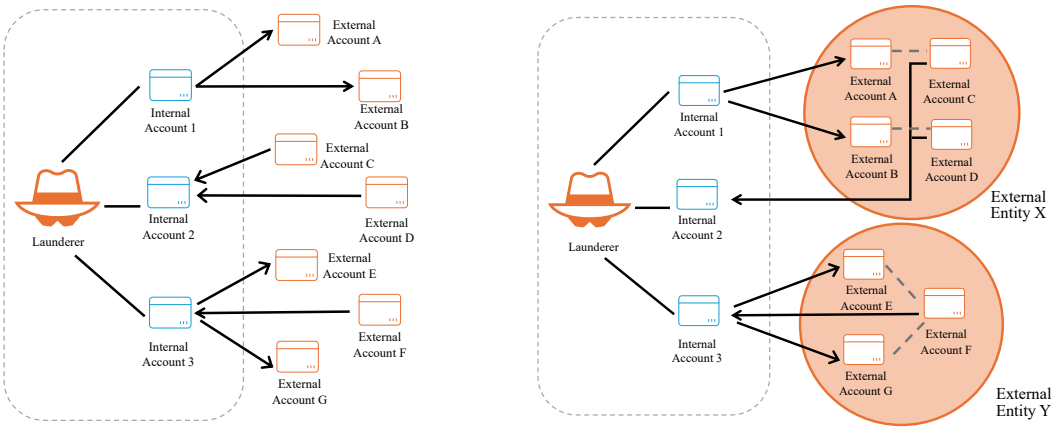


Figure 1: Example comparison of anti-money laundering analysis from the perspective of a single bank. The gray dashed line indicate possible transfer paths between external entity accounts to enhance case clarity. (a) Without modeling external entities, accounts outside the bank are treated as isolated, which breaks the transaction chain and limits observability of cross-institutional laundering patterns. (b) Incorporating external-entity information closes the transaction chain, enabling more complete detection of laundering behaviors.

2025), and often collapse the data into homogeneous transaction graphs over accounts or customers only. In doing so, they discard the heterogeneous entities and relations that are central to real-world AML detection.

Taken together, these limitations suggest that the key bottleneck in public AML research is not only limited data access, but also the overly simplified way public benchmarks represent banking systems. By collapsing customer, account, and transaction structure into homogeneous graphs, existing public AML datasets discard ownership and external-entity relations that are critical to single-bank AML, underscoring the need for more realistic heterogeneous benchmarks. To this end, we present PERISCOPE, a simulation-based benchmark grounded in real banking data that preserves internal and external ownership structure together with transaction-level timestamps. We further characterize its structural and behavioral properties, showing that recovered external-account associations reveal additional signals for suspicious multi-account activity. Experiments at both the user and account levels show that preserving heterogeneous structure improves detection, heterogeneous GNNs are better suited to exploit it, and current AML-specific homogeneous models, in previous benchmarking settings, still leave substantial room for improvement under realistic banking settings.

Our contributions are as follows:

- **A realistic heterogeneous benchmark for single-bank AML.** We release a simulation-based AML dataset that better approximates selected structural aspects of single-bank operations than existing public benchmarks by explicitly modeling the customer–account ownership structure for both internal and external accounts, together with fine-grained transaction timestamps. By better capturing the structural complexity of real single-bank monitoring, the resulting benchmark not only supports more realistic empirical evaluation of AML models, but also provides a valuable testbed for exploring money-laundering behaviors and relational patterns in banking graphs.
- **Domain-informed insights into AML graph construction and modeling.** Using this benchmark, we derive practical insights into how banking-specific structure affects AML detection. In particular, our analysis shows that preserving customer–account–transaction relations and recovering external-account associations provide useful signals that are weakened in simplified graph abstractions, providing evidence that the data reflects partial money-laundering patterns, and offering empirical guidance for designing more realistic and effective graph-based AML systems.

2 Related Works

AML GNNs & benchmarks GNNs have become a leading approach for AML because financial transactions are naturally relational. Prior work has adapted generic and AML-specific GNNs to transaction graphs to address challenges such as scale, class imbalance, heterophily, and temporal dynamics (Kipf, 2016; Veličković et al., 2017; Wu et al., 2022; Cardoso et al., 2022; Cheng et al., 2023; Zheng et al., 2023; Lin et al., 2024). Mainstream GNN methods typically model homogeneous networks among real bank accounts or entities, or transaction-wallet networks in Bitcoin transaction networks, but each has limitations when applied to user-user networks. The former will lose additional heterogeneous information, while the latter, by assuming the network as an undirected relational graph, cannot handle transaction multiplicity, namely multiple transactions between two accounts. Also, current progress remains limited by public benchmarks, which typically represent banking data as single-relation account graphs, as in T-Finance (Tang et al., 2022), AMLWorld (Altman et al., 2023), TransXion (Chen et al., 2026). This abstraction discards ownership structure and other heterogeneous relations central to real banking systems. Even datasets used for heterogeneous graph modeling, such as Elliptic (Weber et al., 2019), capture only limited heterogeneity and miss latent cross-account associations that are often important for laundering detection. As a result, existing open benchmarks provide only a simplified view of AML, leaving realistic heterogeneous modeling largely unexplored.

3 Proposed Dataset

We introduce PERISCOPE, a heterogeneous transaction dataset designed for AML research. In contrast to existing public AML datasets that reduce financial activity to a homogeneous transaction graph, PERISCOPE preserves the heterogeneous relational structure intrinsic to real banking systems. Specifically, it models multiple entity types, including internal users, internal accounts, and their external counterparts, together with associated ownership and transaction relations. We discuss the ethics of our dataset in Section 7.

3.1 Raw Data

The original data are stored in relational tables centered on transactions between in-bank accounts and counterparties, which may belong to external institutions and are often routed through third-party settlement intermediaries. For in-bank accounts, reliable KYC records allow us to recover user–account ownership relations; for counterparties, anonymous external-entity identifiers provided by settlement institutions enable partial linkage to external entities. The data authority ensures **external entities and in-bank users occupy disjoint roles**, as the two sides originate from different parties. Although such cross-institution associations are difficult to access and often only partially observable in practice, they provide valuable signal for AML modeling. Labels are derived from real investigations conducted jointly by the bank and regulators, identifying accounts confirmed or highly suspected to be involved in money laundering.

3.2 Dataset Construction

Heterogeneous transaction graph modeling. We convert the raw relational records into a heterogeneous transaction graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (formally defined in Appendix A.1) with four node types, including in-bank users, in-bank accounts, counterparty entities, and counterparty accounts, and two relation types. This schema reflects common laundering behavior: entities may control multiple accounts across institutions, fund flows occur at the account level, and risk is often attributed at the entity level. Ownership edges link users or entities to accounts and are modeled as bidirectional, while transaction edges represent directed transfers between in-bank and counterparty accounts, with edge direction following fund flow and a normalized timestamp preserving temporal order. Each transaction edge carries numerical and categorical attributes, which are preprocessed separately: numerical features are standardized for privacy protection and stable simulation, and categorical features are encoded by semantic group. The resulting dataset is organized into node tables, edge tables, and transaction attribute tables.

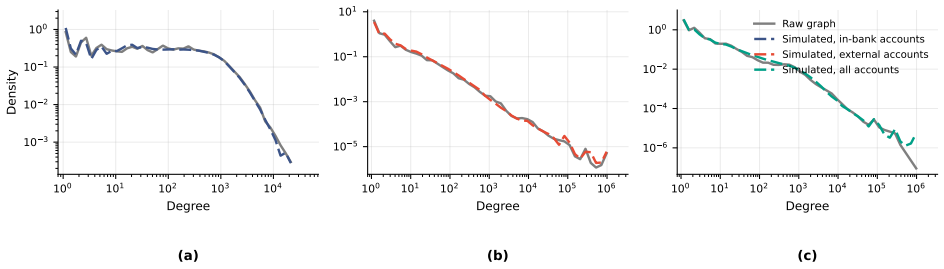


Figure 2: Degree distributions of the raw and simulated graphs. (a) In-bank accounts, (b) external accounts, (c) all accounts; transaction-degree distributions over log-spaced bins with exponential smoothing.

Table 1: Comparison experiment on the bank internal dataset. We repeat the same user–account edge ablation as in Table 4. Results are reported in terms of AUPRC and F1 (%, mean±std over 5 runs). Bold denotes the best result within each metric column.

Task	User				Account			
Setting	Original		w/o		Original		w/o	
Metrics	AUPRC	F1	AUPRC	F1	AUPRC	F1	AUPRC	F1
MLP	48.26±0.41	27.47±2.21	56.12±0.52	39.36±3.88	48.71±0.55	27.18±0.74	47.84±0.83	27.91±2.16
GIN	60.21±1.12	35.02±5.46	63.48±0.43	41.76±2.04	41.10±0.36	20.41±1.29	49.37±0.71	**30.82±3.22**
HGIN	**67.84±0.19**	**43.85±3.62**	**64.21±0.58**	**43.78±2.57**	**54.42±0.61**	**35.06±3.71**	**52.38±0.45**	28.16±2.04
RGCN	54.26±0.34	36.08±4.74	59.12±0.18	23.05±1.63	50.36±0.92	31.22±1.18	47.76±0.49	27.51±3.68

Simulation-based data generation. To produce a releasable dataset while preserving the key characteristics of the original transaction system, we adopt the deep graph simulation framework of (Darabi et al., 2025). The simulator takes as input the heterogeneous topology, transaction features, and counterparty categories, jointly learns the distributions over graph structure, relation types, transaction attributes, and entity categories, and then samples a synthetic heterogeneous transaction graph to replace the original one.

We further validate the fidelity of the simulated graph against the original transaction system. In Figure 2 and Appendix A.2, PERISCOPE shows high consistency with the real dataset across multiple statistical feature distributions of the two datasets, with only minor and acceptable deviations. Furthermore, comparing Table 1 and Table 4, the trends and magnitudes of the related results are largely consistent, indicating a high degree of comparability between the two sets of experimental outcomes. For more details on framework selection, method settings, and further analysis of results, see Appendix A.2.

4 OVERVIEW OF PERISCOPE

This section presents a preliminary analysis of PERISCOPE to clarify its structural properties and motivate subsequent model design. Specifically, we first describe the overall heterogeneous graph schema and key dataset statistics. We then examine the differences between normal and anomalous accounts from both structural and transactional perspectives. Finally, we move beyond basic characterization and provide a domain-informed analysis of suspicious users with multiple suspicious accounts, through which we uncover valuable insights into money-laundering transaction patterns and cross-account coordination behaviors, further demonstrating the practical value and research utility of PERISCOPE.

4.1 OVERALL

We present the statistical data of the specific types of graph nodes and edge types in Table 2. We further compare PERISCOPE with representative AML benchmarks and closely related financial graph anomaly detection datasets in Table 3, highlighting its distinctive structural properties. (1) PERISCOPE exhibits a structural pattern characteristic of real banking transaction systems: the overall

Table 2: Statistics of node and edge types in the graph. "AP" means "Anomaly Proportion". Node-type identifiers follow the released data schema; `inner_user` denotes in-bank users.

Category	Type / Pattern	Description	Count
Node	inner_user	Internal user (Anomaly 4,606, AP 10.2%)	44,971
	in_account	Internal account (Anomaly 6,054, AP 11.3%)	53,742
	outer_entity	External entity	645,168
	out_account	External account	1,036,705
Edge	inner_user ↔ in_account	Internal user owns internal account	53,742
	outer_entity ↔ out_account	External entity owns external account	1,036,705
	in_account → out_account	Internal account transfers to external account	8,444,019
	out_account → in_account	External account transfers to internal account	1,842,604

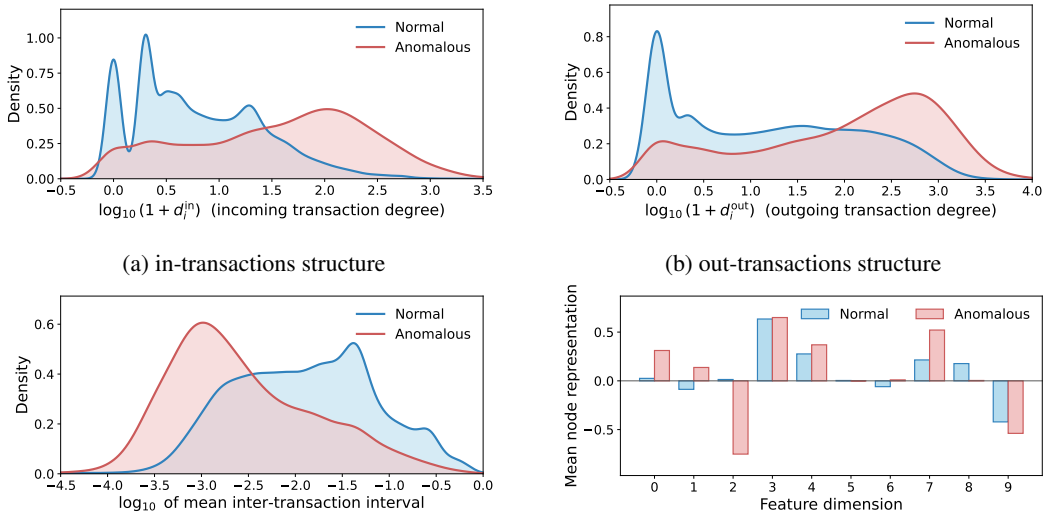
Table 3: Comparison with representative AML and financial graph anomaly detection benchmarks. "MD" means "Mean Degrees", "MDTN" means "Mean Degrees per Target Node" and "AP" means "Anomaly Proportion". DGraph and T-Finance are **real** fraud detection datasets of non-bank financial institutions; Elliptic++ is derived from the **real** public Bitcoin ledger, covering both transactions and wallet addresses; AMLWorld is a completely **simulator-generated** dataset of bank money laundering (we select HI versions). TransXion is the **simulator-generated** benchmark based on realistic individual profiles. The dataset we released is a learning **simulation dataset based on real data**.

Dataset	# nodes	# edges	# task	# labels	MD	MDTN	AP %
DGraph (Huang et al., 2022)	3.7M	4300K	Node	1225K	1.16	3.5	1.3
T-Finance (Tang et al., 2022)	39K	21M	Node	All	539	539	4.6
Elliptic++ (Elmougy & Liu, 2023)	1.03M	3.1M	Node	312K	3.02	10.0	6.0
AMLWorld-Small (Altman et al., 2023)	515K	5M	Edge	All	9.7	9.7	0.1
AMLWorld-Large (Altman et al., 2023)	2.06M	176M	Edge	All	85.4	85.4	0.1
TransXion (Chen et al., 2026)	47K	3M	Edge	All	63.7	63.7	0.2
PERISCOPE(Ours)	**1.7M**	**12.5M**	**Node**	**53742**	**7.35**	**233**	**11.3**

graph has a relatively low average degree due to the large number of external nodes and ownership relations, yet the labeled in-bank accounts are highly connected because they concentrate most observable transaction activity. As a result, the graph is **globally sparse** but **locally dense** around the prediction targets, a property rarely captured by existing AML benchmarks. (2) Compared with conventional benchmarks, we adopt a broader criterion for labeling laundering-related account behaviors. This relaxed definition expands the coverage of anomalous behaviors, allowing the benchmark to capture not only highly canonical laundering patterns but also more ambiguous yet practically alert-worthy suspicious activities, such as sudden large transfers or an unusually high number of incoming payments, intended to reflect broader risk-screening scenarios. Combined with its explicit heterogeneous schema over users, accounts, and external entities, this makes PERISCOPE a challenging and realistic benchmark for AML research. In addition, we derive user-level labels for in-bank users by marking a user as anomalous whenever any associated in-bank account is anomalous, enabling both **account-level** and **user-level** risk modeling.

4.2 COMPARISON ON ANOMALOUS AND NORMAL

Anomalous accounts have different graph structures and neighbor edge features from normal accounts. As in Figure 3, the distribution of node degrees in (a) and (b) both show that anomalous accounts tend to have higher node degrees, indicating that the topological structure of the graph is crucial in money laundering identification. (c) indicates that anomalous accounts tend to initiate transactions at a lower mean time span compared to normal accounts. (d) indicates that anomalous accounts and normal accounts show significant differences in multiple distinct characteristic dimensions. Therefore, we generate node representations based on average pooling, and we ensure that the pooling is performed after data splitting to prevent future information leakage.



(a) in-transactions structure

(b) out-transactions structure

(c) mean inter-transaction interval

(d) Neighbors' transaction features

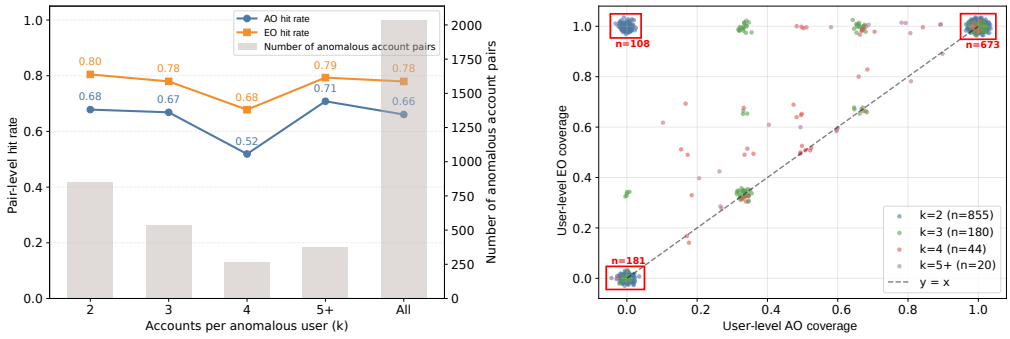
Figure 3: Statistics distribution of normal and anomalous internal accounts. (a) shows the probability-density distribution of each account's incoming transaction degree d_i^{in} on a $\log_{10}(1 + d_i^{\text{in}})$ scale, (b) shows the same for the outgoing transaction degree d_i^{out}, (c) shows the probability density of each account's mean inter-transaction interval, i.e., the average time gap between consecutive transactions of the account, where all incoming and outgoing transactions are ordered chronologically, plotted on a $\log_{10}$ scale, and (d) shows the class-wise mean of each account's average-pooled neighbor transaction feature vector over the feature dimensions.

4.3 Analysis on suspicious multi-account users

For suspicious users associated with multiple suspicious accounts, we construct pairs of suspicious accounts under the same user and define two overlap relations. *Account-level Overlap* (**AO**) holds if two suspicious accounts have both transacted with the same external account. *Entity-level Overlap* (**EO**) holds if they have transacted with different or identical external accounts belonging to the same external entity. By definition, AO is a special case of EO: AO implies EO, while EO does not necessarily imply AO. Therefore, EO generalizes direct counterparty sharing from the account level to the entity level.

This distinction is closely related to money laundering strategies. AO mainly captures explicit account reuse, which is common in relatively direct *fan-in* or *fan-out* patterns, where multiple suspicious accounts concentrate funds into, or distribute funds through, the same counterparty account. EO, in contrast, captures a more concealed form of coordination: suspicious accounts interact with different accounts controlled by the same external entity. Such behavior is more consistent with *scatter*, entity-level *fan-in/fan-out*, and layered *cycle* patterns, in which launderers deliberately diversify counterparty accounts while preserving the same underlying transactional target. In practice, this diversification can weaken simple account-based rules, since no single external account appears repeatedly involved. Hence, the increment from AO to EO reflects hidden structural dependencies that cannot be observed from account-level transactions alone.

Figure 4 provides clear empirical evidence for this difference. In Figure 4(a), suspicious users with multiple suspicious accounts show substantial overlap not only at the account level but more prominently at the entity level, and the increase from AO to EO remains significant across users with different numbers of accounts. Figure 4(b) further shows that EO markedly improves the coverage between suspicious accounts. Together, these results indicate that external entity information reveals a considerable number of otherwise hidden connections, thereby substantially enhancing the ability of PERISCOPE to capture realistic multi-account laundering behaviors. These observations also suggest that entity-aware signals provide complementary evidence beyond direct transaction sharing. We further analyze these account pairs in the Appendix B.



(a) (b)

Figure 4: Visualization of AO and EO analysis. (a) Pair-level hit rate, defined as the fraction of suspicious account pairs satisfying AO or EO, grouped by the number of suspicious accounts associated with each user; "All" denotes the aggregate over all users. (b) User-level coverage, defined for each suspicious multi-account user as the fraction of all possible suspicious-account pairs covered by AO or EO. Red boxes mark users at $(0, 0)$, $(0, 1)$, and $(1, 1)$, respectively. Points above the diagonal indicate users for whom EO covers additional account pairs beyond AO. Jitter is added for visual clarity.

5 Experiments on Dataset

In this section, we conducted further experiments on PERISCOPE around three questions (For more details on the experiment's conduct, please refer to the Appendix C). ***Q1***: Does preserving heterogeneous structure improve AML detection compared with simplified homogeneous graph constructions? ***Q2***: Which model families are better suited to heterogeneous and simplified graph constructions? ***Q3***: How do representative AML-specific homogeneous models under previous conventional settings perform on PERISCOPE?

5.1 Heterogeneous Graph Construction and Model Selection on PERISCOPE(*Q1&Q2*)

Setup To answer ***Q1*** and ***Q2***, we compare heterogeneous and homogeneous graph constructions on PERISCOPE through controlled ablations at both the user and account levels. For the user-level task, we evaluate the full heterogeneous graph against a variant that removes all `in_account` and `out_account` edges and directly reassigns transactions to `inner_user` and `outer_entity`. For the account-level task, we further remove user–account edges together with user nodes. To study model selection under the two graph constructions, we consider three model families: an MLP (Rumelhart et al., 1986) without message passing; homogeneous GNNs, including GCN (Kipf, 2016), GAT (Veličković et al., 2017), and GIN (Xu et al., 2018); their heterogeneous extensions HGCN, HGAT, and HGIN with relation-specific message passing; and relational heterogeneous models RGCN (Schlichtkrull et al., 2018) and RGAT (Busbridge et al., 2019). This design directly tests whether heterogeneous construction is preferable for AML and which model class is best suited to each graph type.

Discussion Table 4 provides a direct answer to ***Q1***: for AML on PERISCOPE, the heterogeneous construction is preferable because the best performance achieved on the original heterogeneous graph is higher than the best performance achieved on the simplified homogeneous variant for both tasks. This indicates that preserving user–account relations and the underlying heterogeneity leads to stronger predictive performance than collapsing the graph structure.

For ***Q2***, the results suggest that model selection should be aligned with graph construction rather than based on model complexity alone. On the original heterogeneous graph, heterogeneous GNNs universally outperform the MLP baseline and standard homogeneous GNNs, showing that relation-aware message passing is the most suitable inductive bias when user, account, and transaction semantics are explicitly preserved. In contrast, after the graph is simplified, the advantage of heterogeneous models becomes less stable and several homogeneous models become competitive,

Table 4: Ablation study on the effectiveness of user–account edges for simultaneous user- and account-level verification. For the user-level task, *w/o* removes all `in_account` and `out_account` edges and directly maps transactions to `inner_user` and `outer_entity`; for the account-level task, *w/o* removes user nodes instead. Results are reported in terms of AUPRC and F1 (%, mean±std over 5 runs) across MLP, homogeneous GNNs, relation-aware heterogeneous variants, and relational graph models in chunks respectively. Darker green indicates better performance.

Task	User				Account			
Setting	Original		w/o		Original		w/o	
Metrics	AUPRC	F1	AUPRC	F1	AUPRC	F1	AUPRC	F1
MLP	47.50±0.29	28.13±3.84	55.43±0.59	38.52±4.14	47.95±0.61	34.63±0.59	47.23±0.99	27.37±2.33
GCN	53.72±1.05	33.83±3.74	46.74±0.49	22.51±6.17	50.48±0.72	26.97±3.97	46.86±1.04	30.55±1.30
GIN	59.39±1.45	33.91±6.53	62.99±0.37	41.04±1.93	43.29±0.28	19.36±1.47	50.84±0.83	30.15±3.86
GAT	45.16±1.50	19.93±5.84	37.29±1.26	21.24±0.62	40.85±1.06	11.69±5.10	41.41±0.53	16.32±0.91
HGCN	62.76±0.10	36.52±1.85	58.71±0.44	24.51±3.93	54.36±0.63	36.73±2.48	48.88±0.47	24.27±4.80
HGAT	64.16±0.76	47.29±2.43	60.93±0.13	26.14±2.41	53.30±0.62	38.63±0.59	49.04±0.40	23.48±2.22
HGIN	66.93±0.13	42.94±4.13	63.55±0.61	40.80±2.83	53.65±0.68	34.12±4.30	49.90±0.39	23.44±2.15
RGCN	53.61±0.29	35.35±7.32	58.53±0.12	22.14±1.86	47.62±1.02	30.55±1.30	47.11±0.54	23.88±4.06
RGAT	54.43±0.36	26.75±2.92	59.96±0.16	23.48±2.01	49.79±1.12	29.64±2.34	45.89±0.47	22.94±3.87

especially on F1 metrics. This means that the best model is not universal: heterogeneous graph construction favors heterogeneous architectures, while simplified graph construction reduces the value of relation-specific design.

A further observation is that the two tasks do not benefit equally from heterogeneity. The user-level task is more sensitive to removing user–account links, suggesting that user representations depend more heavily on cross-type interactions for gathering informative context. The account-level task is relatively more robust, likely because account nodes already encode substantial transactional information. Overall, these findings jointly support the answers to ***Q1*** and ***Q2***: heterogeneous graph construction is more appropriate for AML on PERISCOPE, and heterogeneous GNNs are the most suitable model family when that structure is preserved.

5.2 PERFORMANCE OF THE AML MODELS(***Q3***)

Setup To answer *Q3*, we benchmark representative AML models in the transaction-only regime, which is the common setting assumed by most existing homogeneous AML methods. Like Section 5.1, we evaluate both user-level suspicious-user detection and account-level detection under this setting. Our comparison includes three representative AML-specific models, LineMVGNN (Poon et al., 2025), MEGA-GNN (Bilgi et al., 2024), and DGA-GNN (Duan et al., 2024) based on reproducibility, together with widely used homogeneous graph baselines, namely Node2Vec (Grover & Leskovec, 2016), GraphSAGE (Hamilton et al., 2017), and PNA (Corso et al., 2020). Note that we do not consider using heterogeneous AML-specific models, as these models are specifically designed for the Bitcoin network rather than real bank data, and we explain the reasons in § 2.

Discussion Table 5 shows that AML-specific models consistently outperform general-purpose homogeneous baselines on both user- and account-level tasks, validating the benefit of architectures tailored to AML patterns. At the same time, scalability remains a major practical constraint: LineMVGNN runs out of memory on our dense dataset due to the cost of line-graph construction. More importantly, even the strongest homogeneous AML models achieve performance that is only on par with heterogeneous baselines rather than establishing a clear advantage, indicating substantial room for improvement.

6 CONCLUSION & LIMITATIONS

We introduced PERISCOPE, a simulation-based heterogeneous benchmark designed to better reflect real-world single-bank AML settings. Unlike existing public AML datasets that collapse financial ac-

Table 5: Comparison with homogeneous graph baselines under the transaction-only setting. We report AUPRC and F1 (%, mean±std over 5 runs) for both user-level and account-level classification. The upper block lists commonly used general GNN benchmarks, while the lower block reports leading AML specific models. Best results within each metric are highlighted in **bold**; "OOM" denotes out-of-memory failure on a single 32 GB GPU.

Task	User		Account	
Metrics	AUPRC	F1	AUPRC	F1
Node2Vec	39.86±1.42	24.73±4.05	41.24±1.13	18.47±3.62
SAGE	51.27±0.83	26.18±2.45	47.32±0.69	29.84±2.13
PNA	56.49±0.71	28.62±1.93	50.78±0.57	26.91±1.78
LineMVGNN	OOM	OOM	OOM	OOM
MEGA-GNN	**62.18±0.61**	38.81±1.52	**53.14±0.55**	28.46±1.36
DGA-GNN	60.74±0.78	**37.46±1.38**	52.07±0.61	**33.52±1.21**

tivity into homogeneous transaction graphs, PERISCOPE preserves user–account ownership structure, external entity associations, and transaction-level temporal information, thereby exposing relational signals that are typically discarded in prior benchmarks. Our empirical analysis shows that these heterogeneous signals are not merely descriptive but operationally valuable: anomalous behavior exhibits distinctive structural and transactional patterns, and recovering external-entity associations reveals hidden links among suspicious multi-account users that are invisible at the account level alone. Building on this benchmark, our experiments further show that preserving heterogeneous structure improves the best observed performance in our benchmark, while current leading homogeneous AML models, although stronger than generic baselines, remain limited in both scalability and accuracy gains on this dataset. In summary, these results suggest that the main bottleneck in public AML research is not only the lack of data, but also the overly simplified way in which data are commonly represented.

However, this work has several limitations. PERISCOPE is grounded in real banking data but remains simulation-based, and thus cannot fully capture institutional variation or long-term behavioral evolution in real AML environments. Moreover, our experiments provide an initial benchmark rather than a comprehensive study of model design, optimization, and deployment. Finally, we have not fully disclosed our dataset creation process or the sources and specific contents of the original datasets, in order to prevent malicious individuals from misusing this data for improper purposes in this field or against specific banks. However, we emphasize that we have made every effort to verify the authenticity of the data generation process and the availability of the data, However, we emphasize that we have made every effort to verify the authenticity of the data generation process and the availability of the data, including a comparative analysis of the data production procedures and methods, as well as relevant experimental evidence and statistical characteristics.

7 ETHICS

This work advances AML research by improving realism, openness, and reproducibility through a simulation-based heterogeneous benchmark. We believe it has clear societal value: more realistic AML models can help financial institutions and regulators better identify suspicious fund flows and support reliable risk assessment. Meanwhile, AML research and data release require careful ethical consideration, as improved detection methods may also help adversaries infer monitoring logic, and banking-derived data may raise privacy, compliance, and security concerns.

To mitigate these risks, our released benchmark is fully simulation-based and contains no raw transaction records, personally identifiable information, customer attributes, or directly recoverable information about the underlying institution. We do not release the original banking data, participating institution identities, or full synthetic-data generation details and code, following prior practice such as T-Finance (Tang et al., 2022) and AMLworld (Altman et al., 2023). These are withheld for privacy, compliance, confidentiality, and security reasons, and to reduce misuse risks. Instead, we provide a high-level generation description, the simulated dataset, and partial experimental code, balancing scientific utility and benchmark reproducibility with responsible risk control.

AI USE STATEMENT

LLMs were used to assist in drafting and organizing this manuscript. All numerical results were produced by the documented experiment pipeline. The authors have reviewed all AI-assisted content and take full responsibility for the final content of this work. No LLM is listed as an author.

REPRODUCIBILITY STATEMENT

The full experimental protocol, including dataset splits, evaluation metrics, hyperparameter search, training strategy, and hardware environment, is described in Appendix C. Further details on the experimental environment and result monitoring are provided in the supplementary materials. The dataset is really large, so we provide **an anonymous link** here: `https://dataverse.harvard.edu/previewurl.xhtml?token=a763404c-80c5-42d1-8a85-9d9be6ac9426` for readers to access. The uploaded dataset and code can reproduce the main experimental results, but not the complete data simulation process. We focused on the reasons in the Ethics section.

REPRODUCIBILITY STATEMENT

Given the particular importance of ethics in this paper, we have decided to address it directly within the main text.

REFERENCES

Erik Altman, Jovan Blanuša, Luc Von Niederhäusern, Béni Egressy, Andreea Anghel, and Kubilay Atasu. Realistic synthetic financial transactions for anti-money laundering models. *Advances in Neural Information Processing Systems*, 36:29851–29874, 2023.

H Çağrı Bilgi, Lydia Y Chen, and Kubilay Atasu. Multigraph message passing with bi-directional multi-edge aggregations. *arXiv preprint arXiv:2412.00241*, 2024.

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. Relational graph attention networks. *arXiv preprint arXiv:1904.05811*, 2019.

Mário Cardoso, Pedro Saleiro, and Pedro Bizarro. Laundrograph: self-supervised graph representation learning for anti-money laundering. In *Proceedings of the Third ACM International Conference on AI in Finance*, pp. 130–138, 2022.

Keyang Chen, Mingxuan Jiang, Yongsheng Zhao, Zeping Li, Zaiyuan Chen, Weiqi Luo, Zhixin Li, Sen Liu, Yinan Jing, Guangnan Ye, et al. Transxion: A high-fidelity graph benchmark for realistic anti-money laundering. *arXiv preprint arXiv:2604.17420*, 2026.

Dawei Cheng, Yujia Ye, Sheng Xiang, Zhenwei Ma, Ying Zhang, and Changjun Jiang. Anti-money laundering by group-aware deep graph learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12444–12457, 2023.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems*, 33: 13260–13271, 2020.

Sajad Darabi, Piotr Bigaj, Dawid Majchrowski, Artur Kasymov, Pawel Morkisz, and Alex Fit-Florea. A framework for large-scale synthetic graph dataset generation. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

Bruno Deprez, Wei Wei, Bart Baesens, Kevin Mets, Tim Verdonck, et al. Advances in continual graph learning for anti-money laundering systems. In *10th ECML-PKDD Workshop on MIning DAta for financial applicationS*, 2025.

Mingjiang Duan, Tongya Zheng, Yang Gao, Gang Wang, Zunlei Feng, and Xinyu Wang. Dga-gnn: Dynamic grouping aggregation gnn for fraud detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 11820–11828, 2024.

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*, pp. 3979–3990. ACM, 2023. doi: 10.1145/3580305.3599803.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen, and Michalis Vazirgiannis. Dgraph: A large-scale financial dataset for graph anomaly detection. *Advances in Neural Information Processing Systems*, 35:22765–22777, 2022.

Valter Hudovernik. Relational data generation with graph neural networks and latent diffusion models. In *NeurIPS 2024 Third Table Representation Learning Workshop*, 2024.

Fredrik Johannessen and Martin Jullum. Finding money launderers using heterogeneous graph neural networks. *The Journal of Finance and Data Science*, pp. 100175, 2025.

T. N. Kipf. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Junhong Lin, Xiaojie Guo, Yada Zhu, Samuel Mitchell, Erik Altman, and Julian Shun. Fraudgt: A simple, effective, and efficient graph transformer for financial fraud detection. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 292–300, 2024.

Wei Pang, Masoumeh Shafieinejad, Lucy Liu, Stephanie Hazlewood, and Xi He. Clavaddpm: Multi-relational data synthesis with cluster-guided diffusion models. *Advances in Neural Information Processing Systems*, 37:83521–83547, 2024.

Chung-Hoo Poon, James Kwok, Calvin Chow, and Jang-Hyeon Choi. Linemvgnn: Anti-money laundering with line-graph-assisted multi-view graph neural networks. *AI*, 6(4):69, 2025.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pp. 593–607. Springer, 2018.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London, 1986. doi: 10.1201/9781315140919.

Jianheng Tang, Jiajin Li, Ziqi Gao, and Jia Li. Rethinking graph neural networks for anomaly detection. In *International Conference on Machine Learning*, 2022.

Zhihua Tian, Yuan Ding, Xiang Yu, Enchao Gong, Jian Liu, and Kui Ren. Towards collaborative anti-money laundering among financial institutions. In *Proceedings of the ACM on Web Conference 2025*, pp. 4722–4733, 2025.

Paul Tiwald, Ivona Krchova, Andrey Sidorenko, Mariana Vargas Vieyra, Mario Scriminaci, and Michael Platzer. Tabularargn: A flexible and efficient auto-regressive framework for generating high-fidelity synthetic data. *arXiv preprint arXiv:2501.12012*, 2025.

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

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.

Ruofan Wu, Boqun Ma, Hong Jin, Wenlong Zhao, Weiqiang Wang, and Tianyi Zhang. Grande: a neural model over directed multigraphs with application to anti-money laundering. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 558–567. IEEE, 2022. ISBN 1-6654-5099-1. GSCC: 0000010 2025-09-27T06:59:21.633Z 0.05.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pp. 7333–7343, 2019. URL `https://arxiv.org/abs/1907.00503`. arXiv: 1907.00503.

Wen Zheng, Bingbing Xu, Emiao Lu, Yang Li, Qi Cao, Xuan Zong, and Huawei Shen. Midlg: Mutual information based dual level gnn for transaction fraud complaint verification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5685–5694, 2023.

A Data generation details

A.1 Formal schema definition

The heterogeneous transaction graph is formalized as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with

$$\mathcal{V} = \mathcal{V}^{\text{in}}_{\text{usr}} \uplus \mathcal{V}^{\text{in}}_{\text{acc}} \uplus \mathcal{V}^{\text{out}}_{\text{ent}} \uplus \mathcal{V}^{\text{out}}_{\text{acc}}, \qquad \mathcal{E} = \mathcal{E}^{\text{in}}_{\text{own}} \uplus \mathcal{E}^{\text{out}}_{\text{own}} \uplus \mathcal{E}_{\text{txn}}.$$

The four node sets collect in-bank users (`inner_user`), in-bank accounts (`in_account`), external entities (`outer_entity`), and external accounts (`out_account`), respectively, corresponding one-to-one to the node types in Table 2. Internal ownership edges $\mathcal{E}^{\text{in}}_{\text{own}}$ link in-bank users to in-bank accounts, external ownership edges $\mathcal{E}^{\text{out}}_{\text{own}}$ link external entities to external accounts, and transaction edges $\mathcal{E}_{\text{txn}}$ are directed transfers between in-bank and external accounts.

A.2 Generation details

Relevant generation approaches We also evaluated several alternative generation paradigms, including TabularARGN (Tiwald et al., 2025), RGCLD (Hudovernik, 2024), and ClavaDDPM (Pang et al., 2024), on this large-scale dataset. In our preliminary experiments, these methods either failed to provide satisfactory fidelity/utility trade-offs or incurred training and sampling costs beyond a practically tolerable runtime budget. In particular, methods based on autoregressive factorization or diffusion-style relational generation became increasingly difficult to scale as the dataset size and relational complexity grew, leading to unstable empirical behavior or prohibitive computational overhead. For this reason, we ultimately adopted the framework of (Darabi et al., 2025), which offered a more suitable balance between modeling performance and computational feasibility for our setting.

Generation settings The three kernel functions were evaluated using the open-source implementation provided by (Darabi et al., 2025). For CTGAN, we did not use the default configuration, as the scale and structural complexity of the graphs in our experiments required a more stable and expressive setup. Specifically, we increased the batch size, the number of training epochs, the hidden dimensions of the generator and discriminator, and the latent embedding dimension, while reducing the learning rates of both optimizers. All remaining implementation details followed the standard defaults. To avoid disclosing sensitive operational characteristics of the underlying banking scenario, we omit the exact values of these non-default hyperparameters, as well as further details on the computational resources and execution environment. Following data simulation, we further refined the generated features and labels to preserve the underlying distributional properties of the original dataset.

We adopt the same fidelity metrics as (Darabi et al., 2025). For readability, we briefly summarize the three metrics reported in Table 6. *Degree Distribution* measures how well the generated graph preserves the degree profile of the original graph, i.e., the distribution of node connectivity over the graph. *Feature Correlation* evaluates whether dependencies among feature columns are preserved, including correlations between continuous features as well as associations involving categorical

Table 6: Fidelity comparison of three feature generation kernels within the graph-simulation framework of (Darabi et al., 2025). All metrics (%) measure the discrepancy between simulated and original data. ↑ indicates higher is better while ↓ indicate lower is better.

Metric	CTGAN	KDE	Uniform
Degree Distribution ↑	**72.8**	42.6	35.7
Feature Correlation ↑	**92.38**	89.3	86.4
Degree-Feature Joint Distribution ↓	**23.19**	35.84	32.55

variables. *Degree-Feature Joint Distribution* measures the consistency between graph structure and attributes by comparing the joint distribution of node degrees and features between the real and generated graphs. Table 6 compares three feature-generation kernels under the same graph-simulation framework (Darabi et al., 2025): CTGAN (Xu et al., 2019), KDE (Silverman, 1986), and Uniform, which samples each feature independently and uniformly over its observed support; all three are evaluated as kernels within this framework rather than methods proposed in this work. CTGAN performs best across all metrics, achieving the highest structural and attribute fidelity scores and the lowest joint distribution discrepancy, suggesting that it better preserves the marginal attributes and structure–attribute dependencies of the original transaction system. We thus adopt CTGAN as the final simulator in PERISCOPE.

Evaluation of Fidelity The above distribution comparison does not indicate the absolute similarity between the simulated dataset and the original data. Therefore, we further visualize the three metrics consistent with Table 6, as shown in Figures 2 and 5. In terms of distribution, our released dataset closely matches the real dataset in several statistical characteristics, with only minor distributional perturbations.

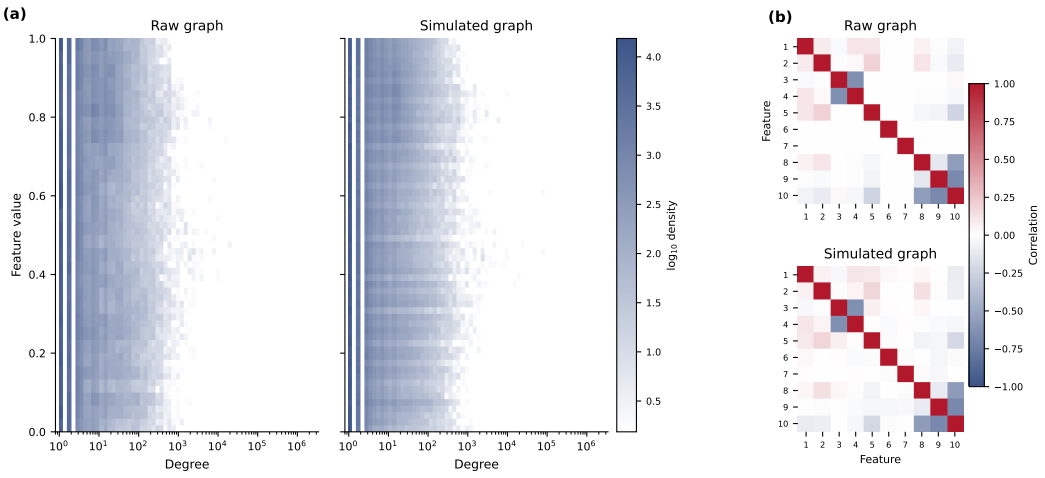


Figure 5: (a) Joint distribution of transaction degree and account feature (quantile scale) for the raw and simulated graphs under a shared log-density color scale. (b) Feature correlation matrices over the ten transaction features: each cell reports the Pearson correlation coefficient between a pair of transaction features, computed over all transaction edges of the respective graph; blue denotes negative and red positive values under a shared symmetric color scale.

B FURTHER ANALYSIS OF SUSPICIOUS MULTI-ACCOUNT USERS

B.1 CONTRASTIVE EVIDENCE FROM BENIGN MULTI-ACCOUNT USERS

Definition. To test whether overlap patterns are specific to suspicious behavior rather than a generic effect of owning multiple accounts, we compared all account pairs formed by suspicious users with those of benign users. The resulting AO and EO hit rates measure how frequently each type of overlap appears within suspicious and benign multi-account users.

Table 7: Pair-level contrastive analysis between suspicious and benign multi-account users. Positive pairs are sampled within black users, while negative pairs are sampled within white users. AO and EO denote the proportions of account pairs satisfying account-level overlap and entity-level overlap, respectively.

Pair type	#Pairs	AO hit rate	EO hit rate	EO–AO gain
Negative pairs within normal users	2,036	48.72%	64.10%	15.37%
Positive pairs within anomalous users	1,843	65.00%	77.65%	12.64%

Table 8: Lift of overlap indicators for distinguishing account pairs sampled within black users from those sampled within white users. The baseline black-user pair rate is $p_0 = 47.51\%$. Lift values larger than one indicate enrichment over the baseline.

Category	Metric	Value
Baseline	Black-user pair rate p_0	47.51%
Lift	AO lift	1.15
Lift	EO lift	1.10
Lift	EO-only lift	0.90

Discussion. Table 7 shows that suspicious pairs exhibit stronger overlap than benign pairs under both AO and EO, indicating that shared counterparties are more characteristic of suspicious multi-account behavior than ordinary multi-account usage. EO consistently captures significantly additional relations beyond AO in both groups, as expected from entity-level consolidation. This demonstrates the value of AO and EO.

B.2 Discriminative Strength of Overlap Signals

Definition. We evaluate whether AO and EO are enriched among suspicious pairs using lift. Let $Y = 1$ denote an account pair sampled within a black user and $Y = 0$ denote a pair sampled within a white user, and let $p_0 = P(Y = 1)$ be the baseline black-user pair rate in the sampled evaluation set. For an overlap indicator $Z \in \{\mathrm{AO}, \mathrm{EO}\}$, lift is defined as $P(Y = 1 \mid Z = 1)/p_0$, i.e., the relative increase in the probability that an overlapping pair comes from a black user. We also report the EO-only lift, computed for pairs satisfying EO but not AO.

Discussion. Table 8 shows that both AO and EO are enriched among black-user pairs, confirming that shared counterparties provide useful evidence for distinguishing suspicious multi-account behavior from benign multi-account usage. AO exhibits stronger discriminative association, while EO broadens the notion of overlap by consolidating accounts at the entity level, which indicating that EO should not be viewed as an independent rule for detection. Rather, its value lies in complementing AO by exposing latent entity-level relations that can be jointly interpreted with account ownership, transaction structure, and temporal context in Periscope.

C Experimental setup and reproducibility details

Dataset splits We split all transaction edges in chronological order into training, validation, and test sets with a ratio of 60%, 20%, and 20%, respectively. Account features are then constructed by averaging the transaction attributes of neighboring account nodes. User features are subsequently generated by averaging the features of the accounts associated with each user. This chronological construction ensures that no future information is leaked during the experimental process.

Evaluation protocol We use AUPRC and F1 as the evaluation metrics. AUPRC is particularly suitable for imbalanced classification settings, as it is more sensitive to the minority class and reflects the overall discriminative ability of the model. F1, on the other hand, is closer to practical deployment scenarios, as it evaluates model performance under a specific decision threshold. In all experiments, we use AUPRC as the primary metric for model selection.

Implementation details We perform random hyperparameter search over 20 trials. Specifically, the learning rate is sampled from $(1e{-}4, 1e{-}2)$, the dropout rate from $(0.1, 0.5)$, and the activation function from $\{relu, tanh, leakyrelu\}$. The maximum number of training epochs is set to 400, and early stopping is applied with a patience of 10 epochs. The number of seed nodes in each batch is set to 1024. The number of hidden channels of models are set to 128. During training, the model is evaluated on the validation set after each epoch, and the checkpoint with the best validation performance is selected for final evaluation on the test set. All reported results are averaged over five independent runs, and we report both the mean and standard deviation for each metric.

Training strategy For different task levels and graph-structure ablation settings, we adopt task-specific sampling strategies to ensure that each model has access to the relevant neighborhood semantics while maintaining a comparable message-passing depth. The sampling patterns are designed to follow the natural information flow of each prediction target: user-level prediction requires information to propagate through associated accounts and their transaction neighborhoods, whereas account-level prediction centers on account nodes and aggregates information from both ownership and transaction relations. We therefore enable only the relations that are semantically relevant at each hop, which avoids unnecessary neighborhood expansion and makes the comparison between the original and ablated graphs relatively fair.

For the user-level task on the original graph, the seed nodes consist of all non-isolated internal users. The sampling strategy for ownership relations is $[-1, 0, 0, -1]$, while that for transaction relations is $[0, -1, -1, 0]$. Here, 0 indicates that the corresponding relation is not sampled during the subgraph expansion step, whereas -1 indicates that all neighbors under that relation are sampled in the current round. For the account-level task on the original graph, the seed nodes consist of all non-isolated internal accounts. We use $[-1, -1, 0, 0]$ for ownership relations and $[0, 0, -1, -1]$ for transaction relations. In the ablated graph, where ownership relations are removed, the sampling strategy for transaction relations is set to $[-1, -1]$. Since the model on the original graph involves 4-hop neighborhoods, the number of GNN layers is set to 4. In contrast, for the ablated graph, the number of GNN layers is set to 2.

Reproducibility Our experiments are conducted on a server equipped with four 5090 GPUs, two AMD EPYC 9654 CPU, and Ubuntu 24.04. Each run of our model is restricted to a single GPU with 32GB of VRAM. Details of the experimental environment, result monitoring, and reproducibility are provided in the supplementary materials and accompanying code repository.