# FraudGT: A Simple, Effective, and Efficient Graph Transformer for Financial Fraud Detection

Junhong Lin
Massachusetts Institute of Technology
US
junhong@mit.edu

Xiaojie Guo
IBM T.J. Watson Research Center
US
xiaojie.guo@ibm.com

Yada Zhu
IBM T.J. Watson Research Center
US
yzhu@us.ibm.com

Samuel Mitchell
Massachusetts Institute of Technology
US
sammit@mit.edu

Erik Altman
IBM T.J. Watson Research Center
US
ealtman@us.ibm.com

Julian Shun
Massachusetts Institute of Technology
US
jshun@mit.edu

## Abstract

Fraud detection plays a crucial role in the financial industry, preventing significant financial losses. Traditional rule-based systems and manual audits often struggle with the evolving nature of fraud schemes and the vast volume of transactions. Recent advances in machine learning, particularly graph neural networks (GNNs), have shown promise in addressing these challenges. However, GNNs still face limitations in learning intricate patterns, effectively utilizing edge attributes, and maintaining efficiency on large financial graphs. To address these limitations, we introduce FraudGT, a simple, effective, and efficient graph transformer (GT) model specifically designed for fraud detection in financial transaction graphs. FraudGT leverages edge-based message passing gates and an edge attribute-based attention bias to enhance its ability to discern important transactional features and differentiate between normal and fraudulent transactions. Our model achieves state-of-the-art performance in detecting fraudulent activities while demonstrating high throughput and significantly lower latency compared to existing methods. We validate the effectiveness of FraudGT through extensive experiments on multiple large-scale synthetic financial datasets. FraudGT consistently outperforms other models, achieving 7.8–17.8% higher F1 scores, while delivering an average of 2.4× greater throughput and reduced latency. Our code and datasets are available at https://github.com/junhongmit/FraudGT.

## CCS Concepts

• **Mathematics of computing → Graph algorithms**; • **Computing methodologies → Neural networks**; • **Information systems → Data mining**; • **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Applied computing → Business process monitoring**.

*ICAIF '24, November 14–17, 2024, Brooklyn, NY, USA*
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1081-0/24/11
https://doi.org/10.1145/3677052.3698648

## Keywords

Financial transaction networks, fraud detection, graph transformers, graph neural networks, graph learning

**ACM Reference Format:**
Junhong Lin, Xiaojie Guo, Yada Zhu, Samuel Mitchell, Erik Altman, and Julian Shun. 2024. FraudGT: A Simple, Effective, and Efficient Graph Transformer for Financial Fraud Detection. In *5th ACM International Conference on AI in Finance (ICAIF '24), November 14–17, 2024, Brooklyn, NY, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3677052.3698648

## 1 Introduction

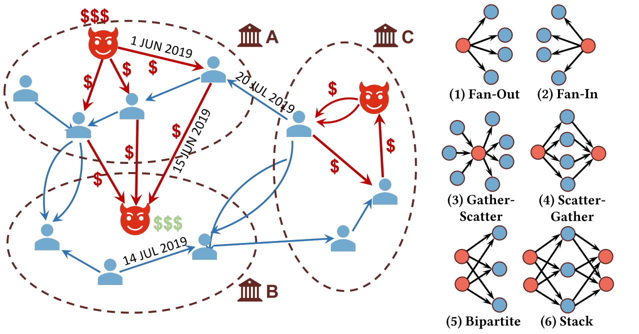
Fraud detection is a critical task in the financial industry, encompassing various applications such as anti-money laundering [9, 32, 58], malicious account/commodity detection (e.g., in online payment systems [10, 68] and e-commerce systems [7, 8, 42]), and spam detection [50, 57]. Financial fraud can lead to significant financial losses, reputation damage, and regulatory penalties for financial institutions. Traditional fraud detection methods, which rely heavily on rule-based systems and manual audits [28], are often inadequate due to the evolving nature of fraud schemes and the sheer volume of transactions that need to be monitored. Rule-based systems also suffer from low accuracy.

Recent advances in machine learning, particularly in graph neural networks (GNNs), have shown promise in enabling effective and efficient fraud detection. GNNs are well-suited for learning from graph-structured data and have been successfully applied in various important domains such as biology [31], chemistry [67], and recommendation systems [59]. Their ability to capture complex structures in data makes them an ideal choice for financial fraud detection, where transactional relationships, as shown in Figure 1(a), can be represented as graphs, as illustrated in Figure 1(b). Here each node represents a financial account, and each edge represents a financial transaction between two accounts. Financial transaction graphs are often directed multigraphs, where edges (or transactions) have a direction, and there can be multiple edges between two nodes (or accounts). By using message passing to propagate information between connected nodes [20], GNNs can learn representations that capture the graph structure and compute the fraud likelihood for each transaction or account based on these learned representations.

However, traditional GNNs face the following challenges in effectively handling the unique characteristics of financial graphs.

| Trans ID. | Timestamp | Src Bank | Src Account | Dst Bank | Dst Account | Amount | Currency | Payment Type |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 JUN 2019 10:45 | A | 1 | A | 2 | 1200 | EUR | Cheque |
| 1 | 15 JUN 2019 13:50 | A | 2 | B | 3 | 710 | USD | Credit Card |
| 2 | 14 JUL 2019 08:36 | B | 4 | B | 5 | 1100 | CHF | ACH |
| 4 | 20 JUL 2019 14:06 | C | 6 | A | 2 | 2500 | USD | Wire |
| 5 | … | … |  | … |  | … | … | … |

**(a) Transactions in Tabular Format**



**(b) Transactions in Graph Format** **(c) Additional Fraud Patterns**

**Figure 1: Financial transactions in tabular format (a) can be modeled as graphs (b), where nodes represent accounts and edges represent transactions between these accounts. Multiple transactions can exist between accounts, making it a directed multigraph. Two example money laundering patterns, smurfing (scatter-gather) and round tripping (simple cycle), are depicted in (b), where red nodes represent fraudsters and blue nodes represent innocent users. (c) demonstrates additional possible patterns for financial fraud.**

(1) **Learning Intricate Fraudulent Patterns.** Financial graph learning involves complicated money laundering patterns such as smurfing (scatter-gathering), round-tripping (cycles), and bipartite subgraphs, which are often found in realistic data [3, 23, 27, 52]. As a concrete example, Figure 1(b) and Figure 1(c) demonstrate 7 possible money laundering patterns encountered in practice. However, these patterns are rather generic and also appear extensively in innocent transactions. As a result, detecting financial crime relies not just on detecting individual patterns but also on learning relevant combinations of patterns. Conventional GNNs suffer from issues such as limited expressivity [19, 61], over-smoothing [40, 51], and over-squashing [1], which can limit their effectiveness in capturing such complex crime patterns. In particular, it has been shown that most GNNs fail to detect certain subgraph patterns, such as cycles [13, 14].

(2) **Effective Use of Transaction Information Along Graph Edges.** In financial transaction graphs, the majority of the information is located in the edge features (transaction information), as presented in Figure 1(a). However, many common GNNs, including GCN [38], GAT [56], and GraphSAGE [24], do not effectively use edge features, especially multi-dimensional edge features, limiting their performance on financial fraud detection.

(3) **Efficiency on Large Financial Graphs.** Rapid detection and processing of suspicious financial transactions are crucial in avoiding financial losses, especially on platforms that generate millions of transactions in a short time [48]. Recent studies [19] have proposed GNN designs that achieve state-of-the-art detection accuracy but require sampling a large neighborhood, leading to a high computational complexity, which in turn leads to low throughput and high latency [5]. Developing a GNN method that is both effective and computationally efficient for financial fraud detection remains a challenge.

To address these challenges, we introduce FraudGT, a simple, effective, and efficient graph-transformer-based model specifically designed for fraud detection in financial transaction graphs. Graph transformers (GTs) have been shown to be at least as expressive as second-order invariant graph networks [37, 44], which are more expressive than message-passing GNNs [20]. This increased expressivity makes GTs well-suited for learning complex patterns, such as those found in sophisticated fraud schemes. To effectively utilize edge information, FraudGT introduces an innovative edge-based message passing gate, which selectively passes effective messages to neighboring nodes, allowing them to discern important transactional features indicative of fraudulent activities. By filtering out less relevant information, the model can focus on the most critical features. FraudGT incorporates edge attributes-based attention bias to better differentiate between normal and fraudulent transactions, providing a more nuanced understanding of transactional relationships. We also incorporate three enhancements—reverse message passing, port numbering, and ego ID—that improve graph learning on directed multigraphs. We demonstrate on a collection of large-scale synthetic financial transaction datasets that the resulting FraudGT variants, such as PE-FraudGT (equipped with port numbering and ego ID) and Multi-FraudGT (equipped with all three enhancements) can achieve improved fraud detection performance over the state-of-the-art gradient-boosting and GNN methods by 7.8–17.8% in F1 score. Lastly, we observe that FraudGT is data-efficient and delivers greater throughput and lower latency compared to existing methods—it achieves remarkable detection accuracy while requiring a much smaller sampled neighborhood. In contrast, existing GNN methods need to sample neighborhoods that are 4.5× larger on average to achieve their best F1 score, resulting in lower throughput and higher latency while still not surpassing FraudGT in terms of F1 score.

In summary, the contributions of this paper are threefold:

- **Model.** We introduce FraudGT, a simple, effective, and efficient graph-transformer-based model tailored for financial fraud detection. We demonstrate how edge attributes can be effectively leveraged to enhance fraud detection performance.
- **Evaluation.** We conduct extensive evaluations on various publicly available large-scale synthetic financial transaction datasets demonstrating that FraudGT achieves state-of-the-art performance, significantly outperforming existing baselines.
- **Reproducibility.** We open-source our code at https://github.com/junhongmit/FraudGT, allowing our experimental results to be reproduced.

## 2 Related Work

*Fraud Detection.* Research in fraud detection follows two main lines of work. The first line of work leverages graph information using non-GNN methods, relying on manual inspection and predefined rules to identify fraudulent transactions. These methods include event-based models that analyze individual transactions for anomalies [12, 34] and sequence-based models which detect fraud by identifying suspicious patterns over time [29, 41]. The second line of work utilizes GNNs to capture complex relationships in financial transaction graphs. GNN models have shown to be effective in many fraudulent scenarios, including anti-money laundering [9, 32, 58], credit card fraud [18, 60, 62], malicious account detection [17, 42], and fake review manipulation [57].

*Graph Neural Networks (GNNs).* GNNs have become a cornerstone in graph-based learning tasks due to their ability to model complex relationships within graph-structured data. Traditional GNNs rely on a message-passing mechanism [20], which collects information from neighbors and combines it with the node's own features to update the representation of a node . GNNs are divided into two categories: spectral GNNs and spatial GNNs.

Spectral GNNs apply graph convolution operations in the spectral domain. ChebNet [16] approximates graph convolution using polynomial expansion. GCN [38] performs spectral convolutions on graphs to capture structure and feature information.

Spatial GNNs apply the convolution operation on the graph structure by leveraging the information of neighborhood nodes. GraphSAGE [24] proposes a general inductive framework that can efficiently update the representation of sampled nodes. GAT [56] leverages a self-attention mechanism to enable distinct treatment of various neighbors during the embedding update of a node.

Despite their success, common GNNs exhibit several limitations. GNNs with limited expressivity [1, 40, 51] may struggle to learn intricate patterns within a graph, which are essential for tasks like financial fraud detection. Common GNNs also inadequately utilize or neglect rich information contained in edge attributes [21], primarily focusing on node features. This limitation hinders their effectiveness on tasks where edge information is crucial, such as financial transaction graphs.

*Graph Transformers (GTs).* GTs [39, 45, 46, 46, 63, 63, 66] extend the transformative capabilities of conventional transformer architectures, which have made significant strides in both natural language processing [35, 55, 66] and computer vision [25, 47, 65]. By using powerful attention mechanisms, transformers overcome limitations in traditional message-passing GNNs [24, 38], such as over-smoothing and over-squashing [51]. However, many of these models overlook the critical role of edge features in financial transaction graphs. Our work aims to bridge this gap by introducing edge-related components to enhance a GT's ability to discern important transactional features indicative of fraudulent activities, addressing the shortcomings of existing GTs in the context of financial fraud detection.

## 3 Proposed Method

In this section, we first present preliminaries on graph representation of financial transaction networks and graph transformers. Then, we introduce the architecture and methodology of FraudGT.

### 3.1 Preliminaries

*Graph Representation.* A financial transaction network can be represented as a directed multigraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X}, \mathbf{E})$, where $\mathcal{V}$ is a set of $n$ nodes representing accounts and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ corresponds to a set of edges representing transactions between these accounts and there can be multiple edges between two nodes. If the graph is node-attributed or edge-attributed, the node attribute matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d_n}$ assigns attributes to each node, and the edge attribute tensor $\mathbf{E} \in \mathbb{R}^{n \times n \times d_e}$ assigns attributes to each edge. $d_n$ and $d_e$ are the dimensions of node and edge attributes, respectively. We use $\mathbf{E}_{ij} \in \mathbb{R}^{d_e}$ to denote the attribute of the edge that connects node $v_i$ and $v_j$, and $\mathbf{E}_{ij} = \mathbf{0}$ means there is no edge between $v_i$ and $v_j$.

*Graph Transformers.* The goal of graph transformers is to learn a node representation that captures graph structure, based on feature-based proximities between different positions in the input node feature matrix. The learned representation is then used in downstream tasks, such as computing the fraud likelihood of an account or transaction. A graph transformer is a stack of $L$ layers with blocks of multi-head attention (MHA) modules and fully connected feed-forward networks (FFN) in each layer. Let $\mathcal{G}$ be a graph with node feature matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times d_n}$, where $\boldsymbol{x}_i \in \mathbb{R}^{d_n}$ is the node feature of node $v_i$. In each layer $l$ ($l > 0$), given the hidden feature matrix $\boldsymbol{H}^{(l-1)} \in \mathbb{R}^{n \times d_n}$, where $\boldsymbol{H}^{(0)} = \boldsymbol{X}$, the MHA module first linearly projects the input $\boldsymbol{H}^{(l-1)}$ to the query, key, and value spaces. This is computed using the following equations, where the projection using weight matrices $\boldsymbol{W}_Q^{(h,l)}, \boldsymbol{W}_K^{(h,l)}$, and $\boldsymbol{W}_V^{(h,l)} \in \mathbb{R}^{d_n \times d_h}$ results in the matrices $\boldsymbol{Q}^{(h,l)}$, $\boldsymbol{K}^{(h,l)}$, and $\boldsymbol{V}^{(h,l)}$, representing the query, key, and value spaces, respectively:

$$\boldsymbol{Q}^{(h,l)} = \boldsymbol{H}^{(l-1)} \boldsymbol{W}_Q^{(h,l)}, \ \boldsymbol{K}^{(h,l)} = \boldsymbol{H}^{(l-1)} \boldsymbol{W}_K^{(h,l)}, \ \boldsymbol{V}^{(h,l)} = \boldsymbol{H}^{(l-1)} \boldsymbol{W}_V^{(h,l)}. \tag{1}$$

Then, multiple attention heads are used to compute the scaled dot-product, as shown in Equation (2), where the softmax function is applied row-wise, $\boldsymbol{W}_{O_n}^{(l)} \in \mathbb{R}^{d_n \times d_n}$ is a learnable weight matrix, $d_h$ denotes the feature dimension of the matrices $\boldsymbol{Q}^{h,l}$ and $\boldsymbol{K}^{h,l}$, $h = 1$ to $H$ denotes the index of different attention heads, and $\|$ denotes the concatenation operator.

$$\mathrm{MHA}\left(\boldsymbol{H}^{(l-1)}\right) = \big\|_{h \in [1,H]} \left(\mathrm{softmax}\left(\frac{\boldsymbol{Q}^{(h,l)} (\boldsymbol{K}^{(h,l)})^T}{\sqrt{d_h}}\right) \boldsymbol{V}^{(h,l)}\right) \boldsymbol{W}_{O_n}^{(l)}. \tag{2}$$

The multi-head attention module $\mathrm{MHA}(\boldsymbol{H}^{(l-1)})$ concatenates several attention heads together. By combining the result with additional residual connections and normalization, the transformer layer updates features $\boldsymbol{H}^{(l-1)}$ as follows:

$$\hat{\boldsymbol{H}}^{(l)} = \mathrm{MHA}\left(\boldsymbol{H}^{(l-1)}\right) + \boldsymbol{H}^{(l-1)} \tag{3}$$

$$\boldsymbol{H}^{(l)} = \mathrm{FFN}\left(\hat{\boldsymbol{H}}^{(l)}\right) + \boldsymbol{H}^{(l)} = \left[\sigma\left(\hat{\boldsymbol{H}}^{(l)} \boldsymbol{W}_1^{(l)}\right) \boldsymbol{W}_2^{(l)}\right] + \boldsymbol{H}^{(l)}, \tag{4}$$

where $\sigma$ refers to the activation function, and $\boldsymbol{W}_1^{(l)} \in \mathbb{R}^{d_n \times d_f}$ and $\boldsymbol{W}_2^{(l)} \in \mathbb{R}^{d_f \times d_n}$ are trainable parameters in the feedforward network (FFN) layer. The final output $\boldsymbol{H}^{(L)} \in \mathbb{R}^{n \times d_n}$ can be used as the updated node representation for downstream tasks.

### 3.2 FraudGT

To handle financial transaction graphs, we introduce a novel graph transformer model, FraudGT, which consists of new components compared to existing graph transformers, including an edge-based

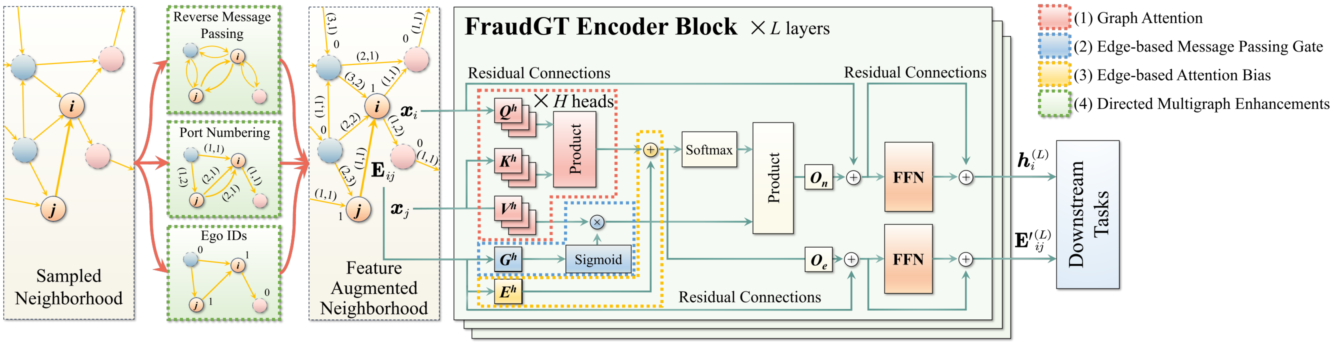


**Figure 2: Overall FraudGT architecture. The four components are indicated by dashed outlines in different colors. $x_i$ and $\mathbf{E}_{ij}$ denotes the node attribute of node $v_i$ and edge attribute of the edge between node $v_i$ and $v_j$, respectively. FFN stands for feed-forward network.**

message passing gate and an edge attributes-based attention bias to effectively utilize the transaction information along graph edges, and three enhancements to handle the multigraph nature of financial transaction graphs.

FraudGT, illustrated in Figure 2, consists of four components: (1) graph attention, (2) edge-based message passing gate, (3) edge-based attention bias, and (4) directed multigraph enhancements. Given a financial transaction graph, the workflow of FraudGT begins with a sampling step, which selectively extracts a subgraph centered around the target transactions, capturing their local neighborhood. Subsequently, the node feature matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times d_n}$ and edge feature tensor $\mathbf{E} \in \mathbb{R}^{n \times n \times d_e}$, where $\boldsymbol{x}_i \in \mathbb{R}^{d_n}$, $\mathbf{E}_{ij} \in \mathbb{R}^{d_e}$ denote the feature of node $v_i$ and edge between $v_i$ and $v_j$ in the sampled neighborhood, pass through the FraudGT encoder blocks. The updated node representations $\boldsymbol{H} \in \mathbb{R}^{n \times d_n}$ and edge representations $\mathbf{E}' \in \mathbb{R}^{n \times n \times d_e}$, which capture graph structure information, are then generated and used in the downstream task.

Within the $L$ layers FraudGT encoder blocks, as formulated in Equations (5) and (6), $\boldsymbol{X}$ is processed by the (1) *graph attention* to calculate scalar importance weight $\boldsymbol{\alpha}_{ij}^{(h,l)}$ (Section 3.2.1) and perform weighted message passing between connected nodes $v_i$ and $v_j$, employing the graph structure to refine the node features. $\mathbf{E}$ is leveraged in (2) the *edge-based message passing gate* to generate a gating vector $\mathbf{G}_{ij}^{(h,l)}$ (Section 3.2.2) between $v_i$ and $v_j$, which allows messages in selective channels passing to neighboring nodes. To help differentiate between normal and suspicious transactions, (3) an *edge-based attention bias* is added to the graph attention (Section 3.2.3). As financial transaction graphs are commonly directed multigraphs, we apply (4) *directed multigraph enhancements* to augment the sampled neighborhood and greatly improve the effectiveness of FraudGT (Section 3.2.4).

$$\mathrm{MHA}\left(\boldsymbol{h}_i^{(l-1)}\right) = \underset{h\in[1,H]}{\Big\|} \left( \underset{\forall j \in \mathcal{N}(i)}{\oplus} \left( \boldsymbol{\alpha}_{ij}^{(h,l)} (\boldsymbol{h}_j^{(l-1)})^T \boldsymbol{W}_V^{(h,l)} \odot \mathbf{G}_{ij}^{(h,l)} \right) \right)^T \boldsymbol{W}_{O_n}^{(l)} \tag{5}$$

$$\mathrm{MHA}\left(\mathbf{E}'^{(l-1)}_{ij}\right) = \underset{h\in[1,H]}{\Big\|} \left( \hat{\boldsymbol{\alpha}}_{ij}^{(h,l)} \right)^T \boldsymbol{W}_{O_e}^{(l)}, \tag{6}$$

where $\boldsymbol{h}_i^{(l-1)}$ denotes the feature of node $v_i$ with $\boldsymbol{h}_i^{(0)} = \boldsymbol{x}_i$, $\hat{\boldsymbol{\alpha}}_{ij}^{(h,l)} \in \mathbb{R}^{d_h}$ is defined in Equation (8), and $\boldsymbol{W}_{O_e}^{(l)} \in \mathbb{R}^{d_e \times d_e}$ is a learnable weight matrix. $\oplus$ denotes element-wise addition, $\odot$ denotes the Hadamard product, and $\|$ denotes the concatenation operator.

### 3.2.1 *Graph Attention.*

The core of FraudGT lies in its attention mechanism, which allows the model to focus on the most relevant parts of the graph. Given a node $v_i$, FraudGT computes attention scores over its direct neighbor nodes $v_j \in \mathcal{N}(v_i)$ using both node features and edge attributes. This is achieved through a direct neighbor attention mask, ensuring that the attention is restricted to immediate neighbors, thereby capturing local transactional patterns effectively. The attention score $\boldsymbol{\alpha}_{ij}^{(h,l)}$ of attention head $h$ in layer $l$ between nodes $v_i$ and $v_j$ is computed as follows:

$$\boldsymbol{q}_i^{(h,l)} = \left(\boldsymbol{h}_i^{(l-1)}\right)^T \boldsymbol{W}_Q^{(h,l)}, \qquad \boldsymbol{k}_j^{(h,l)} = \left(\boldsymbol{h}_j^{(l-1)}\right)^T \boldsymbol{W}_K^{(h,l)} \tag{7}$$

$$\hat{\boldsymbol{\alpha}}_{ij}^{(h,l)} = \frac{\boldsymbol{q}_i^{(h,l)} \odot \boldsymbol{k}_j^{(h,l)} + \boldsymbol{b}_{ij}^{(h,l)}}{\sqrt{d_k}}, \qquad \boldsymbol{\alpha}_{ij}^{(h,l)} = \mathrm{softmax}\left(\hat{\boldsymbol{\alpha}}_{ij}^{(h,l)}\right), \tag{8}$$

where $\boldsymbol{h}_i^{(l-1)}$ denotes the feature of nodes $v_i$, and $\boldsymbol{b}_{ij}^{(h,l)}$ is the attention bias that will be introduced in Section 3.2.3.

### 3.2.2 *Edge-Based Message Passing Gate.*

Inspired by the gating operation in Bresson and Laurent [6], FraudGT proposes an edge-based message passing gate to discern important features of financial transactions, which is calculated as $\mathbf{G}_{ij}^{(h,l)} = \sigma\left((\mathbf{E}'^{(l)}_{ij})^T \boldsymbol{W}_G^{(h,l)}\right) \in \mathbb{R}^{d_e}$, with $\sigma$ being the sigmoid function and $\boldsymbol{W}_G^{(h,l)} \in \mathbb{R}^{d_e \times d_h}$ being a learnable weight matrix. This mechanism ensures that only the most relevant information is passed to the neighboring nodes.

The message-gating mechanism is distinct from the attention mechanism. While the attention score calculates the importance of each neighbor and performs weighting over the entire message provided by a particular neighbor, the message passing gate provides channel-wise weighting for a given neighbor message, focusing on the important channels. This synergistic effect of neighbor-wise and channel-wise message weighting makes FraudGT more expressive and better equipped to capture complex transactional patterns.

### 3.2.3 *Edge-Based Attention Bias.*

One of the challenges in financial fraud detection is the detection of rare, suspicious transactions from a large number of normal transactions. FraudGT addresses this issue by incorporating an edge-based attention bias. This bias helps the model pay more attention to suspicious transactions by adjusting the attention scores based on edge attributes. The attention bias term $\boldsymbol{b}_{ij}^{(h,l)} = \left(\mathbf{E}'^{(l)}_{ij}\right)^T \boldsymbol{W}_E^{(h,l)} \in \mathbb{R}^{d_h}$ is element-wise added to the original attention score as shown in Equation (8), with $\boldsymbol{W}_E^{(h,l)} \in \mathbb{R}^{d_e \times d_h}$ being a learnable weight vector and $\mathbf{E}'^{(l)}_{ij}$ being the feature of edge between $v_i$ and $v_j$. This adjustment ensures that transactions with attributes that are more indicative of fraud receive higher attention scores.

*3.2.4 Directed Multigraph Enhancements.* Financial transaction graphs are often modeled as directed multigraphs since multiple transactions can exist between accounts. Therefore, components that are beneficial for directed multigraph learning can potentially improve fraud detection performance. Egressy et al. [19] presents a set of enhancements—reverse message passing, port numbering, and ego ID—that perform feature augmentations to improve the expressivity of GNNs in directed multigraphs. We incorporate combinations of these enhancements into FraudGT to obtain different variants. Based on our comprehensive experiments, we observe that some of the variants can improve fraud detection accuracy while maintaining efficiency. The enhancements are described as follows:

- **Reverse Message Passing (RMP).** Standard GNNs only pass messages in the direction of edges in directed graphs. Nodes without incoming edges receive no messages from their neighbors and, therefore, cannot leverage the graph structure to refine their node features. To overcome this issue, bidirectional message passing is provided to allow communication in both directions. The message direction is indicated in the edge feature to let the model distinguish between incoming and outgoing edges, as illustrated in Figure 2.
- **Port Numbering.** Multiple transactions can exist between two accounts. Distinguishing edges from the same neighbor and edges from different neighbors can help the model detect more complicated fraudulent patterns. Port numbering [49] serves this purpose by assigning local IDs to each neighbor at a node (Figure 2). We assign each directed edge an incoming and outgoing port number, and edges coming from (or going to) the same node, receive the same incoming (or outgoing) port number.
- **Ego IDs.** Although RMP and port numbering help with detecting more suspicious patterns, they are not sufficient for detecting directed cycles. Ego IDs [64] were introduced to help detect cycles in graphs by marking a center node with a distinct (binary) feature so that it can be recognized when a sequence of messages cycles back around to it. We adopt this idea in our framework.

*3.2.5 Training and Prediction.* In this paper, we aim to predict the anomaly score of each edge (transaction). The final classifier is comprised of a simple feed-forward network MLP(·) and a sigmoid function $\sigma$. Let || denote vector concatenation. The prediction $\hat{y}_{ij}$ for an edge between nodes $v_i$ and $v_j$ is

$$\hat{y}_{ij} = \sigma(\mathrm{MLP}(\boldsymbol{h}_i \parallel \mathbf{E}'_{ij} \parallel \boldsymbol{h}_j)). \tag{9}$$

FraudGT is trained using a supervised learning approach. The objective is to minimize the binary cross-entropy loss between the predicted probabilities and the true labels of the transactions. Let $y_{ij}$ be the true label of the transaction between nodes $v_i$ and $v_j$, and $\hat{y}_{ij}$ be the predicted anomaly score of the transaction. The loss function $\mathcal{L}$ is defined as

$$\mathcal{L} = -\sum_{(i,j)\in\mathcal{E}} \left[ y_{ij} \log(\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij}) \right]. \tag{10}$$

**Table 1: Statistics of datasets used in the experiments.**

| Dataset | # Nodes | # Edges | Illicit Rate | Time Span | Split [%] |
|---|---|---|---|---|---|
| `AML Small-HI` | 515,088 | 5,078,345 | 0.102% | 10 days | 64/19/17 |
| `AML Small-LI` | 705,907 | 6,924,049 | 0.051% | 10 days | 64/19/17 |
| `AML Medium-HI` | 2,077,023 | 31,898,238 | 0.110% | 16 days | 61/17/22 |
| `AML Medium-LI` | 2,032,095 | 31,251,483 | 0.051% | 16 days | 61/17/22 |
| `AML Large-HI` | 2,116,168 | 179,702,229 | 0.124% | 97 days | 60/20/20 |
| `AML Large-LI` | 2,070,980 | 176,066,557 | 0.057% | 97 days | 60/20/20 |

*3.2.6 Inference Computational Complexity.* For a given batch of transactions, FraudGT extracts a subgraph from the sampled neighborhood and performs $L$ layers attention calculations across all edges. The edge-based message passing gate and attention bias are computed for every edge. Therefore, the complexity of attention computation is linear to the number of edges in the batch, which we denote as $|\mathcal{E}_{batch}|$. Each attention calculation is linear in $d_n$ and $d_e$, the number of hidden dimensions in the node embeddings and edge embeddings, respectively.

RMP and port numbering are feature augmentations that are only performed once during preprocessing and have linear time complexity in the total number of edges. Ego ID adds a binary feature to each node per sampled subgraph and has time complexity linear in the total number of nodes $|\mathcal{V}_{batch}|$ in the sampled subgraph. Therefore, the overall computational complexity of FraudGT on a batch of transactions is $O(|\mathcal{E}_{batch}|Ld + |\mathcal{V}_{batch}|)$, where $d = \max(d_n, d_e)$.

# 4 Experiments

In this section, we present comprehensive experiments evaluating FraudGT. We show that FraudGT matches or outperforms all other methods in fraud detection accuracy, while also being faster.

## 4.1 Experimental Setup

*4.1.1 Datasets.* Given the strict privacy regulations around financial data, real-world datasets are not readily available. While some commercial datasets exist, they are not publicly available [26, 48, 52]. Individual banks and institutions often only have access to their own transaction data, missing the broader context of customer behavior across multiple institutions. Furthermore, these datasets typically suffer from poor labeling, as many money laundering activities go undetected [53, 54], especially when they involve transactions across different banks. As a result, creating ground truth labels is particularly challenging in this domain. Instead, we use large-scale simulated money laundering data [3]. The simulator behind these datasets generates realistic financial transaction graphs by modeling agents (banks, companies, and individuals) in a virtual world. The generator uses well-established laundering patterns to add realistic money laundering (illicit) transactions. We use all three sizes of the synthetic datasets that are publicly available on Kaggle

**Table 2: Benchmark results of F1 scores (%) of various GNN methods. Standard deviations are calculated over 5 runs with different random seeds. We highlight the first and second best results.**

| | | Average Rank | `AML Small-HI` | `AML Small-LI` | `AML Medium-HI` | `AML Medium-LI` | `AML Large-HI` | `AML Large-LI` |
|---|---|---|---|---|---|---|---|---|
| | MLP | 21.2 | 0.42 ± 0.04 | 0.13 ± 0.05 | 0.06 ± 0.12 | 0.15 ± 0.02 | 0.66 ± 0.45 | 0.36 ± 0.03 |
| | LightGBM+GFs [5] | 11.3 | 62.86 ± 0.25 | 20.83 ± 1.50 | 59.48 ± 0.15 | 20.85 ± 0.38 | 58.03 ± 0.19 | 23.67 ± 0.11 |
| | XGBoost+GFs [5] | 8.8 | 63.23 ± 0.17 | 27.30 ± 0.33 | 65.70 ± 0.26 | 28.16 ± 0.14 | 42.68 ± 12.93 | 24.23 ± 0.12 |
| non-Multi-GNNs | GatedGCN [6] | 18.0 | 38.54 ± 2.25 | 17.18 ± 4.02 | 41.61 ± 4.57 | 11.90 ± 3.98 | 36.96 ± 1.88 | 8.97 ± 7.61 |
| | GAT [56] | 21.3 | 0.28 ± 0.15 | 0.13 ± 0.01 | 0.36 ± 0.07 | 0.12 ± 0.01 | 0.94 ± 0.15 | 0.35 ± 0.09 |
| | GIN [30, 61] | 16.5 | 40.04 ± 5.40 | 23.26 ± 3.56 | 45.40 ± 6.15 | 12.19 ± 3.01 | 36.32 ± 2.55 | 6.06 ± 4.73 |
| | GIN+RMP [33] | 14.2 | 45.03 ± 7.02 | 18.80 ± 2.55 | 53.26 ± 4.82 | 11.74 ± 2.00 | 59.29 ± 3.22 | 10.88 ± 4.95 |
| | GIN+Ports [49] | 18.2 | 54.83 ± 2.08 | 18.70 ± 1.08 | 41.96 ± 1.77 | 11.39 ± 5.11 | 40.15 ± 1.38 | 0.20 ± 0.28 |
| | GIN+Ego ID [64] | 15.2 | 46.03 ± 4.38 | 18.21 ± 3.67 | 52.84 ± 5.94 | 21.82 ± 2.13 | 53.31 ± 4.12 | 5.42 ± 5.58 |
| | GIN+EU [4] | 15.7 | 44.97 ± 5.41 | 23.14 ± 9.90 | 53.13 ± 7.89 | 17.96 ± 2.84 | 41.99 ± 2.24 | 4.88 ± 3.88 |
| | PNA [15] | 12.7 | 61.06 ± 5.65 | 17.98 ± 3.69 | 60.17 ± 2.39 | 31.66 ± 0.79 | 54.69 ± 6.66 | 4.10 ± 5.18 |
| | PNA+EU | 10.8 | 57.23 ± 5.61 | 26.43 ± 1.83 | 61.02 ± 2.02 | 26.51 ± 3.31 | 61.98 ± 5.71 | 3.96 ± 6.68 |
| Multi-GNNs | Multi-GIN [19] | 12.5 | 47.42 ± 2.93 | 22.31 ± 5.79 | 54.59 ± 2.25 | 18.72 ± 4.65 | 58.43 ± 5.09 | 17.53 ± 7.44 |
| | Multi-GIN+EU [19] | 10.8 | 57.12 ± 2.86 | 16.23 ± 3.23 | 62.25 ± 2.05 | 22.58 ± 2.40 | 61.50 ± 2.23 | 25.35 ± 1.43 |
| | Multi-PNA [19] | 9.2 | 68.19 ± 2.03 | 31.33 ± 2.58 | 67.22 ± 2.65 | 26.33 ± 2.90 | 51.85 ± 11.16 | 6.59 ± 8.60 |
| | Multi-PNA+EU [19] | 10.7 | 68.60 ± 3.36 | 27.79 ± 3.63 | 63.60 ± 1.58 | 17.95 ± 5.83 | 59.02 ± 4.63 | 4.65 ± 5.80 |
| Proposed Variants | FraudGT | 9.0 | 69.68 ± 1.58 | 28.69 ± 2.05 | 62.38 ± 0.87 | 24.02 ± 0.52 | 54.35 ± 1.65 | 11.02 ± 2.65 |
| | +RMP | 6.3 | 64.84 ± 2.00 | 33.02 ± 3.17 | 66.37 ± 0.47 | 27.01 ± 2.61 | 65.05 ± 1.19 | 19.17 ± 3.22 |
| | +Ports | 3.7 | 74.90 ± 0.55 | 44.17 ± 1.87 | 72.12 ± 1.18 | 38.62 ± 2.85 | 60.89 ± 2.50 | 30.40 ± 5.68 |
| | +Ego ID | 3.8 | 70.01 ± 3.47 | 34.22 ± 1.10 | 71.72 ± 1.29 | 32.59 ± 1.79 | 65.48 ± 0.91 | 27.94 ± 4.77 |
| | +Ports+Ego ID<br>(PE-FraudGT) | 1.8 | **76.41 ± 1.45** | 45.81 ± 1.14 | 74.22 ± 1.74 | 43.53 ± 1.76 | 68.64 ± 2.31 | 30.44 ± 2.76 |
| | +RMP+Ports+Ego ID<br>(Multi-FraudGT) | 1.2 | 76.13 ± 0.95 | **47.01 ± 2.22** | **75.93 ± 1.92** | **44.06 ± 5.27** | **73.34 ± 1.64** | **37.43 ± 4.94** |

[2], and for each size, we use one dataset with a higher illicit ratio (HI) and one with a lower illicit ratio (LI). The dataset sizes, illicit ratios, and split ratios among the training, validation, and test sets are provided in Table 1. The datasets are split temporally, i.e., we split the transactions after ordering them by their timestamps.

*4.1.2 Baselines.* We compare FraudGT with three categories of baselines representing the state-of-the-art (SOTA) work in financial fraud detection. The first category consists of computationally efficient baselines, including MLP [22], which performs classification directly on node and edge features, and LightGBM+GFs and XGBoost+GFs [5], which are gradient-boosting methods using pre-calculated graph-based features (GFs) and tree-based classifiers LightGBM [36] and XGBoost [11] to classify nodes or edges individually. This approach has produced SOTA results in financial applications [43, 58].

The second category consists of GNN models with edge features but without directed multigraphs enhancements, which we term non-Multi-GNNs. They include GatedGCN [6], GAT [56], GIN [30], GIN with reverse message passing (+RMP) [33], GIN with ports numbering (+Ports) [49], GIN with ego ID (+Ego ID) [64], GIN with edge updates (+EU) [4], PNA [15], and PNA with edge updates (+EU).

The third class of baselines is the SOTA GNN models tailored for directed multigraphs [19]: Multi-GINE, Multi-GINE with edge updates (+EU), Multi-PNA, and Multi-PNA with edge updates (+EU).

Lastly, in addition to FraudGT alone, we incorporate a combination of the directed multigraph enhancements described in Section 3.2.4—RMP, port numbering, and ego IDs. We obtain the following variants: FraudGT with RMP (+RMP), FraudGT with port numbering (+Ports), FraudGT with ego ID (+Ego ID), FraudGT with port numbering and ego ID (+Ports+Ego ID or PE-FraudGT), and FraudGT with all three enhancements (+RMP+Ports+Ego ID or Multi-FraudGT). We use neighborhood sampling [24] for all GNN-based models.

*4.1.3 Evaluation.* Since our datasets are very imbalanced, popular metrics for measuring accuracy are not suitable. Instead, we use the F1 score, consistent with previous works [3, 19] and aligns well with what banks and regulators use in real-world scenarios. Test performance is reported for the learned parameters of the highest validation performance. We used a POWER9 processor on the IBM Power System AC922 (8335-GTG) running at 2.3–3.8GHz frequency with a 10MB L3 cache size to perform graph sampling. We used an Nvidia V100 GPU with 32GB of memory to perform training and inference.

## 4.2 Experimental Results

*4.2.1 Classification Results.* Table 2 lists the results of each method across the datasets, with standard deviations calculated over 5 runs with different random seeds. We make the following observations. First, the proposed FraudGT and its variants demonstrate leading performance on all AML datasets, as indicated by their average rankings. Notably, the vanilla FraudGT is competitive with or outperforms non-Multi-GNN baselines, such as GatedGCN and PNA, on `AML Small-HI`, `AML Small-LI`, `AML Medium-HI`, and `AML Large-LI`. For example, on the `AML Small-HI` dataset, FraudGT

**Table 3: Ablation study results of F1 scores (%). Standard deviations are calculated over 5 runs with different random seeds. We highlight the first and second best results.**

| | `AML Small-HI` | `AML Small-LI` | `AML Medium-HI` | `AML Medium-LI` | `AML Large-HI` | `AML Large-LI` |
|---|---|---|---|---|---|---|
| FraudGT w/o Edge-based Message Passing | 1.67 ± 1.28 | 0.25 ± 0.30 | 1.88 ± 0.82 | 0.15 ± 0.07 | 7.98 ± 1.2 | 0.56 ± 0.14 |
| FraudGT w/o Edge-based Attention Bias | 68.82 ± 2.55 | 24.87 ± 4.3 | 58.39 ± 2.85 | 11.37 ± 3.88 | 52.55 ± 1.49 | 9.74 ± 2.55 |
| FraudGT | **69.68** ± 1.58 | **28.69** ± 2.05 | **62.38** ± 0.87 | **24.02** ± 0.52 | **54.35** ± 1.65 | **11.02** ± 2.65 |

achieves an average F1 score of 69.68%, representing a significant performance improvement of 8.62% over the best-performing non-Multi-GNN baseline, PNA. Second, incorporating directed multigraph enhancements significantly boosts the F1 score of FraudGT. The combination of these enhancements creates a synergistic effect. The final Multi-FraudGT variant consistently outperforms all existing methods, including the previously best-performing Multi-GNNs, across all datasets, achieving a 7.8–17.8% improvement. This demonstrates the effectiveness of these enhancements in improving model performance. Lastly, the results highlight that models like MLP, which does not utilize graph structure, and GAT, which does not effectively leverage multidimensional edge information, perform poorly. These findings emphasize the critical role of the effective use of graph structure and edge information in achieving high accuracy in financial fraud detection. Overall, the results confirm the efficacy of FraudGT and its variants for addressing financial fraud challenges.

*4.2.2 Inference Throughput and Latency.* Figure 3 compares the inference throughput and latency of FraudGT and its variants and a subset of the other methods across the datasets. FraudGT and its variants not only achieve high accuracy but also demonstrate higher throughput and lower latency than the other methods. For instance, the variant PE-FraudGT achieves close to the best F1 scores on all datasets (only outperformed by Multi-FraudGT) while having approximately 2.4 times higher throughput and lower per-batch latency on average compared to the SOTA model Multi-PNA. The primary reason for the computational efficiency of PE-FraudGT is that it requires a much smaller sampled neighborhood to achieve a high F1 score. We find that existing methods require a large sampled neighborhood to achieve their best F1 score, resulting in lower throughput and higher latency. For example, on the `AML Small-HI` dataset, PE-FraudGT samples a subgraph with an average of 18k nodes and 162k edges. In contrast, Multi-PNA requires sampling a subgraph with an average of 67k nodes and 729k edges, which is 4.5× larger. Similarly, on the `AML Medium-HI` dataset, the subgraph sampled by PE-FraudGT consists of 23k nodes and 252k edges, while the subgraph sampled by Multi-PNA consists of 93k nodes and 1092k edges. Therefore, to achieve its best F1 score, Multi-PNA requires significantly more computation, and it still lags 7.8–17.8% behind PE-FraudGT in F1 score.

*4.2.3 Ablation Study.* We perform an ablation study on the components introduced in FraudGT, specifically focusing on the *edge-based message passing gate* and the *edge-based attention bias*. Table 3 presents the results of the F1 scores. From the results, we see that the edge-based message passing gate has a significant impact on the overall performance. We see that removing the edge-based message passing gate (FraudGT w/o Edge-based Message Passing) leads to a drastic reduction in F1 scores, highlighting its critical role in discerning important transactional features. The model without edge-based

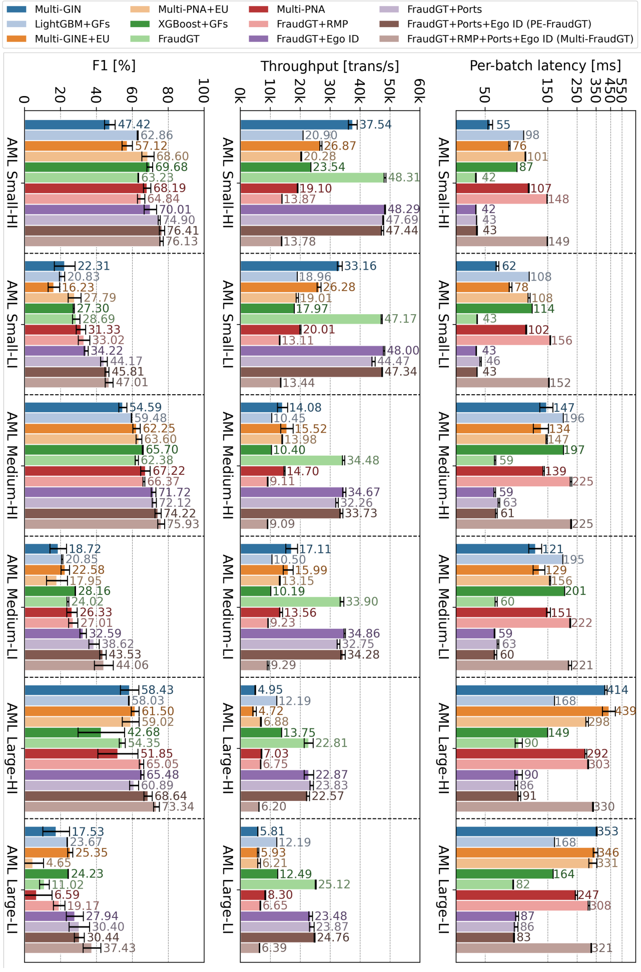


**Figure 3: Average throughput and per-batch latency of our FraudGT variants and GNN baselines, with models ordered by average rank. The standard deviation over 5 runs with different random seeds is also plotted as an error bar at the end of each bar. Throughput is defined as the number of transactions processed per second (trans/s), measured by inputting a batch of $T$ target transactions into the model and timing the average per-batch latency $l$ to generate the output. The average throughput is then calculated as $T/l$.**

attention bias (FraudGT w/o Edge-based Attention Bias) performs better than the model without the edge-based message passing gate, but still falls short of FraudGT with both components, indicating that the edge-based attention also contributes significantly to the model's performance. These findings demonstrate the importance of both the edge-based message passing gate and edge-based attention in achieving state-of-the-art performance in financial fraud detection.

## 5 Conclusion

In this paper, we introduced FraudGT, a simple, effective, and efficient graph transformer model designed for financial fraud detection in transaction graphs. FraudGT addresses several key challenges inherent in financial transaction graphs, including learning complex patterns, effective use of edge information, and computational efficiency. Leveraging the strengths of GTs to capture complex patterns and relationships within financial transaction data, FraudGT incorporates an edge-based message passing gate and an edge-based attention bias, allowing the model to focus on critical transactional features indicative of fraudulent activities. Through extensive evaluation on various publicly-available large-scale synthetic datasets, we show that FraudGT significantly outperforms existing baselines and achieves state-of-the-art performance. While synthetic datasets provide an essential testing ground due to privacy concerns in real-world financial data, for future work, it would be valuable to evaluate FraudGT on real-world financial fraud datasets. This would provide further validation of FraudGT's applicability in real-world settings and enhance its potential for deployment in practical financial fraud detection systems.

## Acknowledgments

This work is funded by the MIT-IBM AI Watson Lab. In addition, Julian Shun is supported by NSF awards #CCF-1845763, #CCF-2316235, and #CCF-2403237.

## References

[1] Uri Alon and Eran Yahav. 2020. On the Bottleneck of Graph Neural Networks and Its Practical Implications. In *International Conference on Learning Representations (ICLR)*.

[2] Erik Altman. 2019. IBM Transactions for Anti-Money Laundering (AML). https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml.

[3] Erik Altman, Jovan Blanuša, Luc Von Niederhäusern, Béni Egressy, Andreea Anghel, and Kubilay Atasu. 2024. Realistic Synthetic Financial Transactions for Anti-Money Laundering Models. *Advances in Neural Information Processing Systems (NeurIPS)* 36 (2024).

[4] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational Inductive Biases, Deep Learning, and Graph Networks. *arXiv preprint arXiv:1806.01261* (2018).

[5] Jovan Blanuša, Maximo Cravero Baraja, Andreea Anghel, Luc von Niederhäusern, Erik Altman, Haris Pozidis, and Kubilay Atasu. 2024. Graph Feature Preprocessor: Real-Time Extraction of Subgraph-Based Features from Transaction Graphs. *arXiv preprint arXiv:2402.08593* (2024).

[6] Xavier Bresson and Thomas Laurent. 2017. Residual Gated Graph ConvNets. *arXiv preprint arXiv:1711.07553* (2017).

[7] Bokai Cao, Mia Mao, Siim Viidu, and S Yu Philip. 2017. HitFraud: A Broad Learning Approach for Collective Fraud Detection in Heterogeneous Information Networks. In *IEEE International Conference on Data Mining (ICDM)*. 769–774.

[8] Shaosheng Cao, XinXing Yang, Cen Chen, Jun Zhou, Xiaolong Li, and Yuan Qi. 2019. TitAnt: Online Real-time Transaction Fraud Detection in Ant Financial. *Proceedings of the VLDB Endowment (PVLDB)* 12, 12 (2019).

[9] Mário Cardoso, Pedro Saleiro, and Pedro Bizarro. 2022. LaundroGraph: Self-Supervised Graph Representation Learning for Anti-Money Laundering. In *Proceedings of the ACM International Conference on AI in Finance*. 130–138.

[10] Liang Chen, Jiaying Peng, Yang Liu, Jintang Li, Fenfang Xie, and Zibin Zheng. 2020. Phishing Scams Detection in Ethereum Transaction Network. *ACM Transactions on Internet Technology (TOIT)* 21, 1 (2020), 1–16.

[11] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 785–794.

[12] Tianyi Chen and Charalampos Tsourakakis. 2022. AntiBenford Subgraphs:Unsupervised Anomaly Detection in Financial Networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 2762–2770.

[13] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. 2020. Can Graph Neural Networks Count Substructures? *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 10383–10395.

[14] Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna. 2019. On the Equivalence Between Graph Isomorphism Testing and Function Approximation with GNNs. *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[15] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. 2020. Principal Neighbourhood Aggregation for Graph Nets. *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 13260–13271.

[16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *Advances in Neural Information Processing Systems (NeurIPS)* 29 (2016).

[17] Zhihao Ding, Jieming Shi, Qing Li, and Jiannong Cao. 2023. Effective Multi-Graph Neural Networks for Illicit Account Detection on Cryptocurrency Transaction Networks. *arXiv preprint arXiv:2309.02460* (2023).

[18] Yifan Duan, Guibin Zhang, Shilong Wang, Xiaojiang Peng, Ziqi Wang, Junyuan Mao, Hao Wu, Xinke Jiang, and Kun Wang. 2024. CaT-GNN: Enhancing Credit Card Fraud Detection via Causal Temporal Graph Neural Networks. *arXiv preprint arXiv:2402.14708* (2024).

[19] Béni Egressy, Luc Von Niederhäusern, Jovan Blanuša, Erik Altman, Roger Wattenhofer, and Kubilay Atasu. 2024. Provably Powerful Graph Neural Networks for Directed Multigraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 38. 11838–11846.

[20] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *International Conference on Machine Learning (ICML)*. 1263–1272.

[21] Liyu Gong and Qiang Cheng. 2019. Exploiting Edge Features for Graph Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9211–9219.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

[23] Oscar M Granados and Andrés Vargas. 2022. The Geometry of Suspicious Money Laundering Activities in Financial Networks. *EPJ Data Science* 11, 1 (2022), 6.

[24] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[25] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. 2022. A Survey on Vision Transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2022), 87–110.

[26] Daniel A Harris, Kyla L Pyndiura, Shelby L Sturrock, and Rebecca AG Christensen. 2022. Using real-world transaction data to identify money laundering: Leveraging traditional regression and machine learning techniques. *STEM Fellowship Journal* 7, 1 (2022), 21–32.

[27] Jing He, Jiao Tian, Yuanyuan Wu, Xinyi Cia, Kai Zhang, Mengjiao Guo, Hui Zheng, Junfeng Wu, and Yimu Ji. 2021. An Efficient Solution to Detect Common Topologies in Money Launderings Based on Coupling and Connection. *IEEE Intelligent Systems* 36, 1 (2021), 64–74.

[28] Waleed Hilal, S Andrew Gadsden, and John Yawney. 2022. Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances. *Expert Systems with Applications (ESWA)* 193 (2022), 116429.

[29] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 895–904.

[30] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020. Strategies for Pre-training Graph Neural Networks. In *International Conference on Learning Representations (ICLR)*.

[31] Kexin Huang, Cao Xiao, Lucas M Glass, Marinka Zitnik, and Jimeng Sun. 2020. SkipGNN: Predicting Molecular Interactions with Skip-Graph Networks. *Scientific Reports* 10, 1 (2020), 1–16.

[32] Woochang Hyun, Jaehong Lee, and Bongwon Suh. 2023. Anti-Money Laundering in Cryptocurrency via Multi-Relational Graph Neural Network. In *Advances in Knowledge Discovery and Data Mining: Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. 118–130.

[33] Guillaume Jaume, An-phi Nguyen, María Rodríguez Martínez, Jean-Philippe Thiran, and Maria Gabrani. 2019. EDGNN: A Simple and Powerful GNN for Directed Labeled Graphs. *arXiv preprint arXiv:1904.08745* (2019).

[34] Jiaxin Jiang, Yuan Li, Bryan Hooi, Bingsheng He, Jia Chen, and Johan Kok Zhi Kang. 2024. Spade: A Real-Time Fraud Detection Framework on Evolving Graphs. *Proceedings of the VLDB Endowment (PVLDB)* 16 (2024), 461–474.

[35] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. AMMUS: A Survey of Transformer-Based Pretrained Models in Natural Language Processing. *arXiv preprint arXiv:2108.05542* abs/2108.05542 (2021).

[36] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[37] Jinwoo Kim, Saeyoon Oh, and Seunghoon Hong. 2021. Transformers Generalize DeepSets and Can Be Extended to Graphs & Hypergraphs. *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), 28016–28028.

[38] Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[39] Kezhi Kong, Jiuhai Chen, John Kirchenbauer, Renkun Ni, C Bayan Bruss, and Tom Goldstein. 2023. GOAT: A Global Transformer on Large-Scale Graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*. 17375–17390.

[40] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[41] Shenghua Liu, Bryan Hooi, and Christos Faloutsos. 2017. HoloScope: Topology-and-Spike Aware Fraud Detection. In *Proceedings of the ACM on Conference on Information and Knowledge Management (CIKM)*. 1539–1548.

[42] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. 2018. Heterogeneous Graph Neural Networks for Malicious Account Detection. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 2077–2085.

[43] Wai Weng Lo, Gayan K Kulatilleke, Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. 2023. Inspection-L: Self-Supervised GNN Node Embeddings for Money Laundering Detection in Bitcoin. *Applied Intelligence* 53, 16 (2023), 19406–19417.

[44] Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. 2019. Invariant and Equivariant Graph Networks. In *International Conference on Learning Representations (ICLR)*.

[45] Erxue Min, Runfa Chen, Yatao Bian, Tingyang Xu, Kangfei Zhao, Wenbing Huang, Peilin Zhao, Junzhou Huang, Sophia Ananiadou, and Yu Rong. 2022. Transformer for Graphs: An Overview from Architecture Perspective. *arXiv preprint arXiv:2202.08455* (2022).

[46] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. 2022. Recipe for a General, Powerful, Scalable Graph Transformer. *Advances in Neural Information Processing Systems (NeurIPS)* 35 (2022), 14501–14515.

[47] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 12179–12188.

[48] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2021. xFraud: Explainable Fraud Transaction Detection. *Proceedings of the VLDB Endowment (PVLDB)* 15, 3 (2021), 427–436.

[49] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. 2019. Approximation Ratios of Graph Neural Networks for Combinatorial Problems. *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[50] Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. 2020. Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation. In *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)*, Vol. 14. 626–637.

[51] Yunchong Song, Chenghu Zhou, Xinbing Wang, and Zhouhan Lin. 2023. Ordered GNN: Ordering Message Passing to Deal with Heterophily and Over-Smoothing. In *International Conference on Learning Representations (ICLR)*.

[52] Michele Starnini, Charalampos E Tsourakakis, Maryam Zamanipour, André Panisson, Walter Allasia, Marco Fornasiero, Laura Li Puma, Valeria Ricci, Silvia Ronchiadin, Angela Ugrinoska, et al. 2021. Smurf-Based Anti-Money Laundering in Time-Evolving Transaction Networks. In *European Conference on Machine Learning and Data Mining (ECML PKDD)*. 171–186.

[53] United Nations Office on Drugs and Crime. 2022. Money Laundering. https://www.unodc.org/unodc/en/money-laundering/overview.html Accessed: 2024-10-05.

[54] U.S. Department of the Treasury. 2022. National Money Laundering Risk Assessment. https://home.treasury.gov/system/files/136/2022-National-Money-Laundering-Risk-Assessment.pdf 21 pages.

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[56] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.

[57] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xiong. 2019. FDGARS: Fraudster Detection via Graph Convolutional Networks in Online App Review System. In *Proceedings of the International Conference on World Wide Web (WWW)*. 310–316.

[58] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. *arXiv preprint arXiv:1908.02591* (2019).

[59] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys (CSUR)* 55, 5 (2022), 1–37.

[60] Sheng Xiang, Mingzhi Zhu, Dawei Cheng, Enxia Li, Ruihui Zhao, Yi Ouyang, Ling Chen, and Yefeng Zheng. 2023. Semi-Supervised Credit Card Fraud Detection via Attribute-Driven Graph Representation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 14557–14565.

[61] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations (ICLR)*.

[62] Kuan Yan, Junbin Gao, and Dmytro Matsypura. 2023. FIW-GNN: A Heterogeneous Graph-Based Learning Model for Credit Card Fraud Detection. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 1–10.

[63] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Bad for Graph Representation?. In *Advances in Neural Information Processing Systems (NeurIPS)*. 28877–28888.

[64] Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, and Jure Leskovec. 2021. Identity-Aware Graph Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 35. 10737–10745.

[65] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2022. Scaling Vision Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12104–12113.

[66] Shuaicheng Zhang, Qiang Ning, and Lifu Huang. 2022. Extracting Temporal Event Relation with Syntax-Guided Graph Transformer. In *North American Chapter of the Association for Computational Linguistics (NAACL)*. 379–390.

[67] Tianyi Zhao, Yang Hu, Linda R Valsdottir, Tianyi Zang, and Jiajie Peng. 2021. Identifying Drug–Target Interactions Based on Graph Convolutional Network and Deep Neural Network. *Briefings in Bioinformatics* 22, 2 (2021), 2141–2150.

[68] Qiwei Zhong, Yang Liu, Xiang Ao, Binbin Hu, Jinghua Feng, Jiayu Tang, and Qing He. 2020. Financial Defaulter Detection on Online Credit Payment via Multi-View Attributed Heterogeneous Information Network. In *Proceedings of the International Conference on World Wide Web (WWW)*. 785–795.