JCTA Journal of Computing Theories and Applications ISSN:3024-9104

Review Article

Graph-Based Financial Fraud Detection: A Property-Centric Survey of Static, Temporal, Cycle-Aware, and Generative Methods and Their Evaluation

Daniel T. Ale *, E. O. Ogunti, S. A. Oyetunji, A. A. Ponnle, and K. B. Adedeji

Department of Electrical and Electronics Engineering, Federal University of Technology, Akure 340001, Ondo State, Nigeria; e-mail : danielale9291@gmail.com; eoogunti@futa.edu.ng; saoyetunji@futa.edu.ng; aaponnle@futa.edu.ng; kbadedeji@futa.edu.ng

* Corresponding Author : Daniel T. Ale

Abstract: Financial fraud is increasingly coordinated across multiple accounts and transactions, meaning that its evidence often lies in relationships between accounts, the sequence of events, and, in schemes such as money laundering and wash trading, closed loops of transfers rather than in individual records. This review examines graph-based methods for detecting such fraud, covering static and temporal graph neural networks, graph anomaly detection, cycle- and motif-aware methods, generative models, and the evaluation practices used to assess them. The review draws on 147 sources identified through a documented literature search protocol. Four findings emerge. First, no single method family performs best under all conditions: tree-based models remain highly competitive on feature-rich datasets, whereas graph-based and temporal models offer greater advantages when discriminative signals reside in relational structures or transaction sequences. Second, detecting cyclic laundering is fundamentally a representational problem: standard message-passing architectures have limited ability to capture cycle structures, while recent performance gains have been associated with representations that preserve the directed multigraph structure of transactions rather than simply increasing model complexity. Third, evaluation practices, including temporal leakage and threshold selection using test data, can inflate reported performance and undermine the reliability of comparisons. Fourth, a taxonomy of eight method families evaluated against four properties required for coordinated, temporal, and partially observed fraud detection—temporal awareness, probabilistic generation, cycle awareness, and intrinsic explainability—reveals that none of the reviewed families provides all four properties. This identifies a central research gap in the current literature. Building on these findings, the review proposes cyclic fraud as a structural stress test for future detection methods and provides an evaluation checklist to address the methodological weaknesses identified.

Keywords: Anti-Money Laundering; Cycle Detection; Diffusion Models; Financial Fraud; Graph Anomaly Detection; Graph Neural Networks; Money Laundering; Temporal Graphs.

Received: September, 14th 2026
Revised: October, 18th 2026
Accepted: October, 20th 2026
Published: November, 17th 2026

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) licenses (https://creativecommons.org/licenses/by/4.0/)

1. Introduction

Digital financial infrastructures move value through dense webs of accounts, merchants, agents, and wallets, with transactions arriving as irregular event streams rather than evenly spaced observations [1], [2]. Occupational fraud alone is estimated to cost organizations around five per cent of annual revenue, and the layering stage of money laundering is deliberately engineered to defeat record-level scrutiny [1]. In this setting, fraud is increasingly a property of relations over time rather than of individual records: coordinated actors distribute suspicious activity across many entities and many steps precisely so that no single step looks abnormal [3], [4]. A recurring and especially difficult pattern is the cycle, a closed loop such as A→B→C→A that underpins wash trading, artificial-volume inflation, and the layering stage of money laundering [5]–[7].

What follows from this is that a detector reasoning over records in isolation, or over a single frozen view of the transaction graph, will let organised fraud through as a matter of course. The reason is a mismatch of scope: what condemns the activity is never held by one transaction or one account, but distributed across a sub-graph that is ordered in time and, more often than not, only partly recorded. Successive waves of work over the last decade have widened that scope, moving from hand-built features to static graph neural networks (GNNs), then to temporal ones, and most recently to deep generative models, with each wave answering a shortcoming of the one before. It is that progression this review traces, judged against what cyclic, time-ordered, partially observed fraud actually requires of a detector.

1.1. Motivation and Research Gap

Several capable surveys of graph-based fraud detection already exist. Systematic reviews catalogue the application of GNNs to financial fraud [4], survey graph-based anomaly detection for fraud more broadly [8], review anomaly detection techniques for financial crime [9], and chart progress and future directions for graph learning in this domain [3]. Earlier reviews established the data-mining [10] and machine-learning [11], [12] framing of the problem, and the statistical foundations were laid out well before the graph era [13]. What these reviews share is an organization by algorithm family or by application vertical. That organization is natural and useful, but it makes a specific question hard to answer: given a fraud typology with known structural and temporal properties, which capabilities does a detector need, and which of them does the literature actually supply?

The organising principle adopted here is different: the review is arranged around the properties, structural and statistical, that coordinated fraud makes necessary. Ordering the material this way brings one particular combination of capabilities into view as conspicuously absent from the literature, and it accounts for an otherwise puzzling phenomenon: architectures that top node-classification leaderboards can perform poorly against organised laundering. A single notation is used throughout, so that what each family assumes about the problem can be read off its equations instead of being reconstructed from surrounding prose. Table 1 places this review beside its closest neighbours.

Table 1. Positioning of this review against closely related surveys of graph-based fraud detection. ✓ denotes substantive treatment, ~ partial treatment, and – absence

Survey	Organization	Temporal awareness	Probabilistic generation	Cycle awareness	Evaluation pitfalls
Ngai et al. [10]	by data-mining technique	–	–	–	~
West and Bhattacharya [11]	by learning method	–	–	–	~
Pourhabibi et al. [8]	by anomaly-detection family	~	–	~	~
Hilal et al. [9]	by anomaly technique	~	–	–	~
Motie and Raahemi [4]	by GNN architecture	~	–	–	~
Li et al. [3]	by graph-learning task	✓	~	–	~
This review	by required property	✓	✓	✓	✓

1.2. Review Questions and Contributions

Four questions drive the review, each answered in the sections indicated.

1. RQ1. How is financial fraud represented as a graph-learning problem, and which typologies leave which structural signatures? (Section 3)
2. RQ2. Which method families address relational structure, continuous time, structural incompleteness, and cyclic constraints, and what are their governing equations and inductive biases? (Section 4, Section 5, and Section 6)
3. RQ3. How should such methods be evaluated on imbalanced, temporal, partially observed data, and which methodological pitfalls recur? (Section 7)
4. RQ4. Which combination of capabilities does the literature not yet provide, and how should future methods be evaluated against it? (Section 8 and Section 9)

Against these questions, the review makes five contributions. First, it synthesises evidence across method families and links recurring limitations to the studies that support them

(Section 8.1). Second, it derives four required properties, temporal awareness, probabilistic generation, cycle awareness, and intrinsic explainability, from the limitations identified across the literature, and uses them to build a taxonomy of eight method families that exposes the central research gap (Section 8.2). Third, it presents the governing equations of each family in a single notation, so that architectural differences reduce to differences in a small number of operators. Fourth, it brings together topics that are usually surveyed separately: anti-money-laundering (AML) systems, graph anomaly detection, adversarial robustness, explainability, scalability, and privacy-preserving collaboration. Fifth, it treats evaluation practice as a finding in its own right, documenting the pitfalls that inflate reported performance and linking each to an item of a reproducibility checklist (Section 7.4)

1.3. Key Takeaways of the Review

The review highlights four key takeaways, summarised below as a guide to its main conclusions. The supporting evidence, interpretation, and implications of each finding are examined in Section 8.1.

1. No single approach is best under all conditions (Section 4 and Section 7.3).
2. Cyclic structure is a representational problem (Section 4.2 and Section 5.1).
3. Evaluation practice can inflate reported performance (Section 7.4).
4. No method family provides all four required properties (Section 8.2 and Section 8.3).

1.4. Organization of the Review

Figure 1 summarises the organization. Section 2 documents the review methodology. Section 3 sets out how financial fraud is represented as a graph-learning problem, from fraud typologies and their graph signatures to graph construction and features. Section 4 traces the evolution of detection from rule-based monitoring and classical machine learning through static to temporal graph neural networks. Section 5 covers structural incompleteness and cycle-aware methods, graph anomaly detection, and generative inference with score-based diffusion. Section 6 reviews AML systems and the concerns that decide whether a method can be deployed: robustness, explainability, scalability, and privacy. Section 7 examines evaluation methodology and the pitfalls that inflate reported performance. Section 8 states the principal findings, derives the four required properties and the taxonomy, and discusses cyclic fraud as a stress test. Section 9 sets out open challenges and future directions, and Section 10 concludes.

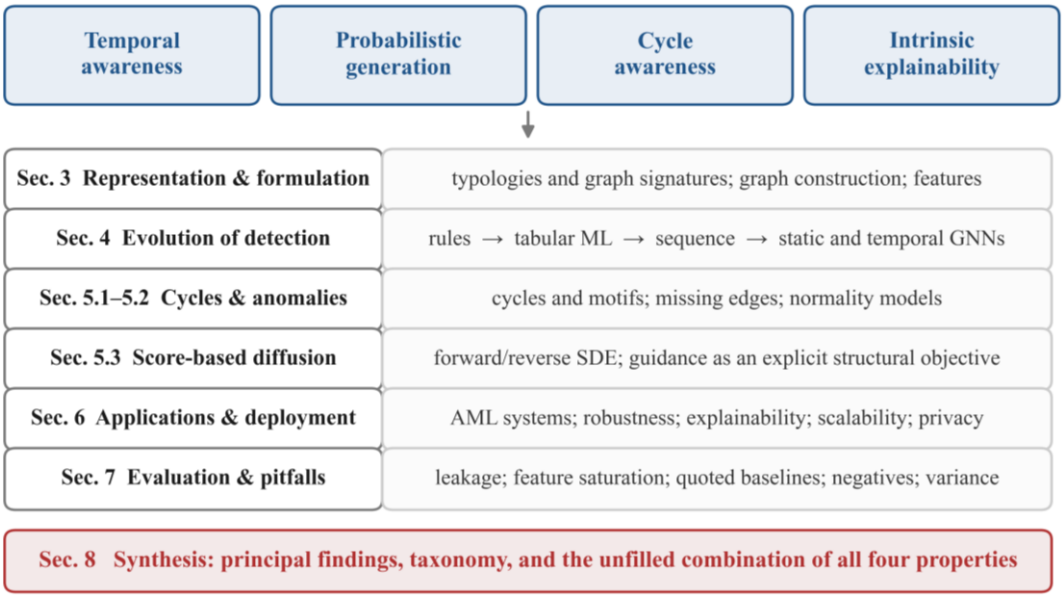


Figure 1. Organization of this review around the four required properties. Each method family reviewed here supplies a subset of the properties, and the synthesis in Section 8 identifies the combination that no family yet provides

2. Review Methodology

Although this is a narrative rather than a strictly systematic review, the search and selection process was documented so that the coverage can be assessed and reproduced. This section states the protocol, the inclusion and exclusion criteria, the coding scheme used to extract properties from each study, and the threats to the validity of the review itself.

2.1. Search Protocol

Five bibliographic sources were queried: Scopus, IEEE Xplore, the ACM Digital Library, Web of Science, and DBLP, the last of which gives the most reliable coverage of the machine-learning conference literature in which most graph-learning advances first appear. The search combined three concept groups with Boolean conjunction. The domain group comprised financial fraud, money laundering, anti-money laundering, wash trading, and transaction monitoring. The method group comprised graph neural network, temporal graph, dynamic graph, graph anomaly detection, diffusion model, score matching, and generative model. The structure group comprised cycle, motif, subgraph, and closed walk. Queries were run over titles, abstracts, and author keywords, and were restricted to the period 2015–2025, with foundational works outside that window admitted by backward citation chaining when they define a method still in active use.

2.2. Inclusion and Exclusion Criteria

Studies were screened and selected based on predefined inclusion and exclusion criteria. For clarity and consistency, the inclusion criteria are labelled I1–I3, while the exclusion criteria are labelled E1–E3.

Inclusion criteria:

- I1. The work proposes, evaluates, or surveys a graph-based method applicable to financial fraud, money laundering, or a structurally analogous relational-abuse problem.
- I2. The work defines a method that supplies at least one of the four reviewed properties: temporal awareness, probabilistic generation, cycle awareness (which includes awareness of motifs), or intrinsic explainability.
- I3. Foundational methods outside the fraud domain are included when they define an operator that fraud-domain work builds on directly, for example message passing, denoising score matching, or continuous-time memory updates.

Exclusion criteria:

- E1. Works that treat transactions purely as independent tabular records, unless cited as a baseline or as evidence of benchmark saturation.
- E2. Works reporting only proprietary results with no reproducible protocol, dataset description, or metric definition.
- E3. Duplicate reports of the same contribution; in each case the archival version was retained and the preprint discarded.

2.3. Screening and Data Extraction

Records were screened first on title and abstract and then on full text. For each retained study, the following fields were extracted: the task addressed, the graph construction used, the temporal treatment (static, discrete snapshot, or continuous time), whether the model is discriminative or generative, whether cyclic or higher-order structure is represented explicitly, the explainability mechanism if any, the datasets used, the split protocol, and the metrics reported. Each study was then coded against the four properties of Figure 1 on a three-level scale, namely supported, partially supported, and absent. The coding rules were fixed in advance.

- Temporal awareness requires that event ordering influence the representation, not merely that a timestamp appear as a feature. We distinguish time that is modelled, when the computation of the architecture depends on event order or elapsed time, for example through memory updates or time encodings, from time that is merely supplied, when a timestamp or time-step index enters as an ordinary input attribute. Only modelled time counts as temporal awareness.

- Probabilistic generation requires a learned distribution over structure or attributes rather than a point estimate.
- Cycle awareness requires an explicit representation of closed structure, whether by enumeration, by walk statistics, or by a spectral relaxation.
- Intrinsic explainability requires that the evidence for a decision be a by-product of inference rather than a separate post-hoc approximation.

2.4. Threats to the Validity of This Review

Four limitations should be borne in mind. First, selection bias: the machine-learning literature is dominated by conference publication and preprints, so coverage of production systems, which are rarely published, is necessarily thin, and our account of deployment constraints is drawn from the subset of work that reports them. Second, comparability: because splits, metrics, and graph constructions differ across studies, the comparative statements in Section 8 are deliberately qualitative; we do not aggregate numbers across papers, since doing so would compound precisely the methodological inconsistencies documented in Section 7. Third, recency: the generative graph-modelling literature is moving quickly, and a review of it is a snapshot; we mitigate this by organising around properties, which change more slowly than the architectures that supply them. Fourth, dependence on the coding: the research gap reported in Section 8.3 is derived from this protocol and these coding rules, so it describes the reviewed literature rather than establishing a universal limitation.

3. Financial Fraud Representation and Problem Formulation

This section sets out how financial fraud becomes a graph-learning problem. It begins with the anatomy of fraud and the graph signatures that different typologies leave, then turns to the formulation of the learning task and the class imbalance that shapes it, and ends with the construction choices and features that turn raw transaction logs into graphs. Graph construction, often treated as preprocessing, is placed here because it determines what any later model can see.

3.1. Anatomy of Financial Fraud

Fraud is the intentional use of deception for unlawful gain. In digital finance it spans occupational fraud, cyber-enabled fraud, investment scams, and money laundering involving the placement, layering, and integration of illicit proceeds [1], [2]. Across these categories two structural archetypes recur. Acyclic fraud includes star-shaped mule networks and one-to-many scam payouts, in which a hub distributes or collects funds. Cyclic fraud involves closed loops, whether A→B→C→A or repeated A→B→A transfers, and is associated with wash trading, artificial-volume inflation, incentive abuse, and laundering layering [5]–[7]. The defining property of cyclic fraud is that individual transactions in the loop appear benign in isolation; only the closed-loop topology over time reveals intent, which is precisely why transaction-independent classifiers struggle with it.

3.2. Graph Signatures of Fraud Typologies

Every laundering typology imprints a recognisable shape on the transaction graph, as Figure 2 depicts and Table 2 tabulates. Smurfing splits one large sum into many deposits that individually fall below reporting thresholds, and the resulting one-to-many fan is its fingerprint. Round-tripping and integration send value back where it started and so close a loop. Funds relayed through a succession of intermediaries by a mule network trace out a chain. Colluding parties transact densely among themselves and approach a clique. Two consequences follow from cataloguing these shapes. The first is diagnostic: since each fingerprint spans several nodes and edges, no quantity computed from one account's attributes can capture it, which bounds what attribute-only detectors can achieve in principle. The second is operational: supervisors and investigators already reason in exactly these terms, so a detector that reports them speaks a language its audience accepts, which eases both deployment and the justification of an alert.

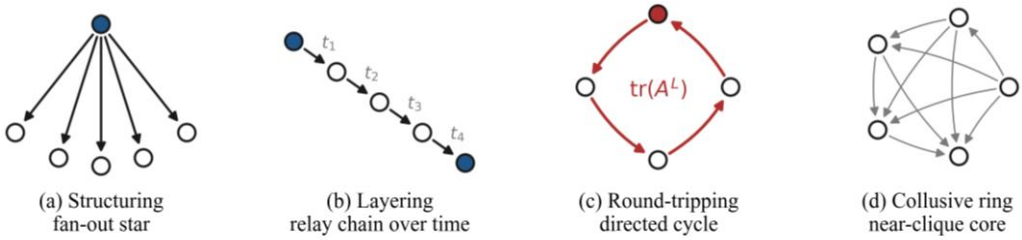


Figure 2. Graph signatures of the principal laundering typologies. Shaded nodes mark the originating or benefiting entity. Only (c) requires reasoning about closure, and only (b) and (c) require reasoning about the order in which events occur

Table 2. Fraud typologies, their graph signatures, and the detection cue each implies

Typology	Graph signature	Detection cue	Order matters
Structuring (smurfing)	fan-out star	high out-degree, sub-threshold amounts	no
Layering	relay chains	long directed paths over a bounded window	yes
Round-tripping	directed cycle	closed walks, tr(AL)	yes
Collusive rings	dense core	high clustering, near-clique	no
Mule networks	star into chain	dormant-then-burst activation	yes
Wash trading	repeated 2-cycles	reciprocal volume with no net position	yes

3.3. Why Graphs

A financial system is naturally a directed temporal multigraph, $\mathcal{G} = (V, R, T, E)$, where V are entities, R relation types, and $T \subseteq \mathbb{R}^+$ continuous time. A transaction is the event $e_i = (u_i, v_i, r_i, t_i, a_i)$, in which a_i collects the attributes of the transfer such as amount, channel, and device.

$$G = (V, R, T, E), E \subseteq V \times V \times R \times T \tag{1}$$

$$e = (u, v, r, t, a) \tag{2}$$

The graph is directed because value flows from sender to receiver, and a multigraph because a pair may transact repeatedly under different types and at different times. Equations (1) and (2) make fan-in, fan-out, repeated transfers, and cyclical movement first-class objects of inference, which independent-record classifiers cannot represent [14], [15].

3.4. Task Formulations

Four learning tasks recur. Node classification labels an account or a transaction as licit or illicit and is the formulation used on the dominant public benchmark [16]. Edge classification labels an individual transfer, which suits transaction-level alerting. Sub-graph classification labels a whole pattern, which is the natural formulation for typology detection but is the least well served by public labels. Link prediction asks whether an interaction will occur, and although it is not itself a detection task, it supplies the self-supervised signal on which temporal models are trained [17], [18]. In all cases the operational output is a risk score conditioned only on the past, Eq. (3),

$$s(v, t) = f_\theta(G_{\leq t}, v) \approx P(y = 1 \mid G_{\leq t}, v) \tag{3}$$

where $\mathcal{G}_{\leq t}$ is the graph restricted to events with timestamp at most t. Writing the task this way makes the leakage constraint explicit: any feature or aggregate computed from events after t violates the formulation, however innocuous it may appear in a preprocessing script.

3.5. The Nodal Distribution Underlying the Score

Written as Eq. (3), the task is a conditional expectation, and it is worth being explicit about the object that expectation is taken over, because the families surveyed below differ chiefly in how much of that object they are willing to model. Let $z \in \mathbb{R}^D$ denote a nodal state, the vector summarising an entity together with the recent relational context in which it

transacted. A discriminative model learns only the decision boundary: $P(y = 1 \mid z)$. A nodal distribution model is the stronger commitment: it specifies the class-conditional densities themselves, Eq. (4), most conveniently through an energy E_y with normaliser Z_y and class prior π_y, so that both the fraudulent and the non-fraudulent regime have an explicit law and the marginal $p(z)$ follows as their mixture.

$$p(z \mid y) = \frac{\exp(-E_y(z))}{Z_y}, \quad p(z) = \sum_{y \in \{0,1\}} \pi_y \, p(z \mid y) \tag{4}$$

The advantage of paying that cost is that detection then reduces to the classical likelihood-ratio test Eq. (5), in which the energies enter only as a difference, so the intractable normalisers cancel and a single scalar threshold τ trades detection rate against false alarms along the optimal operating curve. Three consequences matter for the sections that follow. The likelihood-ratio score is optimal under the assumed class-conditional densities and priors; however, probability calibration depends on correctly specified priors and density estimates and should be assessed empirically. The same densities support sampling, so unobserved but plausible relational configurations can be generated rather than assumed away, which is exactly the response the incompleteness of Eq. (14) demands. And any structural statistic, a closed-walk count among them, can be evaluated under $p(z \mid y)$ instead of on the single observed graph. Score-based diffusion (Section 5.3) is the most practical route to Eq. (4) at transaction scale, since it learns $\nabla_z \log p(z)$ without ever evaluating Z_y.

$$\Lambda(z) = \frac{p(z \mid y = 1)}{p(z \mid y = 0)} = \exp(E_0(z) - E_1(z)) \gtrless \tau \tag{5}$$

3.6. Structural Minority and Message Imbalance

Empirically, illicit activity constitutes what is best described as a structural minority. The nodes and edges involved are few, poorly interconnected, and scattered through large licit components, so what is skewed is the shape of the graph and not simply the ratio of labels. The sharper consequence is what may be called message imbalance. Aggregation builds a representation out of whatever a node's neighbourhood supplies, and that neighbourhood is almost entirely licit, so the majority class has already skewed the representation before the classifier sees a single gradient [14]. Nearly every difficulty documented in the sections that follow traces back to this one fact, and what most distinguishes the families from one another is the strategy each adopts against it. Table 3 fixes the notation used throughout.

Table 3. Unified notation used throughout this review

Symbol	Meaning	Symbol	Meaning
$\mathcal{G} = (V, R, T, E)$	temporal multigraph	$s_i(t)$	memory state of node i
$e = (u, v, r, t, a)$	transaction event	$\varphi(\Delta t)$	time encoding
$\mathcal{N}(i)$	neighbourhood of i	A	(weighted) adjacency matrix
$h_i^{(l)}$	layer-l embedding	$\mathrm{tr}(A^L)$	closed walks of length L
E_o, E_m	observed, missing edges	$p_\theta(e)$	learned edge plausibility
x_t	diffused latent at time t	s_θ	learned score network
$\beta(t)$	noise schedule	γ	guidance strength

3.7. Graph Construction and Representation

Before any model is applied, raw transaction logs must be turned into graph-structured inputs, and the quality of this representation often determines performance more than the choice of architecture. This stage is also the least consistently reported, which is a recurring obstacle to fair comparison.

3.7.1. Graph Construction Choices

A design choice that precedes any model is the graph schema itself. Transactions can be cast as edges between account nodes, or, as in the dominant public benchmark, transactions can themselves be nodes with flow edges between them [16]. The former emphasises account

behaviour and makes account-level typologies such as fan-out directly visible; the latter emphasises transaction provenance and makes chains of value transfer explicit. Because the same activity yields a different vocabulary of motifs under each convention, a loop visible in one need not be a loop in the other. Resolving entities adds a second decision on top of the first. Where one party operates many accounts, leaving them separate breaks a single loop into fragments that look mutually unrelated, while merging too aggressively invents structure with no counterpart in reality. Both choices can matter more to the final numbers than the architecture does, and both are routinely described too briefly for anyone to reproduce.

3.7.2. Node, Edge, and Temporal Features

Node features typically combine entity attributes such as account age, type, and geography with aggregate transactional statistics such as inbound and outbound volume, counterparty count, and balance dynamics. On attribute-rich benchmarks the strong performance of tree ensembles is largely attributable to such hand-engineered aggregates [16]. Edge features encode amount, channel, device, and direction, and edge typing captures the relational semantics exploited by heterogeneous GNNs [19]–[21]. Temporal features are encoded through inter-event gaps, decay terms, and learnable harmonic time embeddings that map a scalar timestamp into a vector via sinusoidal bases, Eq. (6), allowing continuous-time models to condition on recency [22]. Structural and positional features such as degree profiles, clustering coefficients, motif counts, and closed-walk statistics inject information that message passing alone is provably limited in recovering [23], [24], and are therefore a pragmatic complement to learned embeddings.

$$\varphi(\Delta t) = [\cos(\omega_1 \Delta t), \sin(\omega_1 \Delta t), \ldots, \cos(\omega_d \Delta t), \sin(\omega_d \Delta t)] \quad (6)$$

3.7.3. Leakage and Causal Feature Computation

Two tensions run through this stage. The first sets descriptive power against leakage. An aggregate taken over the whole of a node's history will carry information from after the moment of prediction unless the temporal boundary is respected when the feature is computed, and the resulting inflation of reported accuracy leaves no trace in the output. The remedy is to build every feature causally, from data available no later than the prediction time in Eq. (3), and to fit any normalisation on the training partition alone. The second tension sets transferability against fit. Features engineered around the quirks of one corpus seldom survive contact with another, which is the argument for encoding what is known about the problem in the architecture instead, since biases such as respect for event order or sensitivity to closure hold regardless of the dataset.

4. Evolution of Graph-Based Fraud Detection

The sequence of representations that the field has adopted is not an arbitrary succession of fashions. Each generation relaxed a specific assumption of its predecessor, and reading the history this way makes the remaining assumption visible. Figure 3 summarises the progression.

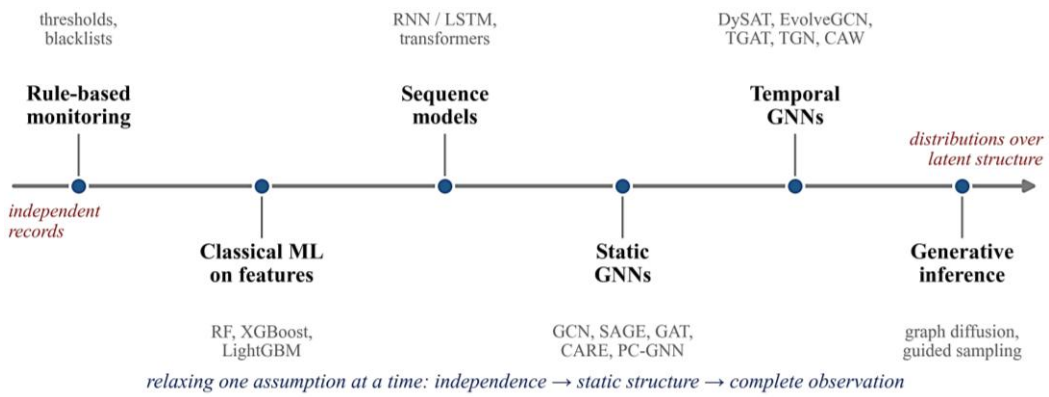


Figure 3. Evolution of fraud-detection paradigms. Each transition relaxes an assumption of the previous stage: independence of records, then a static graph, then a completely observed graph

4.1. From Rule-Based Monitoring to Graph Learning

4.1.1. Rule-Based Monitoring

The earliest monitoring systems were built from hand-written rules: fixed thresholds, blacklist lookups, and limits on transaction velocity. Supervisors liked them because an analyst could always say why an alert fired, but they aged badly. Maintaining them was laborious, and defeating them required only that an adversary break payments into smaller pieces, cycle through identities, and stay under the published limits [4], [13]. What survives from that era is a lesson the later families keep relearning, namely that whether a detector reaches production is frequently settled by how defensible its alerts are and not by how accurate it is.

4.1.2. Classical Machine Learning on Engineered Features

Classical machine learning improved predictive performance by learning from engineered features. Random forests [25] and gradient-boosted trees, particularly the scalable implementations that became standard practice [26], [27], remain formidable baselines on attribute-rich financial data. The limitation is structural rather than statistical: these models treat transactions as independent tabular records and therefore cannot capture relational topology. Their strength on attribute-rich data is, as Section 7 discusses, also a source of benchmark saturation, since a well-tuned tree ensemble can approach the achievable ceiling on datasets whose engineered aggregates already encode much of the relational signal [16].

4.1.3. Sequence Models

A transitional family sits between the tabular and graph eras and is worth recording. While graph methods were still immature, practitioners turned each account's payment history into a time series and handed it to a recurrent network [28], and subsequently to a transformer [29], scoring trajectories that departed from the account's own norm. Within a single account these models see a great deal. Across accounts they see nothing at all, which leaves them as blind as tabular classifiers to the coordination between parties that constitutes organised fraud. Figure 4 places them in the temporal-but-not-relational quadrant, and it was precisely this gap that prompted the move first to relational and then to temporal-relational architectures.

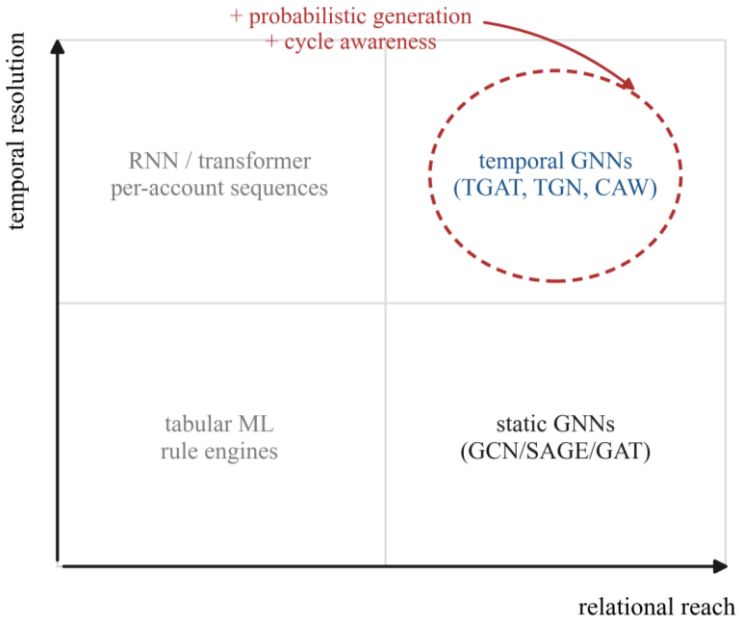


Figure 4. The design space of fraud detectors along relational reach and temporal resolution. Temporal graph networks occupy the demanding upper-right corner; the properties this review argues are still missing, probabilistic generation and cycle awareness, are orthogonal to both axes

4.1.4. The Graph Era and the Shift to Inference

The graph era reframed the problem: by representing the financial system as a network of entities and transactions, detection can exploit structural motifs such as rings, chains, stars,

and dense sub-graphs that indicate coordinated activity. Empirical surveys consistently report that graph methods outperform rule-based and flat-feature approaches on organised fraud [3], [4], [8]. The most recent shift, and the conceptual focus of the later sections, is from discriminative classification of an observed graph toward probabilistic, generative inference over latent structure. This shift is motivated by the recognition that the observed graph is incomplete and that fraud must therefore be reasoned about as uncertainty rather than asserted as fact.

4.1.5. Generative Model Families for Relational Data

Four families dominate deep generative modelling, all of which have graph variants. Variational autoencoders optimise a tractable evidence lower bound and yield smooth latent spaces [30], [31], but their reconstructions are often blurred and their likelihoods loose. Generative adversarial networks produce sharp samples through an adversarial game [32], [33] but train unstably and are prone to mode collapse, which is dangerous when rare fraud modes must be preserved rather than dropped. Autoregressive and flow-based models offer exact or tractable likelihoods [34]–[36] but impose an arbitrary ordering on an unordered object and scale poorly with graph size. Score-based diffusion models learn to reverse a fixed corruption process, train stably, and support guided generation [37], [38]. Diffusion and its use for detection are reviewed in Section 5.3.

4.2. Static Graph Neural Networks

The previous subsection traced how detection moved from rules applied to single records toward models of relations. Static graph neural networks are the first family to learn directly from the transaction graph. This subsection reviews how they aggregate neighbourhood information, the fraud-specific variants they have produced, and the limits on what they can represent; those limits, in particular the inability to count cycles, motivate much of what follows.

4.2.1. Canonical Message Passing

GNNs learn node representations by message passing [39]. A canonical layer update is:

$$h_i^{(l+1)} = \sigma\left(W^{(l)} \cdot \mathrm{AGG}_{j \in N(i)}\, h_j^{(l)} \right) \tag{7}$$

where $h_i^{(l)}$ is the embedding of node i at layer l, $\mathcal{N}(i)$ its neighbourhood, AGG a permutation-invariant aggregator, $W^{(l)}$ trainable weights, and σ a nonlinearity.

4.2.2. Representative Architectures

Equation (7) provides a general formulation of graph neural network aggregation, while representative architectures differ primarily in their aggregation mechanisms and their ability to capture specific graph properties. Table 4 summarises the principal architectures according to four fraud-relevant dimensions: aggregation strategy, inductive capability, temporal modelling, and relation awareness. These characteristics provide a comparative overview of the architectures discussed in this section.

Table 4. Representative graph architectures along fraud-relevant axes. *Inductive* means the model can score nodes unseen at training time

Model	Aggregator	Inductive	Temporal	Relation-aware
GCN [40]	normalised mean	no	no	no
GraphSAGE [41]	sample and pool	yes	no	no
GAT [42]	attention	yes	no	no
R-GCN [19]	per-relation	no	no	yes
HAN / HGT [20], [21]	typed attention	yes	no	yes
CARE-GNN [43]	filtered neighbours	yes	no	yes
PC-GNN [44]	resampled neighbours	yes	no	yes
DySAT / EvolveGCN [45], [46]	snapshot	partial	discrete	no
TGAT / TGN [22], [47]	temporal attention or memory	yes	continuous	partial

Among the foundational architectures, graph convolutional networks (GCNs) use a symmetric-normalised aggregation of neighbouring node representations, as expressed in Eq. (8), where d_i denotes the degree of node i [40]. GraphSAGE introduces inductive neighbour sampling with mean, pooling, or recurrent aggregators, enabling generalisation to unseen nodes at scale [41]. This capability is particularly relevant to fraud detection, where newly created accounts may need to be assessed as soon as they initiate transactions. Graph attention networks (GATs), in contrast, assign learned attention coefficients to neighbouring nodes, as expressed in Eq. (9), allowing the model to selectively weight neighbourhood information [42]. This mechanism may be useful in camouflage scenarios, where suspicious nodes deliberately connect to benign neighbours to obscure their structural characteristics. The corresponding aggregation formulations are given as follows:

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in N(i) \cup \{i\}} \frac{1}{\sqrt{d_i d_j}} W^{(l)} h_j^{(l)} \right) \tag{8}$$

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(a^{\top}\left[Wh_i \,\|\, Wh_j \right] \right) \right)}{\sum_{k \in N(i)} \exp\left(\text{LeakyReLU}\left(a^{\top}\left[Wh_i \,\|\, Wh_k \right] \right) \right)} \tag{9}$$

A finance-relevant refinement is relation awareness. Transaction graphs are heterogeneous, with edge types such as transfer, cash-out, deposit, and reversal carrying distinct semantics. Relation-specific aggregation addresses this heterogeneity by learning a separate transformation for each edge type before combining the resulting representations:

$$h_i^{(l+1)} = \sigma\left(W_0^{(l)} h_i^{(l)} + \sum_{r \in R} \sum_{j \in N_r(i)} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} \right) \tag{10}$$

where $\mathcal{N}_r(i)$ is the r-neighbourhood and $c_{i,r}$ a normalisation constant [19]. Attention-based heterogeneous variants extend this by learning type-aware attention at the node and semantic levels [20], [21]. Relation awareness improves expressiveness but multiplies parameters and aggravates the neighbour-explosion problem that inductive sampling partially mitigates [41].

Beyond general-purpose architectures, several models introduce mechanisms specifically relevant to fraud detection. CARE-GNN filters neighbouring nodes to reduce the influence of potentially misleading neighbours [43], whereas PC-GNN uses neighbour resampling to address class imbalance [44]. Temporal architectures capture the evolving nature of financial transactions. DySAT and EvolveGCN process graph snapshots to model structural changes over discrete time intervals [45], [46], while TGAT and TGN incorporate temporal information through time-aware attention or memory mechanisms [22], [47].

4.2.3. Fraud-Specific Designs

Generic GNNs often degrade under the adversarial conditions encountered in financial fraud detection, motivating the development of specialised aggregation mechanisms. Several fraud-specific architectures have therefore been proposed to address challenges such as neighbour camouflage, graph inconsistency, class imbalance, and heterogeneous relations. Table 5 provides an overview of these detectors and the primary mechanisms they employ, while the following discussion highlights their respective contributions.

CARE-GNN learns to select informative neighbours under camouflage, using reinforcement learning to determine adaptive filtering thresholds [43]. A related approach addresses the inconsistency problem directly, recognising that context, feature, and relation inconsistencies can each degrade neighbourhood aggregation and require distinct remedies [48]. PC-GNN tackles class imbalance at the sampling level through a pick-and-choose strategy that selects neighbourhoods to prevent scarce fraud nodes from being overwhelmed during aggregation [44]. FRAUDRE is designed to address both graph inconsistency and class imbalance [49], while H2-FDetector explicitly distinguishes homophilic from heterophilic connections, which is relevant because fraudulent accounts may have neighbourhoods that differ substantially from their own characteristics [50].

Earlier industrial research established the value of heterogeneous account graphs for detecting malicious accounts [51] and semi-supervised attentive aggregation for settings with

limited labelled data [52]. More recently, multi-relation interactive learning has been introduced to combine signals across different relation types while explicitly addressing class imbalance [15]. Together, these approaches illustrate how fraud-specific GNNs extend conventional aggregation to account for the structural and adversarial characteristics of financial fraud.

Table 5. Fraud-specific graph detectors and the mechanism each contributes

Detector	Primary mechanism	Problem addressed
GEM [51]	heterogeneous account aggregation	multi-device, multi-channel evasion
SemiGNN [52]	hierarchical attention, semi-supervision	label scarcity
CARE-GNN [43]	RL-tuned neighbour filtering	camouflage
GraphConsis [48]	context and relation consistency	inconsistency in aggregation
PC-GNN [44]	pick-and-choose neighbour resampling	structural minority
FRAUDRE [49]	dual resistance modules	inconsistency and imbalance
H2-FDetector [50]	homophily and heterophily separation	heterophilic neighbourhoods
MIDLG [15]	multi-relation interactive learning	relation fusion under imbalance

4.2.4. Expressiveness Limits

Message passing is bounded above in discriminative power by the Weisfeiler–Lehman graph isomorphism test [23], and a direct consequence is that standard GNNs cannot reliably count induced substructures such as cycles of a given length [24]. This is not a peripheral technicality for fraud: the very structures that define round-tripping are among those the architecture cannot represent by aggregation alone. Two responses exist. The first injects substructure counts as features, making the motif directly observable to the model [53]. The second lifts message passing to higher-order cells so that closed structures become first-class objects [54]. Both add expressiveness at computational cost, and both are under-explored in the fraud literature relative to their apparent fit.

4.2.5. Over-Smoothing, Depth, and Cold Start

Despite their effectiveness when the fraud signal is structural [55], [56], message-passing models exhibit several weaknesses in fraud settings. Under structural minority, the aggregated multiset is dominated by benign neighbours, diluting the suspicious signal and moving fraudulent embeddings toward the benign manifold [14]. Under camouflage, adversaries attach benign-looking edges so that the neighbourhood appears normal [43], [57]. A further architectural limit is over-smoothing: stacking many message-passing layers drives node representations toward indistinguishability [58], so the multi-hop reach needed to perceive a long laundering chain cannot simply be obtained by adding depth. Practical models are therefore shallow, typically two or three hops, which bounds the structural radius they perceive. Mechanisms such as jumping-knowledge aggregation [59], residual and identity mappings [60], edge dropping [61], and decoupled propagation [62] partly alleviate over-smoothing, but the fundamental tension between depth and discriminability remains a reason to seek structural signals, such as closed-walk statistics, that do not require deep stacks to surface. A related difficulty is heterophily: standard aggregation assumes that neighbours share labels, an assumption fraudsters actively violate [50], [63]. Finally, in production, models must contend with the cold-start problem, in which newly seen accounts have little neighbourhood and attribute priors must carry the decision until interactions accumulate.

4.3. Temporal Graph Learning

Static models treat the transaction graph as fixed, which discards the order and timing of events. Temporal graph learning restores them. The distinction made in Section 2 applies here: time can be modelled, when the computation of the network depends on event order or elapsed time, as in memory-based and time-encoding architectures, or merely supplied, when a timestamp or time-step index enters as an ordinary input attribute. The Elliptic dataset illustrates the second case, since the time step is one of the features of every transaction [16]; a static model trained on such features uses time without being temporally aware in our sense.

Only models of the first kind can represent the ordering constraints that laundering loops satisfy.

4.3.1. Discrete-Time Snapshot Models

The simplest temporal extension partitions the stream into snapshots and models their evolution. DySAT applies structural and temporal self-attention across snapshots [45], while EvolveGCN evolves the weights of a graph convolution with a recurrent network so that the model itself adapts over time [46]. Snapshot methods are simple to batch and parallelise, but they must choose a window granularity that trades temporal resolution against computational cost, and they blur the order of events within a window. For laundering, where the diagnostic signal is the precise order in which funds traverse a loop, that blurring can be the difference between observing and missing a cycle.

4.3.2. Continuous-Time Dynamic Graphs

Continuous-time models treat activity as an event stream and update node states exactly when events occur. A general continuous-time update is:

$$s_i(t) \;=\; \mathrm{UPDATE}_{\varphi}\big(\, s_i(t^{-}),\, m_i(t) \,\big) \tag{11}$$

where $s_i(t^{-})$ is the state just before the event and $m_i(t)$ an event message constructed from the partner state, transaction attributes, and the encoded time gap:

$$m_i(t) \;=\; \mathrm{MSG}_{\psi}\big(\, s_u(t^{-}),\, s_v(t^{-}),\, \varphi(\Delta t),\, r,\, a \,\big) \tag{12}$$

One family gives every node a recurrent memory and rewrites it whenever that node transacts [47], [64]. The arrangement fits mule accounts well, since a memory survives an arbitrarily long dormancy and is available the moment activity resumes. A second family stores no such state. When a query arrives it assembles a weighted summary over the node's past neighbours, with functional encodings of elapsed time supplying the weights [22], [65]. The benefit is selectivity: the handful of interactions that constitute a loop can be attended to over the many that do not. A third line abandons learned aggregation altogether in favour of anonymous temporal random walks that capture motif identity in an inductive, position-aware manner [66], which is appealing for cyclic structure because walks naturally traverse loops. Recent work has also shown that carefully designed simple architectures can match elaborate ones on standard dynamic benchmarks [67], and has characterised the expressive power that temporal message passing can and cannot attain [68]. A recurring engineering concern is memory staleness: a node inactive for a long interval carries an outdated state, which is addressed by time-decay terms that attenuate stale memory before it is reused.

4.3.3. Self-Supervised Temporal Objectives

Training these models typically uses a temporal link-prediction objective:

$$\mathcal{L}_{\text{link}} \;=\; -\sum_{(u,v,t)\,\in\,E} \Big[\, \log \sigma\big(\, \mathrm{dec}\big(z_u(t),\, z_v(t)\big)\,\big) \;+\; \log\big(\,1 \;-\; \sigma\big(\, \mathrm{dec}\big(z_u(t),\, z_{v'}(t)\big)\,\big)\,\big)\,\Big] \tag{13}$$

Given the interaction history up to time t, the model predicts which interaction occurs next, with negative sampling supplying contrastive non-edges v'. This self-supervised signal is valuable in fraud detection because it requires no labels and teaches the model the dynamics of normal interactions, against which abnormal bursts and loops may stand out. However, it is also the locus of a documented evaluation problem: uniformly sampled negatives are often too easy, and reported performance changes substantially under historical or inductive negative sampling [18], [69]. Any fraud detector that adopts this objective must therefore account for this evaluation pitfall.

4.3.4. Discrete Versus Continuous Trade-Offs

The discrete–continuous distinction has practical implications for fraud detection, as summarised in Table 6. Discrete-time models are generally easier to parallelise but depend on snapshot resolution, which may obscure the ordering of individual transactions. Continuous-time models preserve event-level temporal information but often require sequential state updates, increasing computational costs and limiting parallel processing. Temporal graph architectures also differ in how they represent historical interactions, ranging from snapshot-based updates and persistent memory to temporal attention and anonymous walks [70]–[72].

Despite these differences, the formulations in Eqs. (11) and (12) remain primarily discriminative and do not explicitly address incomplete graph observations. This limitation motivates the discussion of structural incompleteness in the next section.

Table 6. Temporal graph model families and their trade-offs for fraud detection

Family	Representative	Strength	Weakness
Snapshot attention	DySAT [45]	batchable, simple	window blurs event order
Snapshot recurrence	EvolveGCN [46]	adapts to drift	window granularity choice
Memory-based	TGN, JODIE [47], [64]	persists across dormancy	staleness, sequential updates
Temporal attention	TGAT, DyRep [22], [65]	inductive, exact times	cost grows with history
Anonymous walks	CAW [66]	motif-aware, inductive	sampling variance

5. Structural, Anomaly, and Generative Modelling

The models of Section 4 learn from the graph as it is observed. This section groups three lines of work that go beyond that. The first represents closed and higher-order structure explicitly and asks what happens when parts of the graph are missing. The second models normal behaviour and flags departures from it, which matters when labels are scarce and fraud is a minority. The third learns a distribution over graph structure itself, so that a detector can reason about edges that were never recorded. Together they address two limits identified in Section 4.2: the inability of message passing to count cycles, and the assumption that the observed graph is complete.

5.1. Structural Incompleteness, Cycles, and Motifs

5.1.1. Sources and Formalisation of Incompleteness

The graph an analyst actually holds is almost never the graph that exists, and four mechanisms account for the shortfall. Value crosses institutional boundaries that launderers select precisely because no single bank sees both sides. Data-protection law and commercial confidentiality forbid the pooling that would close those gaps. Reporting arrives late, or not at all. And shell companies and mixing services sever the link between an account and the party controlling it. Writing $E_o \subseteq E$ for what is observed, Eq. (14) defines the unobserved remainder together with the rate at which edges go unseen:

$$E_m = E \setminus E_o, \quad \rho = |E_m| / |E| \tag{14}$$

Structural evidence for fraud typically spans several hops and presupposes that the path carrying it is intact, so a deterministic test returns nothing at all once a required edge falls into Em. Work on learning from partial graphs responds with imputation, with joint structure learning, and with link prediction [17], [73], and transforming the graph at test time has been proposed as a way of adapting an already trained model to structure that is perturbed or incomplete [74]. Loops suffer worse than other patterns under these conditions, for the simple reason that a loop is defined by closure and closure is the first casualty of an unrecorded transfer. That asymmetry is the most forceful case that can be made for generative treatment: a detector ought to weigh the completions the data admits, not stake everything on the single graph it happens to hold.

5.1.2. Deterministic and Probabilistic Cycles

Let $\mathbb{1}[e \in E]$ be an edge-existence indicator. A directed cycle over nodes $(v_1, \dots, v_L)$ with closure $v_{L+1} = v_1$ is captured by:

$$\mathrm{Cyc}(v_1, \dots, v_L) = \prod_{k=1}^{L} \mathbb{1}\left[(v_k, v_{k+1}, r_k, t_k) \in E\right] \tag{15}$$

which equals one only if every required edge exists; temporal plausibility additionally requires bounded inter-event gaps: $0 < t_{k+1} - t_k \leq \Delta_{\max},\ k = 1, \dots, L-1$ and monotone ordering.

$$\Phi(c) = \prod_{k=1}^{L} p_\theta(e_k) \in [0, 1] \quad (16)$$

Combining Eq. (15) with Eq. (14) reveals the brittleness: a single edge in Em forces the product to zero, even for a true laundering loop. Replacing the hard indicators with a learned edge-plausibility model $p_\theta(e)$ yields the soft cycle score Eq. (16), a continuous value in [0,1] that degrades gracefully under missing edges and supports near-cycle reasoning. The score is also differentiable in θ, so it can be optimised rather than merely evaluated, which is what makes it usable as a learning or steering signal rather than as a post-hoc filter.

5.1.3. Closed-Walk and Spectral Relaxations

An older and quite different relaxation reaches cyclicity through walk counting. Raise a weighted adjacency matrix A to the power L and its i-th diagonal entry records how many closed walks of that length pass through node i; summing the diagonal gives:

$$\mathcal{C}_L(A) = \mathrm{tr}(A^L) = \sum_i (A^L)_{ii} = \sum_i {\lambda_i}^L \quad (17)$$

equivalently the L-th power sum of the spectrum. Nothing needs to be enumerated to obtain it, it varies smoothly, and it rises wherever short loops are present. The same kernel underpins subgraph centrality [75], which situates the fraud problem within the older study of motifs and higher-order structure in networks [76], [77]. Its decisive property here is differentiability with respect to A, which permits it to be optimised or used to steer rather than merely computed, and which avoids the combinatorics of search entirely. The concession is that a closed walk need not be a simple cycle, so the statistic inflates where neighbourhoods are dense. For a soft objective that inflation is tolerable, and discounting degenerate walks that merely retrace an edge removes part of it.

A closely related construction from causal structure learning is instructive. To enforce acyclicity, the NO TEARS formulation replaces the combinatorial constraint with the smooth algebraic condition Eq. (18), where $\circ$ is the Hadamard product and d the number of nodes [78], [79]. The fraud problem is the mirror image: instead of driving $h(A)$ to zero to forbid cycles, a detector wants to reward the presence of cycles, so the same matrix-exponential trace becomes an objective to maximise rather than a constraint to satisfy. That symmetry is, to our knowledge, under-exploited in the fraud literature and is one of the more concrete openings this review identifies.

$$h(A) = \mathrm{tr}(\, e^{A \circ A}\,) - d = 0 \quad (18)$$

5.1.4. From Static Counts to Continuous-Time Monitoring

Both relaxations above are stated on a single, static adjacency, which quietly reintroduces the snapshot assumption that Section 4.3 spent its length dismantling. Laundering loops are not standing structures; they open and close inside a bounded interval, and a loop whose hops are spread over months is not the same object as one closed within a day. A monitor must therefore evaluate cyclicity on the sub-graph induced by a trailing window:

$$M_i(t) = \sum_{L \in \mathcal{L}} w_L \left(A^L_{[t-\Delta,\, t]} \right)_{ii} \quad (19)$$

where $A_{[t-\Delta,t]}$ is the adjacency of events in $[t-\Delta, t]$ and the weights w_L concentrate the statistic on the short lengths that layering favours, for instance: $w_L \propto 2^{-L}, \quad L \in \{3,4,5\}$. Read this way the quantity is no longer a descriptive graph statistic but a per-node signal defined continuously in transaction time, which is what an alerting system actually consumes.

Three properties make Eq. (19) attractive as an operational monitor, and one caveat constrains it. It is incremental: an arriving event touches one row and one column of A, so the affected diagonal entries can be refreshed by sparse updates rather than by recomputing the trace. It is bounded in cost independently of how many cycles the window contains, since matrix powers are obtained by repeated squaring, in contrast to the combinatorial growth of enumeration. And it is governed by a distribution whenever the adjacency is populated with the edge plausibilities of Eq. (16) or with weights derived from a nodal model Eq. (4), so that

missing edges attenuate the statistic smoothly rather than annihilating it. The caveat is the one already noted, that closed walks are not simple cycles, so dense benign hubs inflate the count and a monitor of this kind should be read against a per-node baseline rather than an absolute threshold. No surveyed system, to our knowledge, deploys such a monitor as a continuously updated, distribution-weighted statistic, which remains an open problem (Section 9).

5.1.5. Motif Counting and Sub-Graph Awareness

Cyclicity is one instance of the broader problem of motif awareness. As noted in Section 4.2, standard message passing is provably limited in its ability to count substructures [24]. For fraud, the motifs of interest, namely rings, fans, and dense bipartite cores, are well characterised, so methods that make these structures directly observable to the model are attractive, whether through walk statistics [66], spectral closed-walk measures [75], explicit subgraph isomorphism counts [53], or higher-order message passing [54]. Table 7 compares the available relaxations on the axes that matter for deployment.

Table 7. Relaxations of cycle and motif detection. n is the number of nodes, m the number of edges, L the cycle length, and d the average degree

Approach	Cost	Differentiable	Robust to missing edges
Explicit enumeration	$O(m\, d^{L-1})$	no	no
Temporal motif counting	$O(m\, d^{L-1})$ with time filter	no	no
Soft cycle score, Eq. (16)	$O(L)$ per candidate	yes	yes
Closed walks, Eq. (17)	$O(L\, m)$ by sparse products	yes	partial
Matrix exponential, Eq. (18)	$O(n^3)$ dense, less if sparse	yes	partial
Subgraph count features [53]	precomputed, offline	no	no

5.2. Graph Anomaly Detection and Imbalance-Aware Learning

Cycle-aware methods target one known pattern. Much fraud, however, is not known in advance, and labels are scarce, delayed, and heavily imbalanced. Graph anomaly detection addresses this by modelling what normal structure and behaviour look like and scoring departures from it, and imbalance-aware learning addresses the scarcity of positive examples directly.

5.2.1. Reconstruction-Based Detection

Once labels become scarce, the boundary between fraud detection and graph anomaly detection is largely nominal [80], [81]. One long-standing approach compresses the attributed graph through an autoencoder and treats whatever fails to survive the round trip as suspect. DOMINANT is the canonical instance: structure and attributes are reconstructed together, and each node is ranked by the weighted residual:

$$s(i) = (1-\alpha) \parallel \mathrm{A}_{i:} - \hat{\mathrm{A}}_{i:} \parallel_2 + \alpha \parallel x_i - \hat{x}_i \parallel_2 \tag{20}$$

where α weights the feature-reconstruction residual and $1-\alpha$ weights the structural-reconstruction residual [82]. The bridge to generative modelling needs no elaborate construction. A model that has internalised what ordinary relational structure looks like will assign whatever fraud produces either a low likelihood or a large residual, these being two readings of the same quantity.

5.2.2. Self-Supervised and Contrastive Objectives

Where labels are scarce, contrastive and self-supervised objectives have largely displaced reconstruction as the method of choice. The construction is to set a node against deliberately corrupted versions of the context surrounding it; what the model acquires from this is a sense of what agreement between a node and its neighbourhood normally looks like, and an anomaly is then whatever fails to exhibit it [83]–[86]. Broader surveys of graph self-supervision [87] and of pre-training strategies [88]–[90] map the design space. These objectives are attractive in AML, where confirmed labels are scarce and delayed, because they exploit the abundant unlabelled structure directly.

5.2.3. A Spectral View of Anomaly

A distinct and clarifying perspective analyses anomaly through the graph spectrum. Anomalies induce a right shift of the spectral energy toward higher frequencies, because an anomalous node disagrees with its neighbourhood, and low-pass aggregation therefore suppresses exactly the signal one wants to detect [91]. Beta wavelet filters that are band-pass rather than low-pass recover this signal, and related work shows that heterophily and spectral shift are two views of the same phenomenon [92]. This connects directly to Section 4.2: the reason generic GNNs underperform on fraud is not merely imbalance, it is that their inductive bias is smoother, and fraud lives in the roughness. A subtlety specific to fraud is that "normal" is itself heterogeneous, since legitimate behaviour spans retail, corporate, and institutional patterns, so an effective normality model must be multimodal, which again favours expressive generative families over unimodal density estimators.

5.2.4. Imbalance at the Data, Loss, and Structure Levels

Fraud is a structural minority, so imbalance handling is central, and remedies operate at three levels. At the data level, classical resampling such as SMOTE [93] was adapted to graphs by interpolating minority nodes in embedding space and synthesising the edges to attach them [94], while ego-network synthesis constructs whole neighbourhoods rather than isolated nodes [95]. At the loss level, cost-sensitive weighting and focal-style losses, Eq. (21), down-weight easy negatives so that gradient signal concentrates on the hard, fraud-like examples [96]. Both families are simple and effective but address the label imbalance while leaving the structural imbalance untouched. At the structure level, surveys argue that the deeper issue is precisely this message imbalance, in which the relational context that drives representation learning is overwhelmingly benign [14], and pick-and-choose sampling addresses it by reshaping the neighbourhoods themselves [44].

$$\mathcal{L}_{\text{focal}} = -\alpha_y \left(1 - p_y\right)^{\gamma} \log p_y \tag{21}$$

5.2.5. Generative Augmentation

If the deficit is structural, then so should the augmentation be: what a detector wants is additional topology consistent with fraud, retaining the rings and activity bursts that characterise it, not simply a less lopsided label distribution. Score-based diffusion over graphs suits that requirement because it models a distribution over whole relational configurations, which is a different operation from interpolating between two points in an embedding space. Whether it delivers remains genuinely unresolved. Synthetic structure might teach a detector to recognise real laundering, or it might teach it only to recognise the generator, and the two are indistinguishable unless testing is done on real typologies withheld from training. Benchmarks scored on how good the samples look cannot settle the matter, since sample quality and downstream utility are separate quantities. Table 8 summarises the anomaly families and the signal each exploits.

Table 8. Graph anomaly detection families and the signal each exploits

Family	Representative	Signal exploited	Labels needed
Reconstruction	DOMINANT [82]	residual of structure and attributes	none
Contrastive	CoLA [84]	node–context disagreement	none
Generative–contrastive hybrid	[85], [86]	both of the above	none
Spectral	BWGNN [91]	high-frequency energy shift	few
Heterophily-aware	[50], [92]	neighbourhood dissimilarity	few
Supervised fraud GNN	PC-GNN [44]	labelled minority structure	many

5.3. Score-Based Diffusion and Generative Graph Inference

Reconstruction-based anomaly detectors already use a generative idea, scoring what a model cannot reproduce. Score-based diffusion makes this idea explicit and probabilistic: it learns a distribution over graphs or graph states and can sample completions of a partially observed graph. Among the generative families of Section 4.1, it is the one whose value for fraud lies least in realistic samples and most in calibrated reasoning over plausible, possibly

unobserved, structure. This makes it the natural route to probabilistic generation, the second of the four required properties, and to reasoning about edges that were not observed.

5.3.1. Foundations

Diffusion models originate in non-equilibrium thermodynamics [97] and were refined as denoising diffusion probabilistic models [37]. In continuous time, a forward stochastic differential equation, Eq. (22), transports data to noise, where f is the drift, g the diffusion coefficient, and w a Wiener process [38]. Two parameterisations dominate. The variance-exploding family lets the noise variance grow without bound, while the variance-preserving family, Eq. (23), keeps the marginal variance bounded, which is desirable when weak structural-minority signals must survive corruption.

$$dx = f(x,t)dt + g(t)dw \tag{22}$$

$$dx = -\frac{1}{2}\beta(t)\,x\,dt + \sqrt{\beta(t)}\,dw \tag{23}$$

The corresponding reverse-time process, Eq. (24), depends on the score of the marginal density [98] and is learned by denoising score matching, Eq. (25), whose weighting λ(t) controls conditioning and sample quality [99]–[102]. The practical significance for detection is that the learned score is a gradient field on the data manifold, so it supplies not only samples but a local geometry against which an observed configuration can be judged unusual.

$$dx = [\,f(x,\,t) - g^2(t)\,\nabla_x \log p_t(x)\,]\,dt + g(t)\,d\overline{w} \tag{24}$$

$$\mathcal{L}(\theta) = \mathbb{E}_t\,\lambda(t)\,\mathbb{E}\big[\,\|\,s_\theta(x_t,\,t) - \nabla_{x_t}\log p_t(x_t \mid x_0)\,\|^2\,\big] \tag{25}$$

5.3.2. The Noise Schedule as a Signal-to-Noise Sweep

It is easy to read the forward process as mere data destruction. A more useful reading, and the one that connects diffusion to detection, is that the schedule sweeps a signal-to-noise ratio. The variance-preserving marginal is $x_t = \alpha(t)x_0 + \sigma(t)\epsilon$, with $\sigma(t)^2 = 1 - \alpha(t)^2$, where the ratio of retained signal power to injected noise power is:

$$\mathrm{SNR}(t) = \frac{\alpha(t)^2}{\sigma(t)^2} \tag{26}$$

a quantity that falls monotonically from ∞ at $t = 0$ to 0 at $t = 1$. Because α and σ are fixed by $\beta(\cdot)$, choosing a schedule is choosing how the sweep is distributed, and results on schedule design are best understood as statements about which parts of the range deserve the most training effort [37], [101].

The sweep has a direct and, for this review, central consequence. Suppose the nodal states of Eq. (4) are approximately Gaussian within each class, with means μ_y and common isotropic scale s. Then the diffused state is also Gaussian in each class, $x_t \mid y \sim \mathcal{N}\big(\alpha(t)\mu_y,\,\alpha(t)^2\Sigma + \sigma(t)^2 I\big)$, and the deflection coefficient that governs the best achievable separation between the two classes at diffusion time t is:

$$d^2(t) = \alpha(t)^2(\mu_1 - \mu_0)^\top(\alpha(t)^2\Sigma + \sigma(t)^2 I)^{-1}(\mu_1 - \mu_0) \tag{27}$$

which is non-decreasing in the signal-to-noise ratio for any within-class covariance. Where the covariance is isotropic, $\Sigma = s^2 I$, this reduces to the scalar form:

$$d^2(t) = \frac{\alpha(t)^2\|\mu_1 - \mu_0\|_2^2}{\alpha(t)^2 s^2 + \sigma(t)^2} \tag{28}$$

The distinction matters in practice, because the scalar form is markedly optimistic on learned representations, whose covariance is far from isotropic. Two limits are worth stating plainly. As $t \to 0$ the ratio tends to $\|\mu_1 - \mu_0\|_2^2/s^2$, the detectability of the clean problem; as $t \to 1$ it tends to zero, and the classes are indistinguishable. Between them, d^2 is strictly increasing in the signal-to-noise ratio, so a single trained diffusion furnishes a continuum of detection problems indexed by t, of graded and analytically known difficulty.

Three practical readings follow, and none of them is available to a discriminative model. First, the corruption level at which a class separation collapses measures how robust that separation is; a fraud signature that survives to low signal-to-noise is one an adversary must work hard to erase. Second, guidance and denoising are most valuable in the middle of the range, where the prior still constrains the state but the observation no longer determines it, which is a design argument for guided rather than unguided recovery. Third, the variance-preserving family is preferable to the variance-exploding one for a structural minority precisely because Eq. (26) decays gracefully rather than being driven to zero by an unbounded denominator, so a weak signal is given a long interval over which it remains recoverable. This signal-to-noise view of the schedule is standard in the generative-modelling literature but is rarely carried into fraud detection, where it supplies exactly the vocabulary the problem needs.

5.3.3. Sampling, Likelihood, and Acceleration

A practical concern for any detector built on diffusion is the cost of the reverse process, which naively requires many sequential network evaluations. Two developments make diffusion viable under latency budgets. First, the probability-flow ordinary differential equation associated with Eq. (24), given by Eq. (29), shares the same marginals as the stochastic reverse process but is deterministic, enabling fast, fewer-step integration and exact likelihood evaluation by the instantaneous change-of-variables formula [38]. This is attractive precisely when a calibrated anomaly score is required, because a likelihood is a more defensible quantity than an unnormalised residual. Second, implicit deterministic samplers [103] and higher-order solvers [104] cut the step count by roughly a factor of ten while preserving most of the quality. Detection tolerates coarse sampling: a rough trajectory suffices to read off a residual or a guidance gradient, so the step budget can be cut in favour of throughput.

$$dx = \left[f(x, t) - \frac{1}{2} g^2(t)\, s_\theta(x, t) \right] dt \tag{29}$$

5.3.4. Guidance as an Explicit Structural Objective

A key capability for detection is conditional generation. Classifier guidance steers an unconditional model with the gradient of an auxiliary classifier or energy, Eq. (30), keeping the conditioning objective explicit and inspectable [105], whereas classifier-free guidance folds the condition into the generator and trades auditability for sample efficiency [106]. For regulated fraud detection, where the basis of an alert may have to be justified, explicit guidance objectives are attractive because the steering signal doubles as evidence. In particular, the differentiable cyclicity measure of Eq. (17) can in principle be used directly as the guidance term, Eq. (31), so that the reverse process is steered toward completions that close loops. To our knowledge, no published system combines a continuous-time transaction encoder with a cycle-guided reverse process in this way, and Section 9 lists it among the open problems.

$$\hat{s}_\theta(x, t) = s_\theta(x, t) + \gamma \nabla_x \log p_t(y \mid x) \tag{30}$$

$$g_c(x) = \nabla_x \log C_L(A(x)) \tag{31}$$

5.3.5. Graph Diffusion

Table 9. Graph diffusion formulations and their suitability for transaction-scale fraud inference

Method	State space	What is diffused	Scalability concern
EDP-GNN [107]	continuous	adjacency entries	quadratic in n
GDSS [108]	continuous	features and structure jointly	quadratic in n
DiGress [109]	discrete	categorical edge types	quadratic in n
Discrete-space analysis [110]	discrete	binary adjacency	quadratic in n
Autoregressive diffusion [111]	discrete	ordered node insertions	ordering sensitivity
Latent diffusion [113]	continuous	compact latent code	encoder fidelity

Adapting diffusion to graphs requires perturbing structure rather than pixels. Permutation-invariant score-based generation perturbs adjacency matrices and learns a permutation-equivariant score [107]; GDSS models node features and structure jointly through a system of stochastic differential equations [108]; DiGress uses discrete denoising tailored to the

categorical nature of edges [109]; and the benefit of discrete state spaces for graphs has been analysed directly [110]; and autoregressive diffusion combines ordering with denoising [111]. Surveys consolidate this rapidly growing area [112]. These methods preserve structural statistics such as motif distributions and sample over plausible relational configurations rather than committing to a single completion, directly addressing the incompleteness of Eq. (14). Table 9 contrasts the principal graph-diffusion formulations.

5.3.6. Latent Diffusion and the Route to Scale

A practical caveat is scale: diffusing raw adjacency entries is quadratic in the number of nodes, so financial-scale methods must operate on compact latent encodings of windowed subgraphs. The latent-diffusion strategy is to encode a subgraph into a low-dimensional relational state, diffuse there, and decode only as needed [113]. That bridge is what makes graph diffusion tractable for transaction networks. It also aligns with the continuous-time encoders of Section 4.3, since the latent state can fuse node attributes, temporal memory, and structural summaries into one object on which both generation and detection operate. Figure 5 sketches the resulting pipeline. The same latent route makes the structural augmentation of Section 5.2 feasible at transaction scale.

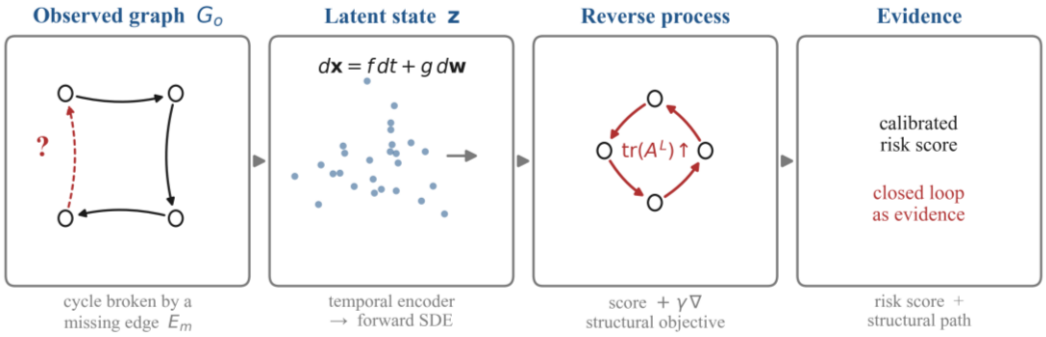


Figure 5. Generative inference over an incompletely observed transaction graph. A continuous-time encoder maps a windowed sub-graph to a latent state; a forward stochastic differential equation corrupts it; and a reverse process, steered by a differentiable structural objective, produces both a calibrated risk score and the closed loop that justifies it

6. Applications, Robustness, Explainability, and Deployment

The preceding sections reviewed methods largely as they are evaluated in research. This section turns to their use: how graph-based methods have been applied to anti-money-laundering problems, and the concerns of robustness, explainability, scalability, and privacy that decide whether a method that performs well on a benchmark can be deployed.

6.1. Graph-Based Anti-Money-Laundering Systems

6.1.1. The Elliptic Line of Work

A focused line of applied work targets anti-money laundering directly. The Elliptic study popularised graph convolutional detection of illicit Bitcoin transactions and established the temporal-split protocol now standard for the dataset [16]. Subsequent work explored supervised comparisons on the same data [114], the label-scarcity regime that characterises real AML operations [115], distance-to-illicit-node features derived from random walks [116], and self-supervised node embeddings for the same task [117]. Collectively this line has produced the field's most reproducible results, but it has also concentrated attention on a single, attribute-rich, transaction-as-node graph whose idiosyncrasies propagate into every method tuned on it.

6.1.2. Synthetic and Simulated AML Benchmarks

Because real cyclic-fraud labels are scarce, synthetic AML simulators provide ground-truth laundering typologies such as circular flows and layering [118]. Their value is a controlled study of structural reasoning, particularly when illicit and benign nodes are made attribute-identical so that detection must rely on topology and timing; their limitation is realism. More recent large-scale synthetic corpora were constructed specifically to provide realistic multi-bank transaction streams with known laundering patterns [119], and accompanying work

shows that architectures respecting the directed multigraph nature of the data, rather than collapsing it to a simple graph, are provably and empirically stronger on these patterns [120]. That result deserves emphasis: it is direct evidence that the representational choices of Section 3.7, not merely model capacity, govern whether cyclic typologies can be detected at all.

6.1.3. Self-Supervised and Heterogeneous AML Models

Because confirmed labels arrive late and sparsely, self-supervision is particularly attractive in AML. LaundroGraph learns representations of the customer–transaction bipartite graph without labels and surfaces suspicious behaviour for analyst review [121]. Heterogeneous graph neural networks applied to real bank data have been shown to improve on production baselines by modelling multiple node and edge types jointly [122]. Graph-mining approaches detect laundering networks such as Hawala by surfacing dense or circular sub-structures [6], and market-integrity work detects wash trading through its circular trade signatures [5], [7]. On the learning side, camouflage-resistant and imbalance-aware fraud detectors [15], [43] and [44] address the adversarial and minority aspects of financial crime specifically.

6.1.4. Operational Realities

Two facts about how AML actually operates receive far less attention in the research literature than their influence warrants. One is fragmentation across institutions. Adversaries route value through the joints between banks precisely so that no single institution holds more than a fragment of the trail. Because those institutions cannot exchange records freely, the fragments are never reassembled. This is the direct origin of the unobserved edges in Eq. (14), and the motivation behind work on federated and privacy-preserving analytics [123]. The other is orientation toward typologies. Supervisory thinking is organised around named patterns, structuring, layering, round-tripping, so a system that returns a recognisable one of these is easier both to embed in an existing workflow and to justify afterwards than a system returning an uninterpretable score. Running through the literature is a consistent pattern: what is actually deployed remains discriminative and snapshot-based, and cyclicity, continuous time, and generative reasoning appear only in isolation, when they appear at all.

Four cross-cutting concerns decide whether a method that performs well on a benchmark can be deployed at all. They are grouped here because they interact: a defence that triples inference cost fails the scalability constraint, and an explanation mechanism that requires centralised data fails the privacy constraint.

6.2. Adversarial Robustness

Fraud is adversarial by definition, so robustness is not optional. Graph structure can be attacked directly: Nettack perturbs edges and features to flip a target node's prediction with few, unnoticeable changes [124]; meta-learning attacks poison the training graph itself [125]; and reinforcement-learning attackers operate in the black-box setting [126]. Camouflage is the fraud-specific instance, in which illicit actors add benign-looking edges to dilute their suspicious neighbourhood [43]. Surveys catalogue both attacks and defences [57]. Defences span several strategies: robust aggregation that down-weights anomalous neighbours, structure learning that jointly cleans the graph while training [73], adversarial training, and certifiable methods that guarantee behaviour within a bounded perturbation budget. Each trades clean-data accuracy or computational cost for resilience, and none resolves the adaptive nature of fraud. A salient point for this review is that probabilistic, structure-modelling approaches may be inherently more robust to camouflage than point-estimate classifiers, because a small number of injected benign edges is less likely to perturb reasoning over a distribution of plausible structures. That hypothesis requires systematic empirical validation under realistic, adaptive threat models rather than static benchmark attacks.

6.3. Explainability and Regulatory Acceptance

Regulatory deployment requires faithful explanations. Post-hoc graph explainers identify the sub-graph and features most responsible for a prediction: GNNExplainer maximises mutual information between a compact sub-graph mask and the output [127]; PGExplainer learns a parameterised, inductive mask generator [128]; subgraph exploration methods attribute predictions to discrete substructures [129]; and probabilistic graphical-model explanations provide an alternative formalism [130]. Gradient- and perturbation-based saliency methods provide further attributions [131], and taxonomic surveys map the space [132]. Domain-

specific systems have also been built, notably explainable fraud-transaction detection deployed at scale [133].

The common limitation is that these explanations are approximations decoupled from the model's own reasoning, which weakens their evidentiary value when an alert must be justified to a regulator or in court, and which motivates the long-standing argument that high-stakes decisions should rely on intrinsically interpretable models rather than post-hoc rationalisations [134]. A distinct and operationally valuable mode is the counterfactual explanation, which identifies the minimal change to the input, typically the removal of a few edges, that would flip the decision [135]; for an AML alert, the answer "these three transfers close the loop" is directly actionable for an investigator. Evaluating explanations is itself non-trivial, since fidelity, sparsity, and stability are often in tension. Generative formulations offer a different route in principle, because sampling trajectories and guidance signals are intrinsic by-products of inference and can therefore serve as structural evidence paths rather than being reconstructed after the fact. Whether that potential is realised in practice remains an open empirical question.

6.4. Scalability and Systems Considerations

A set of constraints that benchmark studies seldom mention turns out, in production, to decide which methods can be used at all. Consider first the scale of training. At hundreds of millions of nodes, a full-batch pass is simply not available, so one samples neighbours [41], partitions into clusters [136] or samples sub-graphs [137], every option surrendering a measured quantity of information to buy feasibility. Consider next inference on a stream. Payments do not wait, and each must be scored inside a fixed latency envelope, which rewards architectures carrying incremental state, memory-based temporal networks being the obvious case, and penalises anything that must rebuild a neighbourhood embedding from scratch on every query. Then there is keeping the model current. Behaviour moves, so retraining or fine-tuning is a recurring obligation, and the expense of that cycle, including how long labels take to arrive, belongs in any honest accounting of what a method costs. Finally there is amortisation: a per-query cost that does not scale with the number of structures being looked for, as enumeration-free cyclic measures rather than explicit cycle search are strongly preferred because their cost does not explode in the dense hubs where fraud concentrates. In practice, these pressures are reconciled by screening in tiers, passing the whole stream through an inexpensive model and reserving the costly one for what that first stage flags. Taken together, they go some way toward explaining why leaderboard position and production viability correlate so weakly.

6.5. Privacy-Preserving and Federated AML

Laundering is engineered to straddle institutional boundaries, with the consequence that the fullest picture of any laundering network exists nowhere in one place. Closing that gap by pooling raw records is what privacy law and commercial confidentiality between competitors both forbid. Federated learning shares model updates rather than records [138], [139], and differential privacy provides the formal guarantee under which such sharing can be defended [140]; early work has applied this combination to collaborative financial-crime detection [123]. The connection to the rest of this review runs through Eq. (14): to a large extent, the edges recorded there as unobserved are the edges that fall across institutional seams. Collaborating without disclosure and reasoning probabilistically about unseen structure are therefore two answers to a single question, and neither is yet well developed in the AML literature.

7. Evaluation Methodology, Benchmarks, and Pitfalls

Sound conclusions depend as much on protocol as on model design. This section reviews the resources the field uses, the metrics that are meaningful under extreme imbalance, and the pitfalls that recur often enough to warrant explicit guidance.

7.1. Benchmarks

Table 10 summarises the most-used resources. The Elliptic Bitcoin dataset is the reference point for work on real data. It holds roughly 200,000 transaction nodes, each carrying a timestamp, of which only a small fraction are illicit. Since time is explicit in the data, the accepted protocol fits on the earlier steps and evaluates on the later ones, which is what keeps

future information out of the model. The dataset also carries a well-documented discontinuity: after a dark-market takedown every method degrades on the closing steps, and honest reporting means showing that decline rather than smoothing it into an average [16]. Synthetic AML simulators provide ground-truth laundering typologies [118], [119]. Multi-relation graphs derived from review or financial platforms provide a third setting with relation-typed neighbourhoods [15], [43], and general graph-learning benchmarks supply standardised splits and leaderboards for the underlying architectures [69], [141] and [142].

Table 10. Representative benchmarks for graph-based fraud detection and their salient properties

Resource	Type	Salient property	Cyclic labels
Elliptic [16]	real, temporal	extreme imbalance; post-shutdown shift	no
Elliptic++ [143]	real, transactions and addresses	address graph alongside transactions	no
Elliptic2 [144]	real, subgraph labels	labelled laundering subgraphs	no
AMLSim [118]	synthetic	ground-truth cyclic typologies	yes
AMLworld [119]	synthetic, multi-bank	realistic scale and typologies	yes
Amazon / YelpChi [43]	real, multi-relation	relation-typed neighbourhoods	no
BOND [142]	benchmark suite	unsupervised outlier detection	no
TGB [69]	benchmark suite	temporal splits, harder negatives	no

Two extensions of Elliptic widen what it can test. Elliptic++ adds a graph of more than 800,000 wallet addresses alongside the transaction graph [143], and Elliptic2 provides labels for about 122,000 subgraphs rather than single nodes, within a background graph of some 49 million entity clusters [144]. Neither carries labels for cyclic typologies, but the address-level and subgraph views allow structures, including loops, that the transaction-level graph cannot contain.

7.2. Metrics Under Extreme Imbalance

Metric choice is consequential under extreme imbalance. Plain accuracy is uninformative, so the literature reports ROC-AUC for ranking quality and precision–recall summaries for the minority regime, and where positives are rare the latter carries far more information than the former [145], [146]. Recall measured at a fixed review budget performs a different job again, converting an ordering into a statement about how much work an investigations team must absorb. Calibration receives less attention than it deserves. Whenever scores are used to rank cases by risk instead of driving one hard accept-or-reject decision, whether those scores behave like probabilities becomes material, and deep networks are known not to by default [147]. Table 11 states the question each metric answers.

Table 11. Evaluation metrics and the question each answers under extreme class imbalance

Metric	Question answered	Caveat
ROC-AUC	are positives ranked above negatives?	optimistic when positives are rare
PR-AUC / AP	is precision sustained across recall?	depends on the base rate
Recall@k	what yield fits an analyst review budget?	requires a realistic k
Precision@k	what fraction of alerts are actionable?	same
FPR	what cost falls on legitimate customers?	must be read with recall
Brier / ECE	are the scores calibrated probabilities?	needs sufficient positives

7.3. Feature Saturation and Comparison with Conventional Models

Feature saturation occurs when the attributes of a dataset already carry most of the discriminative signal, so that a model's use of the graph has little left to add. Elliptic is the clearest case. Each transaction has 166 features: the first 94 describe the transaction itself, including its time step, and the remaining 72 aggregate information from its one-hop neighbourhood [16]. Part of the relational signal that a graph model would learn is therefore already supplied as tabular input. Consistent with this, a random forest on these features outperformed a graph

convolutional network in the original study, and the forest improved further when node embeddings from the graph network were added to its features [16]. Later work on the same data likewise found tree-based classifiers strong and obtained good results by combining graph-derived embeddings with them [114], [117].

Saturation has three implications for comparing graph-based methods with conventional machine learning. First, a comparison that includes only graph models can show progress among them while every one of them remains below a well-tuned tree ensemble, so a strong attribute-only baseline, such as a random forest or gradient-boosted trees [25]–[27], is needed before any claim that the graph adds value. Second, on saturated data a graph model that matches the tabular baseline has not failed, because the dataset leaves little for structure to explain; the result describes the data rather than the method. Third, a structural claim can be tested only where the signal is structural, which calls for benchmarks in which attributes are deliberately uninformative or in which the structure that defines fraud is known, as in synthetic AML generators [118], [119].

7.4. Pitfalls That Inflate Reported Performance

Feature saturation is one of several practices that can make a model appear stronger than it is. Each of the following has a direct effect on reported numbers, and each corresponds to an item of the checklist in Table 12.

1. Temporal leakage. A feature, normalisation statistic, or data split that uses information from after the prediction time, as defined in Eq. (3), lets the model see the future. Random splits of a transaction graph cause it through entities shared between training and test periods, which is why the standard protocol for Elliptic trains on earlier time steps and evaluates on later ones [16].
2. Threshold selection on test data. Choosing the decision threshold that maximises F1 on the test set converts a small gain in ranking quality into a larger gain in F1; thresholds should be chosen on training or validation data.
3. Quoted rather than retrained baselines. Baseline numbers taken from other papers were obtained under different graph constructions, splits, neighbour sampling, and tuning. Because graph fraud results are sensitive to all of these, a quoted baseline is not a controlled comparison.
4. Easy negative samples. In temporal link prediction, evaluation against randomly sampled negative edges overstates performance. A pure memorisation baseline, EdgeBank, performs surprisingly well under this protocol because the negative edges are easy, and harder historical and inductive negative sampling strategies were proposed in response [18]. The Temporal Graph Benchmark adopted harder negatives together with fixed chronological splits [69].
5. Limited variance reporting. Positive examples are scarce, so a few cases move F1 appreciably, and a single random seed or a single pooled test window can hide large variation. On Elliptic, every method performs poorly after a dark-market closure at time step 43, a change that one average over the test period conceals [16].

These effects compound. A study that combines a random split, a test-set threshold, and quoted baselines on a saturated dataset can report an improvement that a controlled evaluation would not reproduce. We therefore treat evaluation practice as one of the principal findings of the review (Section 8.1) rather than as a methodological footnote.

7.5. A Reproducibility Checklist

Choosing the right metric is necessary but not sufficient; three further habits separate an evaluation one can trust from one that merely looks convincing, and Table 12 gathers them for reference. The first concerns attribution of credit. A method that bundles several novelties cannot report which of them earned the improvement unless each is switched off in turn, and the classifier sitting on top of the representation has to be frozen throughout, or a stronger head will quietly absorb the credit. The second concerns uncertainty. Positives are so scarce that a difference of a few examples moves F1 appreciably, which makes intervals over random seeds and over time windows a requirement rather than a courtesy, and which makes a single pooled figure a poor summary of per-window behaviour. The third concerns disclosure. How the graph was constructed, how entities were resolved, where the temporal boundaries fell,

how thresholds were chosen, how negatives were sampled, and how baselines were tuned all need to be stated, because any one of them can move the same model between apparent strength and apparent weakness.

Table 12. Reproducibility checklist for graph-based fraud-detection studies, with the pitfall each item addresses

Item	Requirement	Pitfall addressed
Graph construction	schema, entity resolution rule, and edge semantics stated	quoted baselines
Temporal split	train, validation, and test time ranges given explicitly	temporal leakage
Feature causality	all features computed from data available at prediction time	temporal leakage
Attribute-only baseline	strong tabular model reported alongside graph models	feature saturation
Baselines	re-trained under the identical split, never quoted from prior work	quoted baselines
Thresholds	selected on training or validation data only	threshold selection on test data
Negative sampling	strategy stated; historical or inductive negatives reported	easy negative samples
Variance	results across seeds and time windows with confidence intervals	limited variance reporting
Ablations	one component varied at a time with the classifier head fixed	misattributed credit
Drift	per-window results reported, not only a pooled score	limited variance reporting

8. Comparative Synthesis and Property-Centric Taxonomy

8.1. Principal Findings

This subsection states the four principal findings of the review. For each, it gives the evidence from the reviewed studies, the pattern that the evidence shows, our interpretation, and the implication for research. The studies differ in datasets, graph construction, data splits, and metrics, so the findings are qualitative: they rest on consistent directions of effect across studies, not on numbers aggregated across papers (Section 2). Table 13 summarises them.

Finding 1: no single approach is best under all conditions. Evidence. On Elliptic, whose attributes include aggregates of each transaction's neighbourhood, a random forest outperformed a graph convolutional network, and the forest improved further with graph-derived node embeddings [16]; later studies on the same data also found tree-based classifiers strong [114], [117]. On multi-relation review and transaction graphs with camouflage and heterophily, fraud-specific GNNs outperform generic GNNs [15], [43], [44] and [50]. On synthetic AML data, where laundering is defined by structural patterns, adapting message passing to directed multigraphs improved the minority-class F1 of standard GNNs by up to 30% [120]. Temporal models outperform static ones on interaction graphs whose dynamics carry the signal [22], [47] and [64], whereas on Elliptic a temporal model outperformed its static counterpart only slightly [16], [46]. Pattern. The ranking of method families changes with the dataset, and in a consistent direction: tree-based models lead where attributes are rich, graph models where the signal is relational, and temporal models where it is sequential. Interpretation. Effectiveness is a property of the match between a method and where the discriminative information resides, not of the method alone. Implication. A claim of superiority needs a diagnosis of where the signal lies and a comparison with the family suited to that location, including a strong attribute-only baseline.

Finding 2: cyclic structure is a representational problem. Evidence. Message-passing networks are bounded in expressiveness by the one-dimensional Weisfeiler–Lehman test and cannot count cycles or many other substructures [23], [24]. Models that represent substructures or higher-order cells explicitly recover part of this ability [53], [54]. In AML, the largest reported gains on cyclic and other structural typologies came from changing what the network represents, through port numbering, ego identifiers, and reverse message passing, rather than from deeper or wider models [120]. Pattern. Improvements on cyclic fraud follow changes in representation, not in model capacity. Interpretation. Cycle awareness has to be built into the representation or supplied as a feature. Implication. Detectors for laundering loops should be assessed for the structure they can represent, not only for their accuracy on benchmarks in which loops may be absent.

Finding 3: evaluation practice can inflate reported performance. Evidence. Under random negative sampling, a pure memorisation baseline performs surprisingly well in temporal link prediction, which led to harder negative-sampling protocols [18] and to benchmarks with fixed chronological splits [69]. On Elliptic, feature saturation lets a tree ensemble match or

exceed graph models, and every method performs poorly after the dark-market closure, so a pooled score over the test period overstates late performance [16]. Pattern. The same model can rank differently depending on the negatives, the split, the baselines, and whether results are pooled. Interpretation. Part of the progress reported in the literature reflects the evaluation protocol rather than the capability of the model. Implication. Results should be reported under the controls of Table 12, which address each pitfall of Section 7.4.

Finding 4: no method family provides all four required properties. Evidence. Static and fraud-specific GNNs supply relational reasoning but not modelled time; temporal GNNs model time but remain discriminative; higher-order models represent cycles but neither time nor a distribution over structure; anomaly detectors model normality but are mostly static; and score-based diffusion learns distributions over structure but has rarely been combined with continuous-time encoding or explicit cyclic constraints (Table 14). Pattern. Each family supplies a subset of the properties derived in Section 8.2, and none supplies all four. Interpretation. The combination of temporal awareness, probabilistic generation, cycle awareness, and intrinsic explainability is the central research gap identified by this review. It is derived from the taxonomy built in this synthesis under the coding rules of Section 2, so it describes the reviewed literature rather than establishing a universal limitation. Implication. Section 9 sets out the open challenges that follow.

Table 13. Principal findings of the review, the studies that support them, and their main caveats

Finding	Supporting studies	Caveat
No single approach is best under all conditions	[16], [43], [44], [47], [114], [117] and [120]	datasets, graph construction, and metrics differ; the direction of effect, not its size, is consistent
Cyclic structure is a representational problem	[23], [24], [53] and [120]	AML evidence comes mainly from synthetic data
Evaluation practice can inflate reported performance	[16], [18] and [69]	most direct evidence is from temporal link prediction and Elliptic
No family provides all four required properties	Table 14	derived from this review's coverage and coding rules

8.2. From Literature Limitations to Four Required Properties

The four properties used in the taxonomy were not chosen in advance as categories for grouping methods. Each answers a limitation that recurs in the literature reviewed in the preceding sections, and each is defined by the coding rules of Section 2. The same four names are used throughout the review.

- Temporal awareness answers the loss of event order in static models. Static and snapshot models discard the order and timing of transactions, although laundering depends on them, since funds move through a loop within a bounded time and dormant accounts burst into activity (Section 4.3). Temporal awareness requires that time be modelled, not merely supplied as a feature.
- Probabilistic generation answers two limitations of discriminative models: they treat the observed graph as complete, although transaction data are fragmented across institutions and many edges are never recorded (Section 5.1), and their scores are often uncalibrated and cannot express uncertainty about structure (Section 7). A learned distribution over structure allows a model to reason about unobserved edges and to report calibrated probabilities (Section 5.3).
- Cycle awareness answers the expressiveness limit of message passing (Section 4.2) together with the importance of closed loops in laundering, wash trading, and round-tripping (Section 3).
- Intrinsic explainability answers the regulatory need to justify alerts, and the limitation that post-hoc explainers approximate a trained model after the fact rather than expose the evidence it used [127], [129], [134] (Section 6.3).

8.3. A Property-Centric Taxonomy

Table 14 maps eight method families against the four required properties, following the coding rules defined in Section 2. Static GNNs, represented by the message-passing formulation in Eq. (7), capture relational structures but do not explicitly model temporal dynamics or probabilistic generation. Temporal models extend this formulation through event-driven

temporal encoding (Eq. (11)), although they generally remain discriminative. Imbalance and augmentation methods address the limited representation of minority fraud patterns, while anomaly detectors identify deviations from learned normality but are typically static. Higher-order and subgraph-aware models incorporate structural information, including cycle-related patterns, through approaches such as closed-walk measures and matrix-exponential formulations (Eqs. (17) and (18)), but model neither time nor generation. Score-based diffusion models (Eqs. (22)–(25)), in contrast, provide probabilistic generation over graph structures but have rarely been combined with continuous-time encoding and explicit cyclic constraints. No reviewed method family occupies the final row of Table 14. This empty row represents the research gap identified by the synthesis, subject to two qualifications: it applies to the method families and studies covered by the review protocol, and individual systems may combine some of these properties in ways that a family-level classification does not capture.

Table 14. Taxonomy of method families against the four required properties derived in Section 8.2. ✓ denotes supported, ~ partial, and – absent

Method family	Temporal awareness	Probabilistic generation	Cycle awareness	Intrinsic explainability
Rule-based and classical ML	–	–	–	intrinsic but shallow
Static GNNs	–	–	–	post-hoc
Fraud-specific GNNs	–	–	–	post-hoc
Higher-order / subgraph GNNs	–	–	✓	post-hoc
Temporal GNNs	✓	–	–	post-hoc
Graph anomaly detection	~	~	–	~
Imbalance and augmentation	–	~	–	–
Score-based graph diffusion	~	✓	–	~
Target (open)	✓	✓	✓	intrinsic

8.4. Cyclic Fraud as a Stress Test

Viewed as a whole rather than method by method, the field turns out to be moving along a single trajectory: away from detection, understood as labelling a graph that is taken as given, and toward inference, understood as reasoning across the range of relational configurations that could have produced what was observed. What drives that trajectory is a sequence of discarded assumptions. Independence between records fell first, to relational message passing. The frozen snapshot fell next, to models that track how a graph evolves. What remains standing is the assumption that observation is complete, and it is this last one that generative inference is positioned to remove. Under that reading the families are better understood as a chain of successive relaxations than as rivals, with the equation of the observed graph with the true graph as the assumption still awaiting its turn.

Cyclic fraud is a demanding test case because it exercises all four capabilities at once, and the limitations reviewed earlier explain why. Closure is a property of an entire multi-hop walk, so the evidence for a loop is spread across several entities and transactions; recognising it requires relational reasoning over more than one hop, which standard message passing cannot supply for cycles (Section 4.2). The hops of a laundering loop also occur in order and within a bounded time, so a model without modelled time cannot distinguish a loop closed within a day from unrelated transfers spread over months (Section 4.3). A single unobserved transfer breaks the loop in the recorded graph, so detection under incomplete observation requires reasoning about edges that are missing (Section 5.1). Finally, a loop is a motif, and an alert that identifies the loop and its timing gives an investigator evidence that can be checked (Section 6.3). A method that detects loops reliably under partial observation therefore has to address the other problems as well, which is why cyclic detection serves here as a stress test rather than as a specialised case. For evaluation, the consequence is that accuracy on datasets saturated with informative attributes says little about these capabilities, whereas performance on data in which attributes are uninformative, observation is partial, and loops define the illicit class says a great deal.

8.5. Cross-Cutting Conditions on Adoptability

The four columns do not exhaust what determines whether a method is adoptable; three further conditions cut across all of them. Throughput under a deadline comes first. Screening runs in real time at high transaction rates, which disqualifies any structural reasoning that grows combinatorially and rewards inference that is amortised and free of enumeration, a contrast Table 7 makes concrete. The economics of labels comes second. Confirmations are few, arrive late, and are not always right, so anything able to learn from unlabelled structure, whether by self-supervision or by generative pre-training, holds a practical edge [115], [121]. Drift comes third. Both sides of the distribution move, adversaries by design and legitimate customers by circumstance, as Elliptic's post-shutdown collapse illustrates; detectors therefore need monitoring and scheduled retraining, and an evaluation that reports one pooled figure has concealed exactly the behaviour that matters.

9. Open Challenges and Future Directions

9.1. Open Challenges

The empty final row of Table 14 defines the open research direction, and the building blocks for it already exist in the families surveyed above. The technical challenges follow directly and are consolidated in Table 15. Six challenges follow from the analysis above.

1. Continuous-time generative inference: coupling event-driven encoders, (11), with score-based diffusion, (22), so that temporal dynamics and probabilistic generation reinforce one another rather than being bolted together.
2. Differentiable cyclic constraints: relaxing Eq. (15) into smooth, scalable proxies such as the closed-walk measure of Eq. (17) or the matrix-exponential form of Eq. (18), and using them as guidance terms, Eq. (31), rather than as post-hoc filters.
3. Robustness under missing edges Eq. (14) and adversarial camouflage, evaluated under adaptive rather than static threat models.
4. Intrinsic explainability embedded in the inference process, so that the structural evidence for an alert is a by-product of sampling rather than a separate approximation.
5. Rigorous evaluation with leakage-free temporal splits, re-trained baselines, harder negatives, and both feature-rich and structure-only benchmarks.
6. Scalability to production transaction rates under strict latency budgets, via latent diffusion over windowed sub-graphs and amortised structural measures.

Table 15. Open problems and the capability each most directly advances

Open problem	Capability advanced	Nearest existing building block
Continuous-time generative inference	temporal and probabilistic	TGN [47] with SDE [38]
Differentiable cyclic constraints	cycle-aware and scalable	closed walks [75], NO TEARS [78]
Robustness to missing edges and camouflage	deployment reliability	structure learning [73]
Intrinsic, counterfactual explanations	regulatory acceptance	guidance [105], CF-GNNExplainer [135]
Leakage-free, structure-only benchmarks	rigorous evaluation	AMLworld [119], TGB [69]
Federated and privacy-preserving learning	cross-institution coverage	federated AML [123]

9.2. Longer-Horizon Directions

Looking beyond the immediate gap, several directions appear especially fertile. Foundation models for transaction graphs: large-scale self-supervised pre-training on unlabelled transaction streams could yield transferable representations that mitigate the label scarcity endemic to AML, much as pre-training has transformed language and vision [89], [90]. Causal and counterfactual reasoning: distinguishing coincidental co-movement from orchestrated flow ultimately requires causal notions, and counterfactual explanations map naturally onto the structural-evidence outputs of generative models [135]. Multimodal fusion: real investigations combine transaction graphs with text, device telemetry, and external watchlists, and principled fusion of these modalities with the graph remains largely open. Human-in-the-loop learning: because labels arise from investigation, detectors that select the most informative cases for review can compound their own training signal. Regulatory alignment: outputs that are calibrated, typology-named, and accompanied by intrinsic evidence are far more likely to

clear deployment review than opaque scores [134]. Standardised, leakage-free benchmarks: shared, temporally split, structure-isolating benchmarks with agreed metrics would make the cross-study comparison attempted in Section 8 rigorous rather than indicative [69], [119]. Progress on these fronts will determine whether the conceptual advances surveyed here translate into deployed, defensible systems.

10. Conclusions

This review examined graph-based methods for financial fraud detection, covering static and temporal graph neural networks, anomaly detection, cycle-aware approaches, and generative models, as well as the evaluation practices used to assess their effectiveness. The comparative analysis shows that no single method consistently outperforms others across different fraud detection settings. Rather, effectiveness depends on how well the model representation captures the characteristics of the available evidence, whether in node attributes, relational structures, transaction sequences, or cyclic patterns. Attribute-rich datasets may favour conventional machine learning models, whereas graph-based approaches become more relevant when fraud signals are embedded in relationships, and temporal models are advantageous when transaction dynamics carry discriminative information. The review also identifies important limitations in current evaluation practices, including temporal leakage, inadequate negative sampling, and insufficiently retrained baselines, which can lead to overly optimistic performance claims. These findings indicate that progress in graph-based fraud detection requires not only more expressive architectures but also appropriate representations and rigorous evaluation protocols.

Despite advances across the reviewed method families, no single approach currently integrates temporal awareness, probabilistic generation over graph structures, explicit cycle awareness, and intrinsic explainability. This gap motivates several research directions, particularly the integration of continuous-time graph encoders with generative models that incorporate differentiable cyclic constraints, the development of more rigorous benchmarks that isolate structural and temporal signals under leakage-free evaluation protocols, and the incorporation of explainability directly into the inference process. Addressing these challenges could lead to fraud detection methods that better capture coordinated and evolving financial crime patterns while providing more reliable and interpretable results.

Author Contributions: Conceptualization: D.T.A.; Methodology: D.T.A.; Data curation: D.T.A.; Formal analysis: D.T.A.; Writing and original draft preparation: D.T.A.; Supervision: E.O.O., S.A.O., A.A.P., and K.B.A.; Writing, review and editing: E.O.O., S.A.O., A.A.P., and K.B.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created in this study. All works synthesised are cited in the reference list, and the public datasets discussed in Section 7 are available from their original sources as cited.

Acknowledgments: The authors thank the Department of Electrical and Electronics Engineering, Federal University of Technology, Akure, for institutional support.

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

References

[1] Association of Certified Fraud Examiners, "Occupational Fraud 2024: A Report to the Nations," 2024. [Online]. Available: https://legacy.acfe.com/report-to-the-nations/2024/

[2] U.S. Department of the Treasury, "2024 National Money Laundering Risk Assessment," 2024. [Online]. Available: https://home.treasury.gov/system/files/136/2024-National-Money-Laundering-Risk-Assessment.pdf

[3] E. Li, M. Chen, S. Xiang, and L. Chen, "Graph Learning-Empowered Financial Fraud Detection: Progress and Future Directions," *Intell. Comput.*, vol. 4, Jan. 2025, doi: 10.34133/icomputing.0146.

[4] S. Motie and B. Raahemi, "Financial fraud detection using graph neural networks: A systematic review," *Expert Syst. Appl.*, vol. 240, p. 122156, Apr. 2024, doi: 10.1016/j.eswa.2023.122156.

[5] S. Serneels, "Detecting wash trading for nonfungible tokens," *Financ. Res. Lett.*, vol. 52, p. 103374, Mar. 2023, doi: 10.1016/j.frl.2022.103374.

[6] M. Alenova, A. Utaliyeva, and K.-J. Li, "Detecting Hawala network for money laundering by graph mining," *J. Financ. Data Sci.*, vol. 10, p. 100147, Dec. 2024, doi: 10.1016/j.jfds.2024.100147.

[7] F. Victor and A. M. Weintraud, "Detecting and Quantifying Wash Trading on Decentralized Cryptocurrency Exchanges," in *Proceedings of the Web Conference 2021*, New York, NY, USA: ACM, Apr. 2021, pp. 23–32. doi: 10.1145/3442381.3449824.

[8] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," *Decis. Support Syst.*, vol. 133, p. 113303, Jun. 2020, doi: 10.1016/j.dss.2020.113303.

[9] W. Hilal, S. A. Gadsden, and J. Yawney, "Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances," *Expert Syst. Appl.*, vol. 193, p. 116429, May 2022, doi: 10.1016/j.eswa.2021.116429.

[10] E. W. T. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decis. Support Syst.*, vol. 50, no. 3, pp. 559–569, Feb. 2011, doi: 10.1016/j.dss.2010.08.006.

[11] J. West and M. Bhattacharya, "Intelligent financial fraud detection: A comprehensive review," *Comput. Secur.*, vol. 57, pp. 47–66, Mar. 2016, doi: 10.1016/j.cose.2015.09.005.

[12] A. Ali *et al.*, "Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review," *Appl. Sci.*, vol. 12, no. 19, p. 9637, Sep. 2022, doi: 10.3390/app12199637.

[13] R. J. Bolton and D. J. Hand, "Statistical Fraud Detection: A Review," *Stat. Sci.*, vol. 17, no. 3, pp. 235–255, Aug. 2002, doi: 10.1214/ss/1042727940.

[14] Y. Ma, Y. Tian, N. Moniz, and N. V Chawla, "Class-Imbalanced Learning on Graphs: A Survey," *ACM Comput. Surv.*, vol. 57, no. 8, pp. 1–16, Aug. 2025, doi: 10.1145/3718734.

[15] X. Wang, Z. Liu, J. Liu, and J. Liu, "Fraud detection on multi-relation graphs via imbalanced and interactive learning," *Inf. Sci.*, vol. 642, p. 119153, Sep. 2023, doi: 10.1016/j.ins.2023.119153.

[16] M. Weber *et al.*, "Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics," Jul. 31, 2019. [Online]. Available: http://arxiv.org/abs/1908.02591

[17] M. Zhang and Y. Chen, "Link Prediction Based on Graph Neural Networks," in *NeurIPS*, 2018, pp. 5171–5181.

[18] F. Poursafaei, S. Huang, K. Pelrine, and R. Rabbany, "Towards Better Evaluation for Dynamic Link Prediction," in *Advances in Neural Information Processing Systems 35*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022, pp. 32928–32941. doi: 10.52202/068431-2386.

[19] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling Relational Data with Graph Convolutional Networks," in *ESWC*, 2018, pp. 593–607. doi: 10.1007/978-3-319-93417-4_38.

[20] X. Wang *et al.*, "Heterogeneous Graph Attention Network," in *The World Wide Web Conference*, New York, NY, USA: ACM, May 2019, pp. 2022–2032. doi: 10.1145/3308558.3313562.

[21] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous Graph Transformer," in *Proceedings of The Web Conference 2020*, New York, NY, USA: ACM, Apr. 2020, pp. 2704–2710. doi: 10.1145/3366423.3380027.

[22] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive Representation Learning on Temporal Graphs," Feb. 19, 2020. [Online]. Available: http://arxiv.org/abs/2002.07962

[23] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?," Feb. 22, 2019. [Online]. Available: http://arxiv.org/abs/1810.00826

[24] Z. Chen, L. Chen, S. Villar, and J. Bruna, "Can Graph Neural Networks Count Substructures?," in *NeurIPS*, 2020.

[25] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[26] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA: ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[27] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, Long Beach, California, USA: Curran Associates, Inc., Dec. 2017, pp. 3149–3157.

[28] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[29] A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems 30*, Long Beach, California, USA: Curran Associates, Inc., Dec. 2017, pp. 5998–6008.

[30] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[31] T. N. Kipf and M. Welling, "Variational Graph Auto-Encoders," Nov. 21, 2016. [Online]. Available: http://arxiv.org/abs/1611.07308

[32] I. J. Goodfellow et al., "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems 27*, Montréal, Canada: Curran Associates, Inc., Dec. 2014, pp. 2672–2680.

[33] N. D. Cao and T. Kipf, "MolGAN: An implicit generative model for small molecular graphs," Sep. 27, 2018.

[34] D. J. Rezende and S. Mohamed, "Variational Inference with Normalizing Flows," Jun. 14, 2016. [Online]. Available: http://arxiv.org/abs/1505.05770

[35] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, "GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models," Jun. 23, 2018. [Online]. Available: http://arxiv.org/abs/1802.08773

[36] R. Liao *et al.*, "Efficient Graph Generation with Graph Recurrent Attention Networks," Jul. 17, 2020. [Online]. Available: http://arxiv.org/abs/1910.00760

[37] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," Dec. 16, 2020. [Online]. Available: http://arxiv.org/abs/2006.11239

[38] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling through Stochastic Differential Equations," Feb. 10, 2021. [Online]. Available: http://arxiv.org/abs/2011.13456
[39] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural Message Passing for Quantum Chemistry," *Int. Conf. Mach. Learn.*, Jun. 2017, [Online]. Available: http://arxiv.org/abs/1704.01212
[40] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," Feb. 22, 2017. [Online]. Available: http://arxiv.org/abs/1609.02907
[41] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Advances in Neural Information Processing Systems 30*, Long Beach, California, USA: Curran Associates, Inc., Dec. 2017, pp. 1024–1034.
[42] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018. [Online]. Available: http://arxiv.org/abs/1710.10903
[43] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, New York, NY, USA: ACM, Oct. 2020, pp. 315–324. doi: 10.1145/3340531.3411903.
[44] Y. Liu *et al.*, "Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection," in *Proceedings of the Web Conference 2021*, New York, NY, USA: ACM, Apr. 2021, pp. 3168–3177. doi: 10.1145/3442381.3449989.
[45] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "DYSAT: Deep Neural Representation Learning on Dynamic Graphs Via Self-Attention Networks," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, New York, NY, USA: ACM, Jan. 2020, pp. 519–527. doi: 10.1145/3336191.3371845.
[46] A. Pareja *et al.*, "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 04, pp. 5363–5370, Apr. 2020, doi: 10.1609/aaai.v34i04.5984.
[47] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, "Temporal Graph Networks for Deep Learning on Dynamic Graphs," Oct. 09, 2020. [Online]. Available: http://arxiv.org/abs/2006.10637
[48] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the Inconsistency Problem of Applying Graph Neural Network to Fraud Detection," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA: ACM, Jul. 2020, pp. 1569–1572. doi: 10.1145/3397271.3401253.
[49] G. Zhang *et al.*, "FRAUDRE: Fraud Detection Dual-Resistant to Graph Inconsistency and Imbalance," in *2021 IEEE International Conference on Data Mining (ICDM)*, IEEE, Dec. 2021, pp. 867–876. doi: 10.1109/ICDM51629.2021.00098.
[50] F. Shi, Y. Cao, Y. Shang, Y. Zhou, C. Zhou, and J. Wu, "H2-FDetector: A GNN-based Fraud Detector with Homophilic and Heterophilic Connections," in *Proceedings of the ACM Web Conference 2022*, New York, NY, USA: ACM, Apr. 2022, pp. 1486–1494. doi: 10.1145/3485447.3512195.
[51] Z. Liu, C. Chen, X. Yang, J. Zhou, X. Li, and L. Song, "Heterogeneous Graph Neural Networks for Malicious Account Detection," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, New York, NY, USA: ACM, Oct. 2018, pp. 2077–2085. doi: 10.1145/3269206.3272010.
[52] D. Wang *et al.*, "A Semi-Supervised Graph Attentive Network for Financial Fraud Detection," in *2019 IEEE International Conference on Data Mining (ICDM)*, IEEE, Nov. 2019, pp. 598–607. doi: 10.1109/ICDM.2019.00070.
[53] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein, "Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 657–668, Jan. 2023, doi: 10.1109/TPAMI.2022.3154319.
[54] C. Bodnar *et al.*, "Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks," Jun. 14, 2021. [Online]. Available: http://arxiv.org/abs/2103.03212
[55] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021, doi: 10.1109/TNNLS.2020.2978386.
[56] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020, doi: 10.1016/j.aiopen.2021.01.001.
[57] L. Sun *et al.*, "Adversarial Attack and Defense on Graph Data: A Survey," *IEEE Trans. Knowl. Data Eng.*, pp. 1–20, 2022, doi: 10.1109/TKDE.2022.3201243.
[58] Q. Li, Z. Han, and X. Wu, "Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning," *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, Apr. 2018, doi: 10.1609/aaai.v32i1.11604.
[59] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, "Representation Learning on Graphs with Jumping Knowledge Networks," Jun. 25, 2018. [Online]. Available: http://arxiv.org/abs/1806.03536
[60] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and Deep Graph Convolutional Networks," Jul. 04, 2020. [Online]. Available: http://arxiv.org/abs/2007.02133
[61] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards Deep Graph Convolutional Networks on Node Classification," Mar. 12, 2020. [Online]. Available: http://arxiv.org/abs/1907.10903
[62] J. Gasteiger, A. Bojchevski, and S. Günnemann, "Predict then Propagate: Graph Neural Networks meet Personalized PageRank," Apr. 05, 2022. [Online]. Available: http://arxiv.org/abs/1810.05997
[63] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs," Oct. 23, 2020. [Online]. Available: http://arxiv.org/abs/2006.11468
[64] S. Kumar, X. Zhang, and J. Leskovec, "Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Jul. 2019, pp. 1269–1278. doi: 10.1145/3292500.3330895.
[65] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "DyRep: Learning Representations over Dynamic Graphs," in *International Conference on Learning Representations*, 2019.
[66] Y. Wang, Y.-Y. Chang, Y. Liu, J. Leskovec, and P. Li, "Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks," Oct. 31, 2021. [Online]. Available: http://arxiv.org/abs/2101.05974

[67] W. Cong *et al.*, "Do We Really Need Complicated Model Architectures For Temporal Networks?," Feb. 22, 2023. [Online]. Available: http://arxiv.org/abs/2302.11636
[68] A. Souza, D. Mesquita, S. Kaski, and V. Garg, "Provably Expressive Temporal Graph Networks," in *Advances in Neural Information Processing Systems 35*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022, pp. 32257–32269. doi: 10.52202/068431-2337.
[69] S. Huang *et al.*, "Temporal Graph Benchmark for Machine Learning on Temporal Graphs," in *Advances in Neural Information Processing Systems 36*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023, pp. 2056–2073. doi: 10.52202/075280-0099.
[70] S. M. Kazemi *et al.*, "Representation Learning for Dynamic Graphs: A Survey," Apr. 27, 2020. [Online]. Available: http://arxiv.org/abs/1905.11485
[71] J. Skarding, B. Gabrys, and K. Musial, "Foundations and Modeling of Dynamic Networks Using Dynamic Graph Neural Networks: A Survey," *IEEE Access*, vol. 9, pp. 79143–79168, 2021, doi: 10.1109/ACCESS.2021.3082932.
[72] A. Longa *et al.*, "Graph Neural Networks for temporal graphs: State of the art, open challenges, and opportunities," Jul. 08, 2023. [Online]. Available: http://arxiv.org/abs/2302.01018
[73] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph Structure Learning for Robust Graph Neural Networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Aug. 2020, pp. 66–74. doi: 10.1145/3394486.3403049.
[74] W. Jin, T. Zhao, J. Ding, Y. Liu, J. Tang, and N. Shah, "Empowering Graph Representation Learning with Test-Time Graph Transformation," Feb. 26, 2023. [Online]. Available: http://arxiv.org/abs/2210.03561
[75] E. Estrada and J. A. Rodríguez-Velázquez, "Subgraph centrality in complex networks," *Phys. Rev. E*, vol. 71, no. 5, p. 056103, May 2005, doi: 10.1103/PhysRevE.71.056103.
[76] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network Motifs: Simple Building Blocks of Complex Networks," *Science*, vol. 298, no. 5594, pp. 824–827, Oct. 2002, doi: 10.1126/science.298.5594.824.
[77] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, Jul. 2016, doi: 10.1126/science.aad9029.
[78] X. Zheng, B. Aragam, P. Ravikumar, and E. P. Xing, "DAGs with NO TEARS: Continuous Optimization for Structure Learning," in *NeurIPS*, 2018, pp. 9492–9503.
[79] Y. Yu, J. Chen, T. Gao, and M. Yu, "DAG-GNN: DAG Structure Learning with Graph Neural Networks," Apr. 22, 2019. [Online]. Available: http://arxiv.org/abs/1904.10098
[80] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Min. Knowl. Discov.*, vol. 29, no. 3, pp. 626–688, May 2015, doi: 10.1007/s10618-014-0365-y.
[81] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009, doi: 10.1145/1541880.1541882.
[82] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep Anomaly Detection on Attributed Networks," in *Proceedings of the 2019 SIAM International Conference on Data Mining*, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2019, pp. 594–602. doi: 10.1137/1.9781611975673.67.
[83] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep Graph Infomax," Dec. 21, 2018. [Online]. Available: http://arxiv.org/abs/1809.10341
[84] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly Detection on Attributed Networks via Contrastive Self-Supervised Learning," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 6, pp. 2378–2392, Jun. 2022, doi: 10.1109/TNNLS.2021.3068344.
[85] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y.-P. P. Chen, "Generative and Contrastive Self-Supervised Learning for Graph Anomaly Detection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12220–12233, Dec. 2023, doi: 10.1109/TKDE.2021.3119326.
[86] J. Duan *et al.*, "Graph Anomaly Detection via Multi-Scale Contrastive Learning Networks with Augmented View," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 6, pp. 7459–7467, Jun. 2023, doi: 10.1609/aaai.v37i6.25907.
[87] Y. Liu *et al.*, "Graph Self-Supervised Learning: A Survey," *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2022, doi: 10.1109/TKDE.2022.3172903.
[88] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph Contrastive Learning with Augmentations," in *Advances in Neural Information Processing Systems 33*, Curran Associates, Inc., Dec. 2020, pp. 5812–5823.
[89] J. Qiu *et al.*, "GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Aug. 2020, pp. 1150–1160. doi: 10.1145/3394486.3403168.
[90] W. Hu *et al.*, "Strategies for Pre-training Graph Neural Networks," Feb. 18, 2020. [Online]. Available: http://arxiv.org/abs/1905.12265
[91] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking Graph Neural Networks for Anomaly Detection," May 31, 2022. [Online]. Available: http://arxiv.org/abs/2205.15508
[92] Y. Gao, X. Wang, X. He, Z. Liu, H. Feng, and Y. Zhang, "Addressing Heterophily in Graph Anomaly Detection: A Perspective of Graph Spectrum," in *Proceedings of the ACM Web Conference 2023*, New York, NY, USA: ACM, Apr. 2023, pp. 1528–1538. doi: 10.1145/3543507.3583268.
[93] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
[94] T. Zhao, X. Zhang, and S. Wang, "GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, New York, NY, USA: ACM, Mar. 2021, pp. 833–841. doi: 10.1145/3437963.3441720.

[95] J. Park, J. Song, and E. Yang, "GraphENS: Neighbor-Aware Ego Network Synthesis for Class-Imbalanced Node Classification," in *ICLR*, 2022.
[96] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy: IEEE, Oct. 2017, pp. 2999–3007. doi: 10.1109/ICCV.2017.324.
[97] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep Unsupervised Learning using Nonequilibrium Thermodynamics," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, Lille, France: PMLR, Jul. 2015, pp. 2256–2265.
[98] B. D. O. Anderson, "Reverse-time diffusion equation models," *Stoch. Process. their Appl.*, vol. 12, no. 3, pp. 313–326, May 1982, doi: 10.1016/0304-4149(82)90051-5.
[99] A. Hyvärinen, "Estimation of Non-Normalized Statistical Models by Score Matching," 2005.
[100] P. Vincent, "A Connection Between Score Matching and Denoising Autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, Jul. 2011, doi: 10.1162/NECO_a_00142.
[101] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the Design Space of Diffusion-Based Generative Models," in *Advances in Neural Information Processing Systems 35*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022, pp. 26565–26577. doi: 10.52202/068431-1926.
[102] Y. Song and S. Ermon, "Generative Modeling by Estimating Gradients of the Data Distribution," Oct. 10, 2020. [Online]. Available: http://arxiv.org/abs/1907.05600
[103] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," Oct. 05, 2022. [Online]. Available: http://arxiv.org/abs/2010.02502
[104] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps," in *Advances in Neural Information Processing Systems 35*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022, pp. 5775–5787. doi: 10.52202/068431-0418.
[105] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis," Jun. 01, 2021. [Online]. Available: http://arxiv.org/abs/2105.05233
[106] J. Ho and T. Salimans, "Classifier-Free Diffusion Guidance," Jul. 26, 2022. [Online]. Available: http://arxiv.org/abs/2207.12598
[107] C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon, "Permutation Invariant Graph Generation via Score-Based Generative Modeling," in *AISTATS*, 2020, pp. 4474–4484.
[108] J. Jo, S. Lee, and S. J. Hwang, "Score-based Generative Modeling of Graphs via the System of Stochastic Differential Equations," in *ICML*, 2022, pp. 10362–10383.
[109] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "DiGress: Discrete Denoising diffusion for graph generation," May 23, 2023. [Online]. Available: http://arxiv.org/abs/2209.14734
[110] K. K. Haefeli, K. Martinkus, N. Perraudin, and R. Wattenhofer, "Diffusion Models for Graphs Benefit From Discrete State Spaces," Aug. 15, 2023. [Online]. Available: http://arxiv.org/abs/2210.01549
[111] L. Kong, J. Cui, H. Sun, Y. Zhuang, B. A. Prakash, and C. Zhang, "Autoregressive Diffusion Model for Graph Generation," Jul. 17, 2023. [Online]. Available: http://arxiv.org/abs/2307.08849
[112] C. Liu *et al.*, "Generative Diffusion Models on Graphs: Methods and Applications," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, California: International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 6702–6711. doi: 10.24963/ijcai.2023/751.
[113] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2022, pp. 10674–10685. doi: 10.1109/CVPR52688.2022.01042.
[114] I. Alarab, S. Prakoonwit, and M. I. Nacer, "Comparative Analysis Using Supervised Learning Methods for Anti-Money Laundering in Bitcoin," in *Proceedings of the 2020 5th International Conference on Machine Learning Technologies*, New York, NY, USA: ACM, Jun. 2020, pp. 11–17. doi: 10.1145/3409073.3409078.
[115] J. Lorenz, M. I. Silva, D. Aparício, J. T. Ascensão, and P. Bizarro, "Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of label scarcity," in *Proceedings of the First ACM International Conference on AI in Finance*, New York, NY, USA: ACM, Oct. 2020, pp. 1–8. doi: 10.1145/3383455.3422549.
[116] C. Oliveira, J. Torres, M. I. Silva, D. Aparício, J. T. Ascensão, and P. Bizarro, "GuiltyWalker: Distance to illicit nodes in the Bitcoin network," Jul. 21, 2021. [Online]. Available: http://arxiv.org/abs/2102.05373
[117] W. W. Lo, G. K. Kulatilleke, M. Sarhan, S. Layeghy, and M. Portmann, "Inspection-L: self-supervised GNN node embeddings for money laundering detection in bitcoin," *Appl. Intell.*, vol. 53, no. 16, pp. 19406–19417, Aug. 2023, doi: 10.1007/s10489-023-04504-9.
[118] T. Suzumura and H. Kanezashi, "Anti-Money Laundering Datasets: AMLSim synthetic financial transaction generator," 2021. [Online]. Available: https://github.com/IBM/AMLSim
[119] E. Altman, J. Blanuša, L. von Niederhäusern, B. Egressy, A. Anghel, and K. Atasu, "Realistic Synthetic Financial Transactions for Anti-Money Laundering Models," in *Advances in Neural Information Processing Systems 36*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023, pp. 29851–29874. doi: 10.52202/075280-1300.
[120] B. Egressy, L. von Niederhäusern, J. Blanuša, E. Altman, R. Wattenhofer, and K. Atasu, "Provably Powerful Graph Neural Networks for Directed Multigraphs," *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 10, pp. 11838–11846, Mar. 2024, doi: 10.1609/aaai.v38i10.29069.
[121] M. Cardoso, P. Saleiro, and P. Bizarro, "LaundroGraph: Self-Supervised Graph Representation Learning for Anti-Money Laundering," in *Proceedings of the Third ACM International Conference on AI in Finance*, New York, NY, USA: ACM, Nov. 2022, pp. 130–138. doi: 10.1145/3533271.3561727.
[122] F. Johannessen and M. Jullum, "Finding money launderers using heterogeneous graph neural networks," *J. Financ. Data Sci.*, vol. 11, p. 100175, Dec. 2025, doi: 10.1016/j.jfds.2025.100175.

[123]T. Suzumura, Y. Zhou, R. Kawahara, N. Baracaldo, and H. Ludwig, "Federated Learning for Collaborative Financial Crimes Detection," in *Federated Learning*, Cham: Springer International Publishing, 2022, pp. 455–466. doi: 10.1007/978-3-030-96896-0_20.
[124]D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial Attacks on Neural Networks for Graph Data," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, California: International Joint Conferences on Artificial Intelligence Organization, Aug. 2019, pp. 6246–6250. doi: 10.24963/ijcai.2019/872.
[125]D. Zügner and S. Günnemann, "Adversarial Attacks on Graph Neural Networks via Meta Learning," Jan. 28, 2024. [Online]. Available: http://arxiv.org/abs/1902.08412
[126]H. Dai *et al.*, "Adversarial Attack on Graph Structured Data," Jun. 06, 2018. [Online]. Available: http://arxiv.org/abs/1806.02371
[127]R. Ying, D. Bourgeois, J. You, M. Žitnik, and J. Leskovec, "GNNExplainer: Generating Explanations for Graph Neural Networks," *PubMed*, vol. 32, pp. 9240–9251, 2019.
[128]D. Luo *et al.*, "Parameterized Explainer for Graph Neural Network," Nov. 09, 2020. [Online]. Available: http://arxiv.org/abs/2011.04573
[129]H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji, "On Explainability of Graph Neural Networks via Subgraph Explorations," May 31, 2021. [Online]. Available: http://arxiv.org/abs/2102.05152
[130]M. N. Vu and M. T. Thai, "PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks," Oct. 12, 2020. [Online]. Available: http://arxiv.org/abs/2010.05788
[131]P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, "Explainability Methods for Graph Convolutional Neural Networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2019, pp. 10764–10773. doi: 10.1109/CVPR.2019.01103.
[132]H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in Graph Neural Networks: A Taxonomic Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–19, 2022, doi: 10.1109/TPAMI.2022.3204236.
[133]S. X. Rao *et al.*, "xFraud," *Proc. VLDB Endow.*, vol. 15, no. 3, pp. 427–436, Nov. 2021, doi: 10.14778/3494124.3494128.
[134]C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nat. Mach. Intell.*, vol. 1, no. 5, pp. 206–215, May 2019, doi: 10.1038/s42256-019-0048-x.
[135]A. Lučić, M. A. ter Hoeve, G. Tolomei, M. de Rijke, and F. Silvestri, "CF-GNNExplainer: Counterfactual Explanations for Graph Neural Networks," in *Proc. Int. Conf. Artif. Intell. Stat. (AISTATS)*, 2022, pp. 4499–4511.
[136]W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-GCN," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Jul. 2019, pp. 257–266. doi: 10.1145/3292500.3330925.
[137]H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "GraphSAINT: Graph Sampling Based Inductive Learning Method," Feb. 16, 2020. [Online]. Available: http://arxiv.org/abs/1907.04931
[138]H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," Jan. 26, 2023. [Online]. Available: http://arxiv.org/abs/1602.05629
[139]P. Kairouz and H. B. McMahan, "Advances and Open Problems in Federated Learning," *Found. Trends® Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, Jun. 2021, doi: 10.1561/2200000083.
[140]C. Dwork and A. Roth, *The Algorithmic Foundations of Differential Privacy*. now Publishers Inc, 2013. doi: 10.1561/9781601988195.
[141]W. Hu *et al.*, "Open Graph Benchmark: Datasets for Machine Learning on Graphs," Feb. 25, 2021. [Online]. Available: http://arxiv.org/abs/2005.00687
[142]K. Liu *et al.*, "BOND: Benchmarking Unsupervised Outlier Node Detection on Static Attributed Graphs," in *Advances in Neural Information Processing Systems 35*, San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022, pp. 27021–27035. doi: 10.52202/068431-1959.
[143]Y. Elmougy and L. 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*, New York, NY, USA: ACM, Aug. 2023, pp. 3979–3990. doi: 10.1145/3580305.3599803.
[144]C. Bellei *et al.*, "The Shape of Money Laundering: Subgraph Representation Learning on the Blockchain with the Elliptic2 Dataset," Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/2404.19109
[145]J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, New York, New York, USA: ACM Press, 2006, pp. 233–240. doi: 10.1145/1143844.1143874.
[146]T. Saito and M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets," *PLoS One*, vol. 10, no. 3, p. e0118432, Mar. 2015, doi: 10.1371/journal.pone.0118432.
[147]C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," Aug. 03, 2017, *PMLR*. doi: 10.48550/arXiv.1706.04599.