International Journal for Modern Trends in Science and Technology
Volume 12, Issue 07, pages 191-205.
ISSN: 2455-3778 online
Available online at: http://www.ijmtst.com/vol12issue06.html
DOI: https://doi.org/10.5281/zenodo.21325914

Graph Neural Network Based Fraud Detection in Blockchain Transactions

P. B. Shalini | Dr. B. Kezia Rani

Department of Computer Science and Engineering, Adikavi Nannaya University, Rajahmundry, India

To Cite this Article
P. B. Shalini & Dr. B. Kezia Rani (2026). Graph Neural Network Based Fraud Detection in Blockchain Transactions. International Journal for Modern Trends in Science and Technology, 12(07), 191-205. https://doi.org/10.5281/zenodo.21325914

Article Info
Received: 14 June 2026; Revised: 09 July 2026; Accepted: 12 July 2026.

KEYWORDS

Graph Neural Networks, Blockchain Fraud Detection, Bitcoin Transactions, Graph Convolutional Network, Graph Attention Network, GraphSAGE, Temporal Graph Neural Network, Elliptic Dataset, Semi-Supervised Learning, Cryptocurrency Security.

ABSTRACT

The increasing use of cryptocurrencies for illegal financial activities, including money laundering, ransomware payments, and fraudulent fund transfers, has made fraud detection a critical challenge in blockchain ecosystems. Due to the pseudonymous and decentralized nature of blockchain platforms such as Bitcoin, conventional fraud detection techniques struggle to capture relational dependencies between transactions. This paper presents a Graph Neural Network (GNN)-based framework for detecting fraudulent transactions using the Elliptic Bitcoin dataset. Unlike existing studies that primarily evaluate a single GNN architecture, this work systematically compares four complementary Graph Neural Network models—Graph Convolutional Network (GCN), GraphSAGE, Graph Attention Network (GAT), and Temporal Graph Neural Network (Temporal GNN)—under identical training conditions. The models were trained using semi-supervised learning with class-weighted optimization to address severe class imbalance. Experimental results show that GraphSAGE achieved the best overall performance with 97.19% accuracy and an 87.48% F1-score, outperforming the other models in balancing precision and recall. The trained models were further integrated into an interactive Streamlit dashboard supporting transaction analysis, multi-model comparison, graph visualization, and live Bitcoin transaction monitoring. The results demonstrate that systematic comparative evaluation of multiple GNN architectures under identical experimental conditions provides a reliable and practical framework for blockchain fraud detection.

1. INTRODUCTION

The emergence of cryptocurrencies has significantly transformed the traditional financial landscape by enabling decentralized, peer-to-peer transactions without the need for a central authority or intermediary institution. Bitcoin, the most widely adopted cryptocurrency, operates on blockchain technology, which maintains a distributed and immutable ledger that

records every transaction in a transparent and verifiable manner. Once a transaction is recorded on the blockchain, altering or removing it becomes practically impossible, thereby establishing trust in cryptocurrency networks.

However, the same properties that make blockchain attractive, namely decentralization and pseudonymity, also create major challenges for fraud detection. Users can perform transactions without revealing their real-world identities, which enables malicious actors to exploit the system for illegal financial activities such as money laundering, ransomware payments, Ponzi schemes, darknet transactions, and fraudulent fund transfers. As cryptocurrency adoption continues to grow, ensuring transaction security has become increasingly important for researchers, regulatory authorities, and financial institutions.

Traditional fraud detection systems commonly rely on machine learning approaches such as Logistic Regression, Decision Trees, Random Forest, and Support Vector Machines. These methods treat each transaction as an independent data sample and fail to capture the complex relational dependencies among transactions. In blockchain systems, fraudulent behavior often spans multiple wallet addresses and transaction chains, making isolated transaction-level analysis insufficient for detecting coordinated fraud patterns.

Blockchain transactions can naturally be represented as graphs, where nodes represent transactions and directed edges represent the flow of funds between them. This representation captures both direct and indirect relationships among transactions, providing richer contextual information than flat feature vectors. Graph Neural Networks (GNNs) are specifically designed to learn from graph-structured data through iterative neighborhood aggregation, allowing them to capture both local and global structural patterns effectively. This makes GNNs highly suitable for blockchain fraud detection.

Although Graph Neural Networks have demonstrated promising performance for blockchain fraud detection, most existing studies primarily investigate a single GNN architecture, making it difficult to compare their relative effectiveness under identical experimental conditions. Furthermore, limited research combines comprehensive comparative evaluation with a deployable blockchain fraud analysis platform capable of supporting practical transaction monitoring. To address these gaps, this work presents a unified Graph Neural Network-based fraud detection framework using the Elliptic Bitcoin dataset. Four complementary GNN architectures, namely Graph Convolutional Network (GCN), GraphSAGE, Graph Attention Network (GAT), and Temporal Graph Neural Network (Temporal GNN), were implemented and evaluated using identical preprocessing, training, and evaluation settings to ensure a fair and systematic comparison.

Beyond offline model evaluation, the proposed framework was integrated into an interactive Streamlit-based blockchain fraud analysis dashboard supporting transaction analysis, graph visualization, multi-model prediction, and live transaction monitoring. This practical implementation demonstrates the applicability of graph-based deep learning techniques in assisting blockchain fraud analysis while providing an intuitive environment for comparing the behavior of multiple GNN architectures.

The major contributions of this work are summarized as follows:

- Development of a unified blockchain fraud detection framework incorporating four complementary Graph Neural Network architectures.
- Systematic comparative evaluation of GCN, GraphSAGE, GAT, and Temporal GNN under identical preprocessing, training, and evaluation conditions.
- Effective handling of severe class imbalance through semi-supervised learning and class-weighted optimization.
- Integration of the trained models into an interactive Streamlit-based dashboard for transaction analysis, graph visualization, multi-model prediction, and live monitoring.
- Comprehensive performance analysis demonstrating that GraphSAGE provides the most balanced performance for blockchain fraud detection on the Elliptic Bitcoin dataset.

2. LITERATURE REVIEW

Fraud detection in blockchain transactions has emerged as an important research area due to the rapid adoption of cryptocurrencies and the increasing occurrence of illicit activities such as money laundering, ransomware payments, and fraudulent transfers. Although

blockchain provides transparency and immutability through distributed ledger technology, its decentralized and pseudonymous nature makes direct identification of malicious entities difficult. Traditional machine learning techniques are not well suited for this problem because they analyze transactions independently and fail to capture the relational structure inherent in blockchain data. This limitation has motivated the use of Graph Neural Networks (GNNs), which model blockchain transactions as graphs and exploit structural relationships for improved fraud detection.

Kipf and Welling introduced Graph Convolutional Networks (GCNs) for semi-supervised node classification tasks. GCNs aggregate information from neighboring nodes and effectively capture local structural patterns. Due to this capability, GCNs have been widely adopted in fraud detection tasks involving graph-structured financial data. However, GCNs mainly operate on static graphs and face scalability limitations when applied to very large transaction networks.

Velickovic et al. proposed Graph Attention Networks (GAT), which extend graph convolution by incorporating an attention mechanism. Unlike GCN, which treats all neighboring nodes equally, GAT assigns different importance weights to neighboring nodes through learned attention coefficients. This selective aggregation improves the model's ability to focus on highly relevant suspicious relationships. However, the attention mechanism increases computational complexity and introduces a larger number of trainable parameters.

Weber et al. conducted one of the most influential studies in blockchain fraud detection using the Elliptic Bitcoin dataset. Their work demonstrated that graph-based learning approaches significantly outperform traditional feature-based machine learning models in detecting illicit Bitcoin transactions. This study established the Elliptic dataset as a benchmark for graph-based fraud detection research.

Liu et al. investigated the application of Graph Neural Networks for blockchain fraud detection and demonstrated that graph representation learning significantly improves the identification of illicit transactions compared with conventional machine learning techniques. Their study further highlighted the importance of exploiting structural relationships among blockchain transactions to achieve more accurate fraud detection, reinforcing the effectiveness of graph-based learning approaches in cryptocurrency security.

Hamilton, Ying, and Leskovec proposed GraphSAGE, an inductive representation learning framework that generates embeddings by sampling and aggregating neighborhood information. Unlike GCN, GraphSAGE can generalize to unseen nodes, making it highly suitable for streaming transaction environments where new blockchain transactions continuously arrive. Its primary advantage lies in scalability and inductive learning capability.

Pareja et al. introduced EvolveGCN, which extends graph convolution to dynamic graphs by allowing model parameters to evolve over time.

Rossi et al. studied Temporal Graph Networks for dynamic graph learning, showing that temporal modeling improves performance in environments where structural patterns evolve continuously. These methods are highly relevant for fraud detection because fraudulent behaviors often change over time.

Zhang et al. proposed a temporal-aware Graph Neural Network for cryptocurrency fraud detection, demonstrating that incorporating temporal information improves the identification of evolving fraudulent transaction patterns. Their work showed that modeling temporal dependencies enables Graph Neural Networks to better capture changes in transaction behavior, making temporal graph learning an important direction for blockchain fraud detection.

Several researchers have explored advanced graph learning approaches for financial fraud detection.

Chen et al. proposed ensemble GNN frameworks that combine multiple graph models to improve prediction performance, though at the cost of increased computational complexity.

Zhang et al. explored variational graph autoencoders for fraud detection, learning latent graph representations but facing challenges related to interpretability and false positives. Wang et al. and Cai et al. introduced heterogeneous graph neural networks that incorporate multiple node and relation types such as wallets, exchanges, and transactions, enabling richer representation learning but requiring significantly more labeled data and computational resources.

Asiri and Somasundaram applied Graph Convolutional Networks to the Elliptic Bitcoin dataset and reported very strong performance, achieving 98.5% accuracy with

an AUC of 0.9444. Their results further validated the effectiveness of graph-based deep learning for cryptocurrency fraud detection and serve as a key baseline reference for the present work.

Despite the significant progress achieved in blockchain fraud detection using Graph Neural Networks, several research challenges remain. Most existing studies primarily evaluate a single GNN architecture, making it difficult to understand the comparative strengths and limitations of different graph learning approaches under identical experimental conditions. Furthermore, limited attention has been given to integrating comparative model evaluation with practical blockchain fraud analysis systems capable of supporting real-time transaction monitoring. To address these gaps, the present work systematically evaluates four complementary GNN architectures—GCN, GraphSAGE, GAT, and Temporal GNN—using a unified training and evaluation framework on the Elliptic Bitcoin dataset. In addition, the trained models are integrated into an interactive dashboard that demonstrates the practical applicability of graph-based deep learning for blockchain fraud analysis.

3. PROBLEM STATEMENT

Blockchain technology enables secure and decentralized transaction processing, but its pseudonymous nature introduces significant challenges in identifying fraudulent activities. Transactions in blockchain networks form highly interconnected graph structures where malicious actors often exploit indirect relationships to conceal illicit behavior across multiple hops. Traditional fraud detection methods treat transactions as independent records and therefore fail to capture these complex relational dependencies, making them ineffective for detecting coordinated fraud patterns.

A major challenge in blockchain fraud detection is the severe class imbalance present in real-world datasets. In the Elliptic Bitcoin dataset used in this work, only 4,545 out of 46,564 labeled transactions are classified as illicit, representing approximately 9.7% of labeled data. This imbalance makes it difficult for conventional models to learn meaningful patterns for the minority fraudulent class without becoming heavily biased toward the majority licit class.

Another challenge is the temporal evolution of fraud patterns. Fraudulent entities continuously adapt their strategies to evade detection by modifying transaction behavior over time. A model trained only on static structural information may fail to generalize to newly emerging fraud patterns. Therefore, effective fraud detection requires models capable of capturing both structural dependencies and temporal dynamics.

Scalability is also a critical concern. Blockchain networks contain massive volumes of transactions, with the Elliptic graph alone consisting of more than 200,000 transaction nodes and over 234,000 directed edges. Any practical fraud detection system must therefore balance detection accuracy with computational efficiency to support real-world deployment.

To address these challenges, this work presents a unified Graph Neural Network-based fraud detection framework that systematically evaluates multiple graph learning architectures within a common experimental setting. By implementing GCN, GraphSAGE, GAT, and Temporal Graph Neural Network models under identical preprocessing, training, and evaluation conditions, the proposed framework enables a fair comparison of their effectiveness in detecting fraudulent blockchain transactions. In addition to improving the understanding of different GNN architectures for blockchain fraud detection, the framework also demonstrates its practical applicability through deployment in an interactive fraud analysis dashboard capable of supporting transaction analysis, graph visualization, multi-model prediction, and live monitoring.

4. PROPOSED METHODOLOGY

The proposed methodology presents a unified Graph Neural Network-based framework for blockchain fraud detection, beginning with the raw Elliptic Bitcoin dataset and concluding with trained models deployed within an interactive fraud analysis dashboard. The framework follows a systematic pipeline consisting of data loading, preprocessing, graph construction, model training, performance evaluation, and deployment. Unlike conventional approaches that evaluate a single Graph Neural Network architecture, the proposed framework enables a fair and consistent comparative evaluation of four complementary GNN models under identical experimental conditions. This unified pipeline ensures

that performance differences arise from the learning capabilities of the individual architectures rather than variations in preprocessing, training strategy, or evaluation methodology.

Furthermore, the modular design of the proposed framework enables each stage of the workflow to be independently analyzed and optimized while maintaining overall consistency. This structured approach improves the reliability and reproducibility of the experimental results and facilitates the practical deployment of Graph Neural Network models for blockchain fraud detection.

The proposed framework is designed to ensure systematic processing of blockchain transaction data while preserving both structural and temporal information. Each stage in the pipeline contributes to improving fraud detection performance, from feature engineering and graph construction to model optimization and evaluation. The overall workflow enables efficient learning of suspicious transaction patterns using graph-based deep learning techniques.

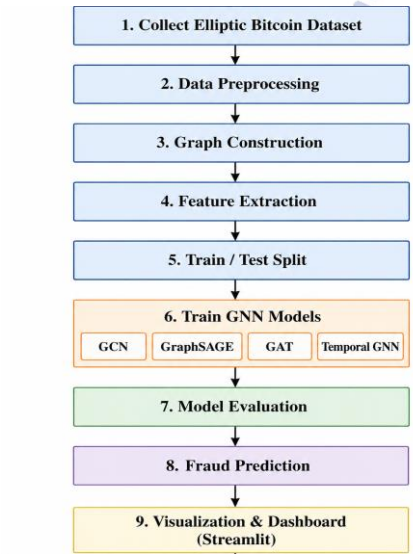

Fig -1: Proposed Methodology Flowchart

A. Dataset Loading

The pipeline begins by loading the three raw files of the Elliptic Bitcoin dataset:

- Transaction feature file
- Edge list file
- Class label file

The feature file contains one row per transaction with a transaction identifier followed by 166 numerical feature values. The edge file stores source-destination transaction pairs representing directed fund flows. The class file contains transaction labels indicating whether a transaction is licit or illicit.

The three files are merged using the transaction identifier so that each node contains both feature information and class labels. In the original dataset, class labels are represented as strings, where class 1 denotes illicit transactions and class 2 denotes licit transactions. For model training, these labels are converted into numerical form:

- Illicit → 1
- Licit → 0
- Unlabeled → −1

Unlabeled transactions are retained in the graph because they contribute structural information during message passing, even though they are excluded from loss computation.

B. Data Preprocessing

Before training, the dataset undergoes preprocessing to improve data quality and ensure numerical stability.

Missing feature values are handled using mean imputation. In this process, the mean value of each feature column is computed and used to replace missing entries. This preserves the statistical distribution of the dataset while avoiding information loss caused by dropping incomplete rows.

After imputation, feature normalization is performed using StandardScaler. Each feature is transformed to have zero mean and unit variance:

$$x' = (x - \mu) / \sigma$$

where: x = original feature value; μ = mean of feature; σ = standard deviation.

Normalization is essential because Graph Neural Networks are sensitive to feature scale differences. Without scaling, high-magnitude features such as transaction amounts may dominate learning.

The fitted scaler is saved after training and reused during inference so that incoming live transactions undergo the same transformation.

C. Graph Construction

After preprocessing, the transaction data is converted into a graph structure using PyTorch Geometric's Data object. Each transaction is represented as a node:

$$V = \{v1, v2, ..., vn\}$$

where n = 203,769 transaction nodes. Each fund transfer between transactions becomes a directed edge:

$$E = \{(u, v) \mid u, v \in V\} \quad (3)$$

where |E| = 234,355. Thus, the blockchain network is modeled as a directed graph:

$$G = (V, E, X) \quad (4)$$

where V = set of nodes (transactions); E = set of edges (fund flow); X = feature matrix. The feature matrix is represented as:

$$X \in \mathbb{R}^{N \times 166} \quad (5)$$

where each node contains a 166-dimensional feature vector. Directed edges are preserved because transaction flow direction is important for fraud detection. Fraudulent transactions often propagate funds through chains of transfers, and edge direction helps the model distinguish between incoming and outgoing fund movements.

D. Train, Validation, and Test Split

Only labeled transactions are used for supervised loss computation. The labeled subset contains 46,564 transactions. These labeled nodes are split into:

- Training set: 70%
- Validation set: 15%
- Test set: 15%

Boolean mask tensors are created to identify nodes belonging to each partition.

Since the dataset is highly imbalanced, class weighting is applied instead of graph resampling. Resampling in graph data can distort neighborhood relationships and negatively affect message passing.

Balanced class weights are computed using:

$$wc = N / (C \times nc) \quad (6)$$

where: N = total training samples; C = number of classes; nc = samples in class c. This gives higher weight to the minority illicit class so that misclassification of fraudulent transactions contributes more strongly to the loss function.

E. Graph Neural Network Architectures

Four Graph Neural Network architectures were implemented and evaluated in this work: Graph Convolutional Network (GCN), GraphSAGE, Graph Attention Network (GAT), and Temporal Graph Neural Network (Temporal GNN). All models take 166-dimensional node features as input and produce binary classification outputs representing licit and illicit transactions.

F. Graph Convolutional Network (GCN)

Graph Convolutional Network (GCN) serves as the baseline model in this work. It learns node representations by aggregating information from neighboring transactions through graph convolution operations. This enables the model to capture local structural fraud patterns such as suspicious fund flow relationships. The implemented GCN uses two hidden graph convolution layers followed by a classification layer for binary fraud prediction.

Dropout with rate 0.5 is applied after each layer to reduce overfitting.

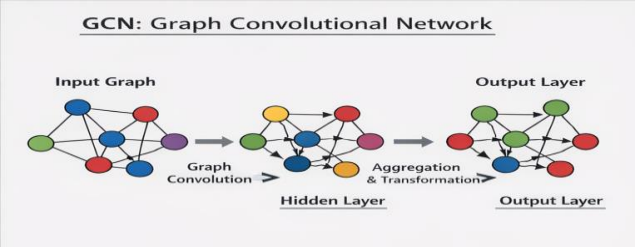

Fig -2: Graph Convolutional Network Architecture

G. GraphSAGE

GraphSAGE improves scalability by learning neighborhood aggregation functions through sampling instead of processing the full adjacency structure at every step. This makes it computationally efficient for large blockchain graphs. Its inductive learning capability allows the model to generalize to unseen transactions during inference, making it highly suitable for live transaction monitoring.

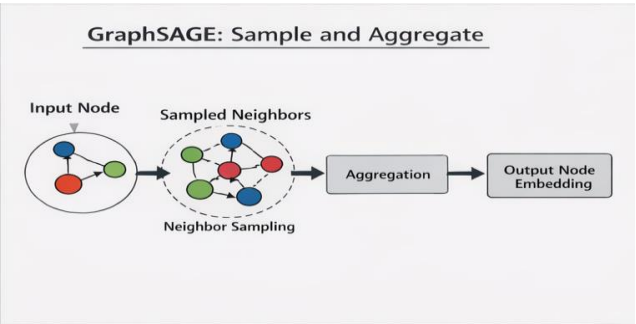

Fig -3: GraphSAGE Architecture

H. Graph Attention Network (GAT)

Graph Attention Network (GAT) introduces an attention mechanism that assigns different importance to neighboring transactions during message aggregation. Instead of treating all neighbors equally, the model

learns which connections are more relevant for fraud detection. This selective weighting helps identify suspicious relationships more effectively, although it increases computational complexity. The attention mechanism enables the model to focus on suspicious transaction relationships more effectively than uniform aggregation.

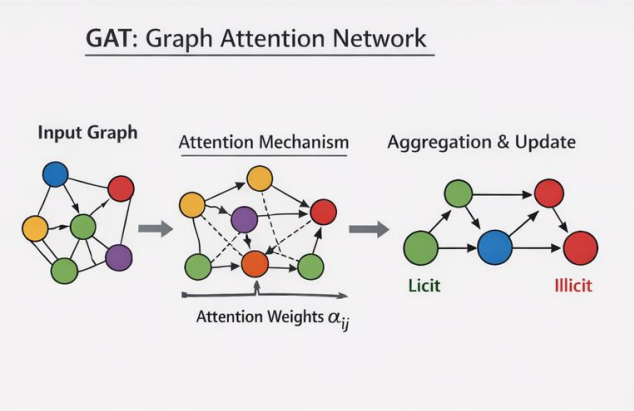

Fig -4: Graph Attention Network Architecture

I. Temporal Graph Neural Network

Fraud behavior often evolves over time, making temporal modeling important for blockchain analysis. The Temporal Graph Neural Network combines graph-based spatial learning with temporal sequence modeling to capture changes in transaction behavior across time steps. This enables detection of fraud patterns that emerge gradually over sequences of related transactions. The Temporal GNN processes transaction embeddings across 49 time steps. Each time step corresponds to approximately two weeks of Bitcoin activity.

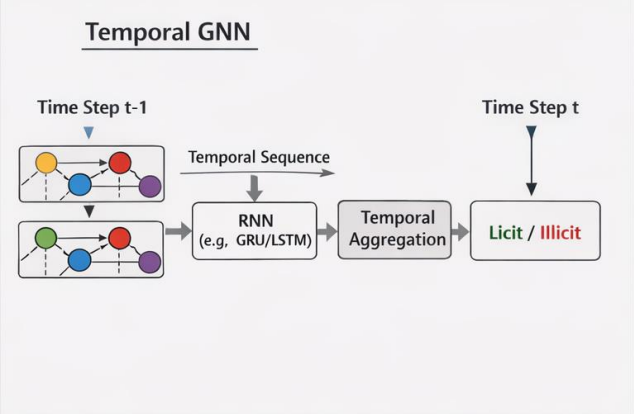

Fig -5: Temporal Graph Neural Network

As illustrated, each GNN architecture captures different aspects of blockchain transaction behavior. GCN emphasizes structural aggregation, GraphSAGE improves scalability and inductive learning, GAT enhances selective neighborhood weighting, and Temporal GNN captures time-dependent fraud evolution.

J. Training Strategy

All four Graph Neural Network models were trained using the Adam optimizer with an initial learning rate of 0.001 and weight decay of 5×10^{-4} for L2 regularization. Adam was selected because of its adaptive learning capability, which improves convergence speed and training stability compared to standard stochastic gradient descent.

The parameter update rule of Adam is defined as:

$$mt = \beta 1\ mt{-}1 + (1 - \beta 1)\ gt$$

$$vt = \beta 2\ vt{-}1 + (1 - \beta 2)\ gt^2$$

$$\theta t = \theta t{-}1 - \eta\ \hat{mt} / (\sqrt{\hat{vt}} + \varepsilon)$$

where: gt = gradient at step t; mt = first moment estimate; vt = second moment estimate; η = learning rate.

A ReduceLROnPlateau learning-rate scheduler was used to reduce the learning rate whenever validation loss stopped improving. The scheduler reduces the learning rate by a factor of 0.5 after 20 epochs without significant improvement.

Class weighting was applied to address severe imbalance between licit and illicit transactions. Misclassification of illicit transactions contributes more strongly to the loss, improving minority-class detection performance. Each model was trained for a maximum of 300 epochs. Early stopping was configured with patience 50, meaning training would stop if validation loss failed to improve for 50 consecutive epochs.

Training used full-batch graph processing, where the complete Elliptic graph is passed through the model in each epoch. Although loss is computed only on labeled training nodes, all nodes participate in message passing. This enables semi-supervised learning, allowing unlabeled transactions to contribute structural information to neighboring labeled nodes.

K. Inference Pipeline

After training, the model checkpoints and fitted StandardScaler are saved for deployment. During inference, a new transaction undergoes the same preprocessing pipeline used during training:

- Raw transaction features are collected
- Missing values are handled

- Features are standardized using saved scaler
- Transaction is converted into graph format
- Model performs forward propagation
- Fraud probability is computed

Fraud probability is defined as:

$$Pfraud = P(y = 1 \mid x) \quad (10)$$

When all four models are used simultaneously, their outputs are combined using majority voting. The ensemble decision is:

$$Final = Fraud\ if\ votes \geq 2,\ else\ Normal \quad (11)$$

A transaction is classified as fraudulent if at least two of the four models predict the illicit class.

5. SYSTEM ARCHITECTURE

The complete fraud detection system is organized into two major layers: a backend processing layer and a frontend visualization layer. The backend is responsible for data preprocessing, graph construction, model training, inference, and evaluation, while the frontend provides an interactive dashboard for transaction analysis and fraud monitoring.

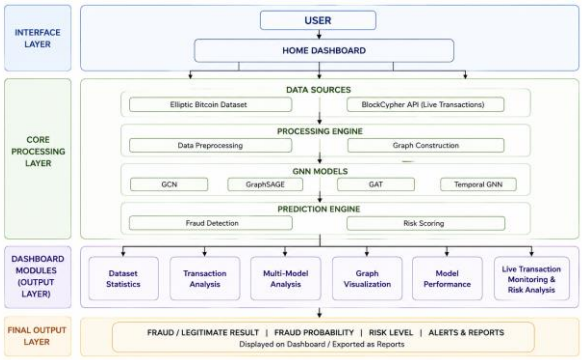

Fig -6: Proposed GNN-Based Fraud Detection System Architecture

A. Backend Architecture

The backend consists of several modular components:

- Data Loader Module
- Preprocessing Module
- Graph Construction Module
- Model Training Module
- Evaluation Module
- Predictor Module

The Data Loader reads the three raw Elliptic dataset files and merges them into a unified data structure containing features, graph edges, and class labels.

The Preprocessing Module performs missing-value handling, feature normalization, and label conversion. It ensures consistent input representation before graph generation.

The Graph Construction Module converts the preprocessed transaction data into a graph structure compatible with PyTorch Geometric. It builds node feature matrices and edge index tensors required for GNN operations.

The Model Training Module contains implementations of GCN, GraphSAGE, GAT, and Temporal GNN. It handles optimizer initialization, training loops, validation, checkpoint saving, and early stopping.

The Evaluation Module computes classification metrics such as Accuracy, Precision, Recall, F1-score, and confusion matrices for performance assessment.

B. Predictor Module

The Predictor Module serves as the bridge between offline model training and live dashboard inference. During application startup, the module loads:

- Saved model checkpoints
- Trained StandardScaler
- Model configuration parameters

When a transaction is submitted through the dashboard, the predictor applies the saved preprocessing pipeline and performs a forward pass through the selected GNN model. The output includes:

- Predicted class label
- Fraud probability
- Confidence score

When all four models are enabled, predictions are aggregated using majority voting to generate the final fraud decision.

C. Frontend Dashboard

The frontend is implemented using Streamlit, providing a user-friendly interface for fraud analysis and model visualization. The dashboard consists of multiple interactive pages designed for different analysis tasks.

D. Dashboard Modules

The major dashboard modules are:

- Home Page
- Dataset and Statistics
- Transaction Analysis
- Multi-Model Analysis
- Graph Visualization
- Live Transaction Monitoring
- Model Performance

Home Page: Displays project overview, system description, and navigation cards.

Fig -7: Dashboard Interface

Dataset and Statistics: Shows structural properties of the Elliptic dataset, including node count, edge count, class distribution, and feature statistics.

Transaction Analysis: Allows users to analyze individual transactions by entering transaction properties such as amount, number of inputs/outputs, and time step.

Multi-Model Analysis: Runs all four models simultaneously and compares their predictions using majority voting.

Graph Visualization: Displays transaction neighborhoods as interactive graphs, helping users visually inspect suspicious connections.

Live Transaction Monitoring: The Live Transaction Monitoring module enables real-time fraud analysis of Bitcoin transactions using live blockchain data fetched through the BlockCypher API. This module extends the offline fraud detection framework into a practical real-world monitoring system.

The dashboard continuously retrieves transactions from the Bitcoin mempool and extracts transaction-level attributes such as transaction ID, transferred amount, number of inputs, number of outputs, transaction fee, and timestamp. These features are transformed into approximate model-compatible feature vectors and passed to the trained GNN prediction pipeline for inference.

Users can configure three monitoring parameters:

- Number of transactions to fetch
- Delay between transaction processing
- Fraud probability threshold

After monitoring starts, incoming transactions are displayed in a live transaction stream. Each transaction is analyzed individually and classified as either legitimate or fraudulent. Legitimate transactions are highlighted in green, while suspicious transactions are highlighted in red. The dashboard also displays fraud probability scores for each transaction. For high-risk transactions, the system generates real-time fraud alerts containing transaction details and investigation flags. These alerts help identify suspicious blockchain activity immediately.

A session summary panel provides aggregate statistics including:

- Total transactions analyzed
- Legitimate transactions
- Suspicious transactions
- Fraud flagged transactions

For the most recent suspicious transaction, predictions from all four GNN models (GCN, GraphSAGE, GAT, and Temporal GNN) are displayed simultaneously. The final fraud decision is obtained through majority voting, allowing comparison of model agreement and confidence.

The module also maintains a complete transaction log containing transaction ID, amount, inputs, outputs, fees, prediction result, fraud probability, and data source. This real-time monitoring framework demonstrates the practical deployment capability of Graph Neural Networks for live blockchain fraud analytics.

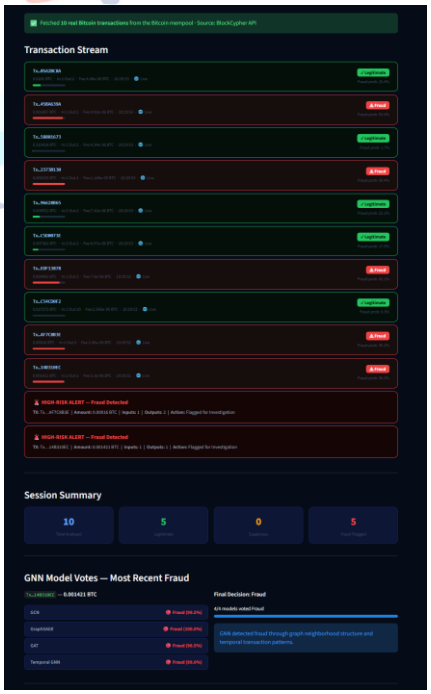

Fig -8: Real-Time Bitcoin Transaction Monitoring and Fraud Detection Using BlockCypher API

Model Performance: Displays training curves, confusion matrices, and evaluation metrics for all trained models.

E. System Workflow

The complete system workflow consists of five stages:

- Raw transaction data ingestion
- Preprocessing and normalization
- Graph construction
- GNN-based prediction
- Result visualization and analysis

This architecture ensures smooth movement of data from raw blockchain transactions to final fraud classification outputs. The modular design also improves maintainability, scalability, and deployment flexibility.

6. DATASET DESCRIPTION

This study uses the Elliptic Bitcoin dataset, a widely recognized benchmark dataset for blockchain fraud detection research. The dataset was compiled by Elliptic, a cryptocurrency analytics company specializing in anti-money laundering and fraud investigation solutions.

The dataset represents a snapshot of the Bitcoin transaction network, where transactions are represented as nodes and fund transfers between transactions are represented as directed edges. It contains both labeled and unlabeled transactions, making it highly suitable for semi-supervised learning. The dataset consists of:

- 203,769 transaction nodes
- 234,355 directed edges
- 49 temporal steps

Each time step corresponds to approximately two weeks of Bitcoin transaction activity. This temporal component makes the dataset suitable for dynamic graph learning and temporal fraud detection. Out of the total 203,769 transactions, only 46,564 transactions are labeled:

- 4,545 illicit transactions
- 42,019 licit transactions

The remaining 157,205 transactions are unlabeled. Illicit transactions include activities associated with:

- Money laundering
- Ransomware payments
- Ponzi schemes
- Malware services
- Darknet markets

Licit transactions correspond to legitimate entities such as cryptocurrency exchanges, mining pools, wallet providers, and payment processors. Each transaction node contains 166 numerical features. These features are divided into two categories:

- 94 Local Features
- 72 Aggregated Features

Local features describe the transaction itself, including transaction amount, number of inputs and outputs, timestamps, and statistical properties of transferred values.

Aggregated features summarize properties of neighboring transactions in the graph. These include mean, standard deviation, and correlation statistics computed from one-hop forward and backward neighborhoods.

The combination of local and aggregated features provides both transaction-level and neighborhood-level contextual information, which is essential for graph-based fraud detection. A major challenge in this dataset is severe class imbalance. Illicit transactions account for only approximately 9.7% of labeled samples. This imbalance makes fraud detection difficult because machine learning models tend to become biased toward the majority licit class.

Table -1: Elliptic Bitcoin Dataset Statistics

Dataset Property	Value
Total Transactions (Nodes)	203,769
Total Edges (Fund Flows)	234,355
Time Steps	49
Labeled Transactions	46,564
Illicit (Class 1)	4,545
Licit (Class 0)	42,019
Unlabeled	157,205
Features per Node	166
Local Features	94
Aggregated Features	72

Fig -9: Class Distribution in Elliptic Dataset

7. EXPERIMENTAL SETUP

The experimental setup was designed to evaluate the performance of the proposed Graph Neural Network based fraud detection framework under consistent and reproducible conditions. All experiments were implemented in Python using PyTorch and PyTorch Geometric for graph neural network operations.

A. Software Environment

The primary libraries used in implementation include:

- Python
- PyTorch
- PyTorch Geometric
- NumPy
- Pandas
- Scikit-learn
- Streamlit
- Plotly

PyTorch Geometric was used for graph construction and GNN layer implementations. Scikit-learn was used for feature normalization, class weight computation, and evaluation metrics. Streamlit and Plotly were used for developing the interactive fraud detection dashboard.

B. Hardware Environment

All experiments were conducted on a CPU-based computing environment. Despite the large size of the Elliptic graph, full-batch graph training was feasible without GPU acceleration due to efficient sparse graph processing in PyTorch Geometric. Each model completed training within practical time limits, with training durations typically ranging from several minutes to under one hour depending on model complexity.

C. Model Configuration

Each GNN architecture follows a two-layer graph learning design followed by a fully connected classification layer. The general configuration is:

- Input Dimension: 166
- Hidden Layer 1: 64
- Hidden Layer 2: 32
- Output Classes: 2

Dropout regularization with rate 0.5 was applied after graph convolution layers to reduce overfitting.

Table -2: Training Hyperparameters

Parameter	Value
Optimizer	Adam
Learning Rate	0.001
Weight Decay	5×10^{-4}
Max Epochs	300
Early Stopping Patience	50
LR Scheduler Patience	20
LR Reduction Factor	0.5
Dropout	0.5
Train Split	70%
Validation Split	15%
Test Split	15%
Loss Function	Weighted NLL Loss

D. Evaluation Metrics

Model performance was evaluated using four standard classification metrics: Accuracy, Precision, Recall, and F1-Score. These metrics were chosen because fraud detection involves severe class imbalance, making accuracy alone insufficient for reliable evaluation.

Accuracy is defined as:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

Precision is defined as:

$$Precision = TP / (TP + FP)$$

Recall is defined as:

$$Recall = TP / (TP + FN)$$

F1-score is defined as:

$$F1 = 2 \times (Precision \times Recall) / (Precision + Recall)$$

Table -3: Evaluation Metrics

Metric	Formula	Interpretation
Precision	TP / (TP + FP)	Correct fraud predictions
Recall	TP / (TP + FN)	Detected actual frauds
F1-Score	2PR / (P + R)	Precision-recall balance
Accuracy	(TP + TN) / Total	Overall correctness

Precision is important because false fraud alerts increase manual investigation cost. Recall is critical because missed fraudulent transactions represent detection failures. Due to the class imbalance in the Elliptic dataset, F1-score was considered the primary evaluation metric for model comparison, as it provides a balanced measure of fraud detection performance.

8. RESULTS AND DISCUSSION

All four Graph Neural Network models were evaluated on the same held-out test partition containing 6,986 labeled transactions. Performance evaluation was carried out using the live evaluation pipeline integrated into the deployed dashboard, ensuring that the reported metrics were computed directly from the saved model checkpoints rather than from manually stored result files. The evaluated models include GCN, GraphSAGE, GAT, and Temporal GNN. Performance was measured using Accuracy, Precision, Recall, and F1-score.

Table -4: Model Comparison on Elliptic Test Partition (6,986 Transactions)

Model	Accuracy	Precision	Recall	F1-Score
GCN	89.26%	48.66%	89.09%	62.94%
GraphSAGE	97.19%	80.49%	95.80%	87.48%
GAT	78.64%	32.01%	96.64%	48.09%
Temporal GNN	92.24%	57.41%	93.71%	71.20%

The experimental results reveal clear performance differences among the evaluated Graph Neural Network architectures despite being trained under identical preprocessing, optimization, and evaluation settings. Among all models, GraphSAGE achieved the best overall performance, attaining an accuracy of 97.19% and an F1-score of 87.48%, while maintaining an effective balance between precision and recall. This demonstrates that GraphSAGE is particularly well suited for blockchain fraud detection, where minimizing both missed fraudulent transactions and unnecessary false alarms is essential.

The superior performance of GraphSAGE can be attributed to its inductive neighborhood aggregation mechanism, which efficiently captures structural relationships while maintaining good generalization to unseen transaction patterns. Unlike Graph Convolutional Networks that rely primarily on fixed graph convolution operations, GraphSAGE learns neighborhood aggregation functions that improve scalability and representation quality, making it more suitable for large and evolving blockchain transaction networks.

Although the Graph Attention Network achieved the highest recall, its considerably lower precision indicates that the attention mechanism aggressively classified legitimate transactions as fraudulent, resulting in a large number of false-positive predictions. Similarly, the Temporal Graph Neural Network outperformed the baseline GCN by incorporating temporal information, demonstrating that transaction evolution provides useful additional information for fraud detection. However, its overall performance remained lower than that of GraphSAGE, indicating that stable neighborhood aggregation was more beneficial than increased architectural complexity for the Elliptic Bitcoin dataset.

Overall, the comparative evaluation demonstrates that increased model complexity does not necessarily translate into superior fraud detection performance. Instead, carefully designed neighborhood aggregation strategies combined with fair experimental evaluation play a more significant role in achieving reliable blockchain fraud detection.

A. Confusion Matrix Analysis

To better understand model behavior beyond overall accuracy and F1-score, confusion matrix analysis was performed for all four models. The confusion matrix provides detailed information about false positives and false negatives, which is especially important in fraud detection. The confusion matrix consists of four components:

- True Positive (TP): Fraudulent transactions correctly classified as fraud
- True Negative (TN): Legitimate transactions correctly classified as normal
- False Positive (FP): Legitimate transactions incorrectly classified as fraud
- False Negative (FN): Fraudulent transactions incorrectly classified as normal

In fraud detection systems, minimizing false negatives is critical because missed fraud directly impacts security. However, excessive false positives also reduce practicality by generating unnecessary alerts for investigation.

Table -5: Confusion Matrix Counts on Test Partition

Model	TN	FP	FN	TP
GCN	5599	672	78	637
GraphSAGE	6105	166	30	685
GAT	4803	1468	24	691
Temporal GNN	5774	497	45	670

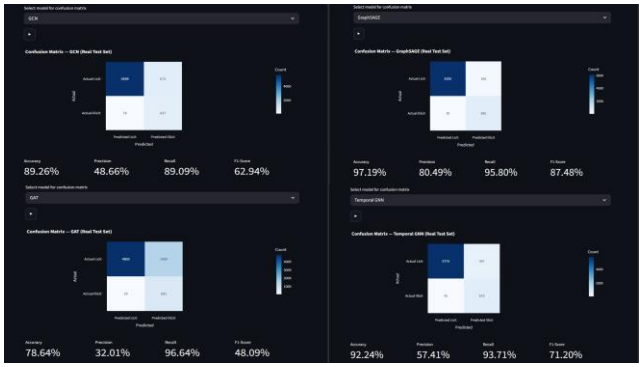

Fig -10: Confusion Matrices of All Four Models

GraphSAGE produced the most balanced confusion matrix among all models. It correctly classified 6,105 legitimate transactions and 685 fraudulent transactions, while generating only 166 false positives and 30 false

negatives. This demonstrates strong fraud detection capability with minimal unnecessary alerts. The GCN baseline showed high fraud detection sensitivity but generated more false positives than GraphSAGE. It misclassified 672 legitimate transactions as fraudulent, which reduced precision despite maintaining strong recall.

The Graph Attention Network exhibited the most aggressive fraud prediction behavior. It correctly detected 691 fraudulent transactions and missed only 24 fraud cases, resulting in the highest recall of all models. However, it incorrectly classified 1,468 legitimate transactions as fraudulent, causing severe precision degradation. The Temporal Graph Neural Network improved significantly over the GCN baseline by reducing false positives from 672 to 497 and false negatives from 78 to 45. This confirms that incorporating temporal information helps improve discrimination between legitimate and fraudulent transaction behavior.

These results reveal an important trade-off between recall and precision. Models such as GAT prioritize fraud detection sensitivity and minimize missed fraud cases, but at the cost of excessive false alarms. GraphSAGE achieves the most practical balance between fraud detection capability and operational efficiency, making it the most suitable model for real-world deployment.

B. Training Summary

Training history was recorded for all four models, including training loss, validation loss, and validation accuracy at each epoch. These metrics provide insights into convergence behavior and generalization performance.

Table -6: Training Summary for Each Model

Model	Epochs	Final Train Loss	Best Val Loss	Best Val Acc
GCN	300	0.3039	0.2637	0.8956
GraphSAGE	300	0.1108	0.1094	0.9635
GAT	300	0.4466	0.3907	0.7828
Temporal GNN	300	0.1975	0.2014	0.9167

GraphSAGE achieved the lowest training loss (0.1108) and the best validation loss (0.1094), indicating highly stable convergence and strong generalization. The close alignment between training and validation curves suggests minimal overfitting. The Graph Attention Network showed the weakest training behavior, with the highest final training loss (0.4466) and lowest validation accuracy (78.28%). Its validation curve plateaued relatively early, suggesting difficulty in learning discriminative patterns from the limited labeled dataset.

Two major factors explain the relatively poor performance of GAT. First, the multi-head attention mechanism significantly increased model complexity. The first GAT layer used eight attention heads, expanding the intermediate representation to 512 dimensions. This introduced substantially more trainable parameters compared to GCN and GraphSAGE.

Second, the severe class imbalance interacted strongly with the attention mechanism. Since illicit transactions form a small minority, the class-weighted loss pushed the model toward aggressively predicting fraud to avoid missing minority samples. As a result, GAT achieved extremely high recall but at the cost of excessive false positives. GraphSAGE outperformed all other models primarily because of its stable mean-aggregation mechanism and inductive learning capability. Unlike GAT, it avoids excessive parameter growth while still effectively capturing neighborhood information. Its aggregation strategy produces robust node embeddings that generalize well even under severe class imbalance.

The Temporal Graph Neural Network performed better than the static GCN baseline, improving F1-score from 62.94% to 71.20%. This confirms that temporal information provides meaningful additional signal for fraud detection. Fraudulent activities often evolve across time, and temporal modeling helps capture such behavior patterns.

The semi-supervised training strategy also contributed significantly to performance. Although only labeled transactions were used for loss computation, unlabeled nodes participated fully in graph message passing. Since more than 75% of nodes are unlabeled, this additional structural information helped all models learn richer transaction representations.

Despite the encouraging performance achieved by the evaluated models, several limitations should be acknowledged. The reported results were obtained using a single train-validation-test partition, and therefore slight variations in performance may occur under different random data splits or cross-validation strategies. Furthermore, the live transaction monitoring module relies on approximate feature extraction from publicly available blockchain data rather than the

complete aggregated neighborhood features provided in the Elliptic dataset. Consequently, predictions generated during live monitoring should be interpreted as indicative assessments rather than direct equivalents of the offline evaluation results.

Nevertheless, the experimental findings consistently demonstrate the effectiveness of Graph Neural Networks for blockchain fraud detection. The unified comparative evaluation further reveals that architectural complexity alone does not guarantee superior performance. Instead, GraphSAGE achieved the best overall balance between accuracy, precision, recall, and computational efficiency, making it the most suitable architecture among the evaluated models for practical blockchain fraud detection applications.

9. ANALYSIS

The experimental evaluation confirms that Graph Neural Networks are highly effective for blockchain fraud detection because they exploit both transaction attributes and graph structural relationships that conventional machine learning models cannot fully capture. By representing blockchain transactions as interconnected graph nodes, the evaluated models effectively learn neighborhood information that improves the identification of suspicious transaction patterns.

A key outcome of this study is the systematic comparison of four complementary Graph Neural Network architectures under identical preprocessing, training, and evaluation conditions. Unlike many previous studies that investigate a single GNN architecture, this comparative evaluation provides a clearer understanding of the strengths and limitations of different graph learning approaches for blockchain fraud detection. The results demonstrate that fair experimental comparison is essential for selecting an appropriate model for practical deployment.

Among the evaluated models, GraphSAGE consistently achieved the best overall performance by providing an effective balance between accuracy, precision, recall, and F1-score. Its inductive neighborhood aggregation strategy enables efficient learning from large blockchain transaction networks while maintaining strong generalization to previously unseen transactions. These characteristics make GraphSAGE particularly suitable for real-world fraud detection environments where new transactions continuously appear.

Although Graph Attention Network achieved the highest recall, its substantially lower precision resulted in a large number of false-positive predictions, increasing the potential cost of manual investigation. Similarly, the Temporal Graph Neural Network demonstrated that incorporating temporal information improves fraud detection compared with the baseline GCN model, although the additional architectural complexity did not outperform GraphSAGE on the Elliptic Bitcoin dataset.

Overall, the study demonstrates that effective blockchain fraud detection depends not only on model complexity but also on stable neighborhood aggregation, balanced learning strategies, and fair comparative evaluation. The proposed framework therefore provides both a comprehensive experimental comparison and a practical implementation that can support future research and real-world blockchain fraud analysis.

10. CONCLUSION

This paper presented a Graph Neural Network-based framework for detecting fraudulent blockchain transactions using the Elliptic Bitcoin dataset. By representing blockchain transactions as graph-structured data, the proposed framework effectively captured structural relationships that are difficult for conventional machine learning methods to model. Four complementary Graph Neural Network architectures, namely Graph Convolutional Network (GCN), GraphSAGE, Graph Attention Network (GAT), and Temporal Graph Neural Network (Temporal GNN), were implemented and evaluated under identical preprocessing, training, and evaluation conditions to ensure a fair and systematic comparison.

Experimental evaluation demonstrated that GraphSAGE achieved the best overall performance, attaining 97.19% accuracy and an F1-score of 87.48%, while providing the most effective balance between precision and recall for blockchain fraud detection. The comparative analysis further showed that increased architectural complexity does not necessarily result in superior detection performance, highlighting the importance of efficient neighborhood aggregation and balanced representation learning.

In addition to comparative model evaluation, the proposed framework was successfully integrated into an interactive Streamlit-based fraud detection dashboard

supporting transaction analysis, graph visualization, multi-model prediction, and live transaction monitoring. The study demonstrates that systematic comparative evaluation of multiple Graph Neural Network architectures provides valuable insights into their practical suitability for blockchain fraud detection, while offering a deployable framework that can support future research and real-world blockchain fraud analysis.

11. FUTURE SCOPE

Several directions can further enhance the proposed blockchain fraud detection framework. First, advanced imbalance-handling techniques such as focal loss, adaptive threshold optimization, and cost-sensitive learning can be investigated to further improve minority-class detection while reducing false-positive predictions. Second, future work can explore heterogeneous Graph Neural Networks and Graph Transformer architectures to model more complex blockchain relationships involving wallet addresses, exchanges, mining pools, and smart contracts.

The real-time monitoring framework can also be extended by constructing dynamic transaction graphs from continuously incoming blockchain data, enabling the computation of richer neighborhood features instead of relying on approximate feature representations. Furthermore, incorporating Explainable Artificial Intelligence (XAI) techniques such as GNNExplainer or attention-based interpretation methods would improve the transparency and interpretability of fraud predictions, thereby increasing user confidence in practical financial applications.

Finally, the proposed framework can be evaluated on additional cryptocurrency datasets and blockchain platforms such as Ethereum to assess its scalability, robustness, and generalization across different blockchain environments. These extensions would contribute toward the development of more accurate, interpretable, and deployable graph-based fraud detection systems.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in Proc. ICLR, 2017.

[2] M. Weber et al., "Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics," arXiv:1908.02591, 2019.

[3] P. Velickovic et al., "Graph Attention Networks," in Proc. ICLR, 2018.

[4] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in Proc. NeurIPS, 2017.

[5] A. Pareja et al., "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs," in Proc. AAAI, 2020.

[6] Y. Liu et al., "Blockchain Fraud Detection Using Graph Neural Networks," IEEE Transactions, 2020.

[7] A. Asiri and K. Somasundaram, "Graph Convolution Network for Fraud Detection in Bitcoin Transactions," Scientific Reports, vol. 15, no. 11076, 2025.

[8] E. Rossi et al., "Temporal Graph Networks for Deep Learning on Dynamic Graphs," arXiv:2006.10637, 2020.

[9] S. Cai et al., "Cash-Out User Detection Based on Attributed Heterogeneous Information Network with a Hierarchical Attention Mechanism," in Proc. AAAI, 2023.

[10] X. Zhang et al., "Temporal-Aware Graph Neural Network for Cryptocurrency Fraud Detection," IEEE Trans. Neural Netw. Learn. Syst., 2024.

[11] X. Chen, Y. Li, and X. Wang, "Ensemble Graph Neural Networks for Fraud Detection," in Proc. IEEE ICDM, 2021.

[12] Y. Zhang, J. Chen, and Q. Liu, "Variational Graph Autoencoders for Fraud Detection in Bitcoin Transactions," in Proc. IEEE Big Data, 2021.

[13] W. Wang, S. Zhou, and Z. Liu, "Heterogeneous Graph Neural Networks for Financial Fraud Detection," in Proc. ACM SIGKDD, 2019.