Expert Systems With Applications 299 (2026) 130121

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A dual-layer GNN with economic penalty mechanisms for blockchain fraud detection

Grace Mupoyi Ntuala [a], Qi Xia [a], Hu Xia [a], Ansu Badjie [a], Patrick Mukala [b], Edson Eliezer Da Silva Tavares [a], Jianbin Gao [a,*], Chiagoziem C. Ukwuoma [c]

[a] University of Electronic Science and Technology of China, 2006 Xiyuan Avenue, Chengdu High-tech Zone (West District), 611731, Chengdu, China
[b] School of Computer Science, University of Wollongong, 20183, Dubai, United Arab Emirates
[c] School of International Education, Chengdu University of Technology, Oxford Brookes College, Sichuan PR, 610059, China

ARTICLE INFO

Keywords:
Blockchain fraud detection
Graph neural networks
Economic penalties
Ethereum
Machine learning
Network security

ABSTRACT

Blockchain fraud detection faces critical challenges with an estimated $12.03 billion in potential losses from fraudulent transactions in cryptocurrency networks. Traditional rule-based and static machine learning approaches fail to capture the complex, evolving relationships in blockchain transaction networks. In this paper, we present HTFD (Hybrid Transactional Fraud Detection), a novel framework that integrates Graph Neural Networks (GNNs) with an economic penalty mechanism for both fraud detection and mitigation. HTFD introduces several key innovations: A dual-layer GNN architecture (GCN + GAT) that jointly exploits structural learning and attention-based neighbor weighting to uncover hidden fraudulent transaction patterns; An economic penalty mechanism that dynamically suppresses the impact of fraudulent nodes by removing suspicious transactions, enhancing security beyond detection; A scalable, real-world validation on three Ethereum stablecoin datasets, achieving superior AUC (96.02%) with balanced precision-recall while maintaining practical throughput and latency; and a practical fraud defense paradigm that demonstrates real-time applicability, offering a balance between detection accuracy, scalability, and network decentralization. Results validate HTFD as a novel and effective blockchain fraud detection framework that not only identifies fraudulent behaviors but also mitigates their impact, making it suitable for securing high-throughput decentralized financial networks.

1. Introduction

Blockchain technology has revolutionized digital transactions by providing secure, transparent, and decentralized solutions across multiple industries, from finance and supply chain management (Gao et al., 2022) to healthcare (Amofa et al., 2024) and beyond (Ahmad et al., 2021; Li et al., 2022; Tripathi et al., 2023; Zhang et al., 2021). By leveraging advanced cryptographic methods and distributed consensus mechanisms (Zou et al., 2024), blockchain networks have enabled unprecedented levels of trust and transparency in digital ecosystems. However, this rapid adoption has also attracted sophisticated fraudulent schemes that exploit the very features that make blockchain attractive: decentralization, pseudonymity, and transaction immutability.

The scale of blockchain fraud represents a critical threat to the entire cryptocurrency ecosystem. Initial Coin Offering (ICO) investors alone face an estimated loss of $12.03 billion due to fraudulent transactions and scams (Phua, 2022), while sophisticated attack vectors including multi-level marketing schemes (Far et al., 2023), wash trading (Morgia et al., 2023; Victor & Weintraud, 2021), and phishing attacks (Liu et al., 2024b; Sun et al., 2024) continue to evolve and proliferate. These fraudulent activities not only cause direct financial losses but also undermine user trust and hinder the broader adoption of blockchain technology.

Traditional fraud detection approaches face fundamental limitations when applied to blockchain environments. Rule-based systems (Kim et al., 2023; Sakurai & Shudo, 2024) and static machine learning models (Kirkland et al., 2024; Li et al., 2024; Sakurai & Shudo, 2024) cannot adequately capture the dynamic, graph-structured nature of blockchain transactions, where nodes represent accounts and edges represent transaction flows. The pseudonymous nature of blockchain users and the irreversibility of transactions create additional challenges that existing detection methods struggle to address. Moreover, fraudsters continuously adapt their strategies, requiring detection systems that can evolve and learn from emerging attack patterns in real-time.

Blockchain transactions inherently form complex, high-dimensional graphs with intricate relationships between entities. This graph

* Corresponding author.
E-mail address: gaojb@uestc.edu.cn (J. Gao).

https://doi.org/10.1016/j.eswa.2025.130121
Received 18 August 2025; Received in revised form 10 October 2025; Accepted 19 October 2025
Available online 23 October 2025

structure contains rich information about transaction patterns, user behaviors, and network topology that traditional machine learning approaches fail to exploit effectively. Recent advances in Graph Neural Networks (GNNs) have demonstrated remarkable success in analyzing graph-structured data across various domains (Cai et al., 2024; Huang et al., 2024; Li et al., 2021), presenting an opportunity to develop more sophisticated fraud detection mechanisms specifically designed for blockchain environments.

To address these challenges, this paper presents HTFD (Hybrid Transactional Fraud Detection), a novel framework that integrates advanced Graph Neural Networks with an innovative economic penalty mechanism for comprehensive fraud detection and mitigation in blockchain networks. The unique contributions of this paper are as follows:

- Novel dual-layer GNN architecture: We design the first blockchain fraud detection model that combines GCN and GAT, enabling simultaneous structural pattern learning and attention-based feature selection.
- Integrated fraud mitigation mechanism: We introduce an innovative economic penalty mechanism that dynamically isolates fraudulent nodes and transactions, extending detection into active mitigation–a capability absent in prior GNN-based fraud detectors.
- Extensive real-world validation: We evaluate HTFD on three large-scale Ethereum stablecoin networks, demonstrating consistent improvements in AUC, precision, recall, and F1 over both classical ML and graph-based baselines.
- Scalability and real-time readiness: We analyze computational complexity, throughput, and latency, showing that HTFD scales to real-world blockchain workloads while maintaining operational efficiency.

The remainder of this paper is organized as follows: Section 2 reviews related work in blockchain fraud detection and graph neural networks. Section 3 provides the necessary preliminaries on blockchain modeling and centrality measures. Section 4 details our proposed HTFD framework and its components. Section 5 presents comprehensive experimental results and performance analysis. Finally, Section 6 concludes the paper and discusses future research directions. To our knowledge, HTFD is the first framework to integrate dual-layer GNNs with an economic penalty mechanism for blockchain fraud detection and mitigation.

2. Related works

This section reviews existing approaches to blockchain fraud detection and graph neural networks. The purpose is to position our work within the literature and highlight the research gaps that HTFD addresses. Fraud detection in blockchain networks has been extensively studied, but existing approaches often fall short in addressing the unique challenges posed by blockchain's decentralized and pseudonymous nature. Below, we discuss related works and highlight how our proposed HTFD framework addresses their limitations.

2.1. Anomaly detection in general networks

Li et al. (2007) proposed methods for anomaly detection in general network environments. However, their approach requires clean datasets for training, which are rarely available in blockchain networks due to noisy, unlabeled, and high-dimensional data. Additionally, their focus on generic anomaly detection lacks specificity for blockchain systems. In contrast, HTFD leverages GNNs, which are designed to operate effectively on noisy, unlabeled data and analyze the graph-structured data inherent to blockchain systems. This enables precise detection of fraudulent activities by capturing intricate relationships between nodes and transactions.

2.2. Lightweight anomaly detection for IoT

Breitenbacher et al. (2019) developed a lightweight host-based anomaly detection system suitable for IoT devices. While effective in resource-constrained environments, their framework relies on static whitelists, making it inflexible for adapting to evolving attack patterns. Moreover, its design limits scalability for high-throughput blockchain networks. HTFD addresses these limitations by employing GNNs with continuous learning capabilities, enabling dynamic adaptation to new and evolving fraud patterns. Additionally, HTFD integrates economic penalties and node isolation strategies, ensuring scalable and real-time mitigation of fraudulent activities without compromising performance.

Existing methods, such as rule-based systems and static machine learning models, fail to capture the dynamic and evolving nature of blockchain networks. Blockchain transactions form complex, high-dimensional graphs where relationships between nodes constantly change. Static models cannot adapt to these changes in real-time, which makes them less effective in detecting emerging fraud patterns. Additionally, these models often struggle with the pseudonymous nature of blockchain users and the complexity of cross-network relationships. In contrast, HTFD leverages Graph Neural Networks (GNNs) to model and continuously learn from these dynamic relationships, providing a scalable, adaptive solution that adjusts as the network evolves. The economic penalty mechanism in HTFD further enhances the model's effectiveness by mitigating the impact of fraudulent activities without disrupting the blockchain's core functionalities. By incorporating both detection and mitigation, HTFD addresses the gaps left by previous models that solely focus on fraud detection.

2.3. Temporal anomaly detection

Andrade et al. (2020) proposed a model for suspicious process detection based on temporal anomaly detection. While effective in identifying concentrated suspicious activities within logs, their approach relies on temporal assumptions and log-based data, limiting its applicability to blockchain environments where transactions are often asynchronous. HTFD overcomes these limitations by using advanced graph-based models to capture the asynchronous and dynamic nature of blockchain transactions. By analyzing centrality features and inter-dependencies, HTFD achieves accurate fraud detection even when temporal patterns are insufficient.

2.4. Community detection in blockchain networks

Chen and Liu (2019) introduced a distributed community detection algorithm using structural entropy to identify community structures in blockchain networks. While their approach enhances anomaly detection and communication efficiency, it focuses solely on community detection and lacks direct fraud mitigation mechanisms. Furthermore, its static framework does not adapt to the evolving nature of blockchain fraud. HTFD integrates insights from community detection with actionable fraud detection and mitigation strategies. By combining GNNs with economic penalties, HTFD identifies fraudulent activities and reduces their impact on the network, dynamically adapting to evolving fraud patterns.

Recent advancements in GNN-based blockchain fraud detection have shown promising results. For instance, several works have leveraged GNNs to capture complex patterns in transaction graphs, such as using Graph Convolutional Networks (GCNs) to model blockchain interactions or Graph Attention Networks (GATs) to weigh transaction relevance. However, these approaches often fail to consider real-time fraud mitigation or adaptive learning for evolving fraud patterns. HTFD extends these approaches by integrating a dynamic economic penalty mechanism, which limits the impact of detected fraudulent activities, thus ensuring the detection system not only identifies fraud but also reduces its spread within the network. This makes HTFD a more holistic solution

Table 1
Feature-based comparison of HTFD and related methods.

Feature	HTFD (Ours)	(Crincoli et al., 2022)	HADES-IoT (Jung et al., 2019)	Yuan et al. (2021)	Breitenbacher et al. (2019)	CaT-GNN (Cheng et al., 2024)	GNN-CL (Duan et al., 2024)	GraphSAGE (Hamilton et al., 2017)	TGAT (Xu et al., 2020)
Graph-Based	✓	×	×	×	×	✓	✓	✓	✓
ML-Based	✓ GNN + LOF	✓ AE	× Rules	× Stats	× Formal	✓ GNN (causal temporal)	✓ GNN + CL + RL	✓ GNN + SAGE	✓ GAT
Real-Time	✓	×	✓	✓	×	×	×	✓	✓
Mitigation	✓	×	×	×	×	×	×	×	×
Adaptability	✓	×	×	×	×	×	✓	✓	✓

compared to recent GNN-based methods that focus solely on detection without an integrated mitigation strategy.

2.5. Detection of Ponzi schemes

Jung et al. (2019) provided a strong foundation for detecting Ponzi schemes using data mining and machine learning. Their approach achieves high precision through 0-day and behavior-based features but relies on static features and lacks graph-based analytics, limiting its ability to detect complex and evolving fraud schemes. HTFD addresses these gaps by integrating graph-based analysis with machine learning, capturing relational dynamics and behavioral patterns. Additionally, HTFD improves scalability and incorporates adaptive mechanisms, making it more robust for detecting sophisticated and evolving fraud in blockchain networks. We give a summary of our proposed model in Table 1 regarding other proposed models based on the metrics aspect of the model.

2.6. Graph neural network approaches for fraud detection

Recent advancements in Graph Neural Networks have shown significant promise for fraud detection across various domains. Hamilton et al. (2017) introduced GraphSAGE (Graph Sample and Aggregate), which generates node embeddings through neighborhood sampling and aggregation. While GraphSAGE demonstrates strong performance on large-scale graphs and provides inductive learning capabilities, it primarily focuses on structural pattern recognition without addressing the temporal dynamics inherent in blockchain transactions. Moreover, GraphSAGE lacks integrated fraud mitigation mechanisms, limiting its applicability for real-time fraud prevention in blockchain environments.

The scholars Xu et al. (2020) proposed TGAT (Temporal Graph Attention Network), extending graph attention mechanisms to dynamic graphs with temporal information. TGAT effectively captures time-evolving relationships between entities, making it relevant for blockchain fraud detection where transaction timing patterns are crucial. However, TGAT's temporal attention mechanism does not incorporate centrality-based structural importance measures, potentially missing critical network topology features that indicate fraudulent behavior. Additionally, TGAT focuses solely on detection without providing fraud mitigation strategies.

Duan et al. (2024) developed CaT-GNN (Causal Temporal Graph Neural Network) for credit card fraud detection, incorporating causal reasoning into GNNs to improve robustness and interpretability. CaT-GNN addresses the limitation of traditional correlation-based approaches by focusing on causal relationships between transactions. While this approach enhances model interpretability, it was specifically designed for credit card networks and lacks adaptation for blockchain-specific characteristics such as pseudonymous transactions and immutable ledgers. Furthermore, CaT-GNN does not provide real-time fraud mitigation capabilities.

Cheng et al. (2024) introduced GNN-CL, combining Graph Neural Networks with contrastive learning and reinforcement learning for financial fraud detection. GNN-CL integrates CNNs, LSTMs, and GNNs while using reinforcement learning to dynamically adjust node importance weights. Although this multi-modal approach demonstrates effectiveness in financial fraud scenarios, it suffers from increased computational complexity and lacks the economic penalty mechanisms necessary for proactive fraud mitigation in decentralized networks.

HTFD addresses the limitations of these existing approaches through several key innovations: (1) Unlike GraphSAGE's sampling-based approach, HTFD combines GCN and GAT layers to capture both local structural patterns and global attention-based relationships simultaneously. (2) While TGAT focuses on temporal attention, HTFD incorporates centrality-based structural importance measures (betweenness and closeness centrality) that provide complementary information about node influence and network positioning. (3) In contrast to CaT-GNN's causal reasoning approach, HTFD integrates an economic penalty mechanism that not only detects fraud but actively mitigates its impact through dynamic transaction removal and node isolation. (4) Unlike GNN-CL's multi-modal complexity, HTFD achieves superior performance through a streamlined dual-layer architecture while maintaining computational efficiency suitable for real-time blockchain applications.

3. Preliminaries

In this section, we provide the background on blockchain graph modeling and centrality measures. These preliminaries are essential for understanding how we extract features and build the proposed HTFD framework. In this paper, the blockchain network is represented as a directed graph, with nodes representing entities (e.g., users, addresses), edges representing transactions between these entities, the weight representing the value of the transaction, and the timestamp representing the time at which the transaction occurs between two entities. To analyze the structure and identify high-influence nodes, we use centrality measures as illustrated in Fig. 1.

3.1. Graph construction

We model the blockchain as a directed graph $G = (V, \vec{E}, W, T)$, where V represents nodes (blockchain addresses), $\vec{E}$ represents directed edges (transactions), W denotes transaction values as edge weights, and T represents transaction timestamps. This graph representation effectively captures the complex relationships and transaction flows within blockchain networks. Each transaction edge is labeled with both its value and its timestamp, which is later used in centrality analysis and in the economic penalty mechanism to capture temporal fraud patterns

3.2. Betweenness centrality

The centrality of the interconnection quantifies the frequency with which a node lies on the shortest paths that connect other nodes in a network. From Fig. 1, the betweenness centrality $C_B(A)$ for a node A is defined as follows:

$$C_B(A) = \sum_{s \neq A \neq t} \frac{\sigma(s,t|A)}{\sigma(s,t)} \tag{1}$$

Where $\sigma(s,t)$ is the total number of shortest routes from node s to node t, and $\sigma(s,t|A)$ is the number of paths that pass through A.

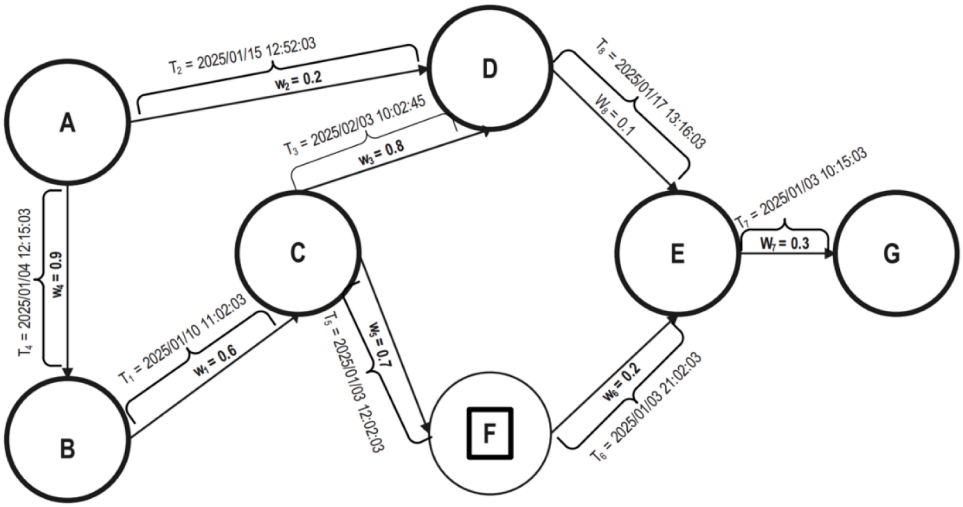

Fig. 1. Centrality measurement.

The betweenness centrality is important because it measures the frequency at which a node appears on the shortest path between other nodes. Nodes with high-betweenness centrality act as key intermediaries, controlling traffic flow in the network, making them valuable for fraud detection, since fraudulent transactions often pass through such nodes.

3.3. Closeness centrality

This measurement quantifies how close a node is to all other nodes in the network. A node with high closeness can quickly reach any other node in the network, making it essential for monitoring high-impact fraudulent actors that could affect the network's overall integrity. The closeness centrality, which is defined by $C_C(A)$, is calculated as:

$$C_C(A) = \frac{1}{\sum_B d(A,B)} \tag{2}$$

Where $d(A,B)$ is the shortest path distance between node A and node B. These centrality measures help identify influential nodes in the network, which is crucial to detect potentially fraudulent activities.

4. Proposed model

This section presents our HTFD framework in detail. We describe the feature extraction process, dual-layer GNN architecture, and economic penalty mechanism that together enable effective fraud detection and mitigation. This section delineates the key components required to implement our model and contextualizes the outcomes within our theoretical framework. The proposed model, illustrated in Fig. 2, encapsulates the comprehensive workflow from data ingestion to the final stage of economic penalty assessments, during which anomalies are identified.

4.1. Input feature extraction and fraud detection

We develop a feature extraction technique to detect anomalies in blockchain networks by analyzing their graph structure. The process begins by calculating the centrality measures for each node in the directed graph, specifically the centrality of the connection (Eq. (1)) and closeness centrality (Eq. (2)). These measures quantify the importance and influence of each address within the network topology.

4.1.1. Anomaly detection

The Local Outlier Factor (LOF) algorithm (Alghushairy et al., 2020) is used to analyze centrality features to differentiate between fraudulent and non-fraudulent node behavior. If a node is classified as fraudulent, the transactions associated with it are labeled as fraudulent (denoted by '1'). On the other hand, if a node is classified as non-fraudulent, its corresponding transactions are labeled as non-fraudulent (denoted by '0'). After this labeling process, a Graph Neural Network (GNN) model is trained on the labeled graph to classify transactions as either fraudulent or non-fraudulent.

4.2. Graph neural networks

Graph Neural Networks (GNNs) (Cai et al., 2024; Huang et al., 2024) are a powerful class of machine learning models designed to operate on graph-structured data. Graphs (Li et al., 2021), which consist of nodes (vertices) representing entities and edges denoting relationships (transactions), are a natural representation of complex systems such as social networks, transportation grids, biological pathways, and blockchain networks. Unlike traditional neural networks that operate on fixed-dimensional data like images or sequences, GNNs excel at capturing the relational and topological information inherent in graph data, making them a versatile tool for solving a wide range of problems. The core principle of GNNs lies in their ability to iteratively learn node and edge-level (transactions) representations through message passing or neighborhood aggregation.

Graph Convolutional Network (GCN). The GCN layer updates node features based on their neighbors by using the Eq. (3):

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right) \tag{3}$$

Where $H^{(l)}$ represents the node features at layer l, $\tilde{A}$ is the adjacency matrix with added self-loops, $\tilde{D}$ is the degree matrix of $\tilde{A}$, and $W^{(l)}$ are learnable weight parameters. σ denotes the activation function, such as ReLU.

Graph Attention Layer (GAT). The GAT layer computes the attention score for each neighboring node using Eq. (4):

$$e_{ij} = \text{LeakyReLU}\left(\mathbf{a}^T[W\mathbf{h}_i \| W\mathbf{h}_j]\right) \tag{4}$$

Where $\mathbf{h}_i$ and $\mathbf{h}_j$ are the feature vectors of node i and its neighbor j, $\mathbf{a}$ is the attention vector, W is a weight matrix, and $\|$ denotes concatenation.

The attention scores are normalized using a softmax function given by Eq. (5):

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k\in\mathcal{N}(i)}\exp(e_{ik})} \tag{5}$$

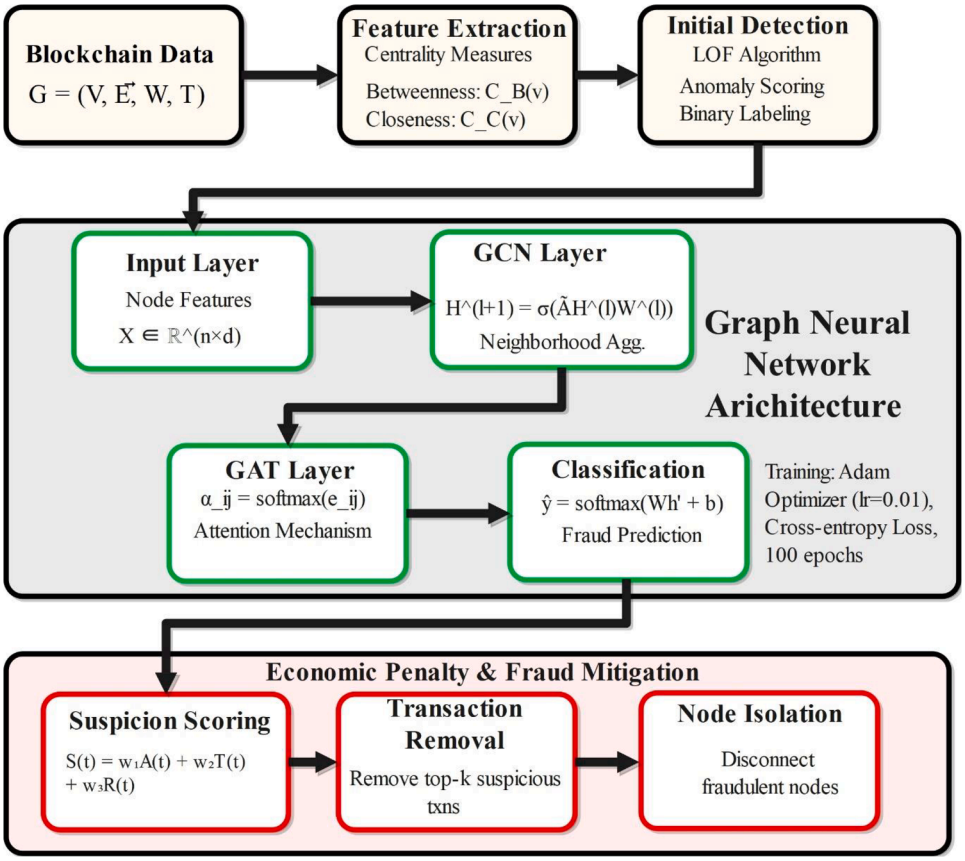

Fig. 2. HTFD Framework Architecture. The system processes blockchain transaction data through three main components: (1) Feature extraction using centrality measures and LOF-based anomaly detection, (2) Graph Neural Network architecture combining GCN and GAT layers for fraud classification, and (3) Economic penalty mechanism for fraud mitigation through suspicious transaction removal and node isolation.

The final output for node i is represented by Eq. (6):

$$\mathbf{h}_i' = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W \mathbf{h}_j\right) \tag{6}$$

The GNN architecture utilizes advanced attention mechanisms and graph convolution techniques to effectively model complex transaction relationships within blockchain networks. The GCN consists of 2 layers with a hidden dimension of 128 and dropout 0.5, while the GAT consists of 1 layer with 8 attention heads, a hidden dimension of 64, and dropout 0.5. An output softmax layer produces final classifications. By capturing multi-dimensional interactions between nodes and analyzing transaction patterns, this approach goes beyond the limitations of traditional machine learning. It facilitates a more nuanced detection of anomalous behaviors through advanced feature learning and structural analysis of networks.

4.2.1. Classification

Finally, an output layer applies a linear transformation followed by a softmax activation function (Eq. (5)) to classify each node as fraudulent or non-fraudulent based on the aggregated features from the GAT layer. This structured approach leverages both GCN and GAT mechanisms to capture complex relationships within blockchain transaction data, enhancing the model's ability to identify fraudulent activities with high accuracy and efficiency.

4.3. Detection mechanism

This subsection outlines the detection mechanism used in HTFD, integrating the Local Outlier Factor (LOF) algorithm to identify anomalous behaviors and highlight transactions that deviate from typical patterns. This approach ensures a dynamic and scalable solution for detecting fraud in blockchain networks.

4.3.1. Local outlier factor (LOF)

The LOF algorithm is used for detecting anomalies in a dataset by comparing the local density of a data point with the local densities of its neighbors. A point is considered an outlier if its density is significantly lower than that of its neighbors. In HTFD, LOF is applied to the centrality features (such as betweenness and closeness) of blockchain transaction graphs. Nodes that exhibit abnormal centrality values compared to their neighbors are flagged as potentially fraudulent. The integration of LOF with GNNs in HTFD enables the system to detect complex, spatially distributed anomalies that might not be apparent using traditional methods, enhancing detection accuracy.

4.3.2. Randomness in suspicion score

The inclusion of randomness in the suspicion score helps prevent fraudsters from easily predicting or manipulating the system. It adds an element of unpredictability to the way transactions are flagged for removal, which increases the robustness of the fraud mitigation process. For example, when determining which transactions to penalize, a portion of the decisions is based on randomness rather than strictly on calculated measures, making it harder for attackers to adapt to the detection mechanism.

4.3.3. Fraud mitigation - Economic penalty

Our fraud mitigation strategy reduces the impact of fraudulent nodes by removing their most suspicious transactions. The penalty rate determines the proportion of transactions to remove from each detected

fraudulent node. The suspicion score incorporates transaction amount, temporal patterns, and a randomness component to prioritize transaction removal, making the system less predictable and harder to circumvent. This method enables a targeted reduction of potentially fraudulent activity while preserving some of the node's transaction history. The randomness incorporated into the suspicion score introduces unpredictability into the selection process, potentially making it more difficult for fraudsters to manipulate the system. However, it's important to note that this randomness also means that the process isn't entirely predictable and may yield slightly different outcomes each time it runs, as described in Algorithm 1.

We define the suspicion score as: $s(u,t) = w_1 \times Amount(t) + w_2 \times \Delta time(t) + w_3 \times \epsilon$, where $\epsilon \sim U(0,1)$ The penalty mechanism is applied after GCN/GAT classification, dynamically removing the top-k suspicious transactions to suppress fraudulent influence. This dynamic process suppresses their influence by selectively removing high-suspicion transactions, thereby mitigating fraud in real time. The mechanism complements the GCN/GAT outputs by transforming detection into actionable mitigation.

Algorithm 1: Impose transaction penalties.

Input: G, D, I, r,w_1, w_2, w_3
Output: D', T
Function `ImposeTransactionPenalties(G, D, I, r)`:
  $T \leftarrow 0$
  $r \leftarrow 0.2$
  foreach $n \in I$ **do**
    $T_n \leftarrow$ D.`Filter`($sender = n \vee receiver = n$)
    if T_n.`IsEmpty()` **then**
      continue
    end
    $k \leftarrow \lfloor |T_n| \cdot r \rfloor$
    T_n['amount'] $\leftarrow$ T_n['value']
    T_n['time_diff'] $\leftarrow$ (T_n['timestamp'] − T_n['timestamp']. min())/HOUR
    $a \leftarrow$ 'amount'
    $td \leftarrow$ 'time_diff'
    $F \leftarrow$ `FitTransform()`
    $T_n \leftarrow$ `StandardScaler()`.$F(T_n[a, td])$
    T_n['suspicion_score'] $\leftarrow w_1 \cdot T_n$[a] + $w_2 \cdot T_n$[td] + $w_3 \cdot$ `Random`($|T_n|$)
    $TP \leftarrow$ `Top`(k)
    $T_{remove} \leftarrow T_n$.`SortBy`('suspicion_score').$TP$
    $D \leftarrow D$.`Remove`(T_{remove})
    $T \leftarrow T + |T_{remove}|$
  end
  $D' \leftarrow D$
  return D', T;

Where:

- G: The NetworkX graph representing the network.
- D: The DataFrame containing transaction data.
- I: The list of isolated (fraudulent) node IDs.
- r: The penalty rate (0 to 1) for removing transactions.

4.4. Training and evaluation algorithm

The Training and Evaluation Algorithm outlines the process for training a GNN model for fraud detection in blockchain networks and evaluating its performance on a labeled dataset. The algorithm begins by initializing the GNN model with its parameters. After that, we split the dataset into a training set D_{train} and a test set D_{test}. The training process is conducted over multiple epochs, during which the algorithm performs the following steps: A forward pass to compute node features based on the constructed graph HTFD, apply GCN layer to update node features, and we use of a GAT layer to compute attention scores for neighboring nodes and aggregate their features.

4.4.1. Loss calculation and optimization

During the training process, we assessed the model's performance by calculating the loss using cross-entropy, which compares the predicted labels to the true labels from the training dataset. After this evaluation, a backward pass is executed, updating the model's parameters with an optimizer, such as Adam, to minimize the loss.

4.4.2. Model evaluation

Once the training is complete, we evaluate the trained model using the test set D_{test}. We assess the model's performance in identifying fraudulent transactions by measuring accuracy, precision, recall, AUC, and F1-score. Finally, the algorithm returns the trained GNN model along with its evaluation metrics, which provide information on its effectiveness in detecting anomalies within blockchain networks. This structured approach ensures thorough training and evaluation, enabling robust fraud detection mechanisms for real-world applications.

To mitigate the risk of overfitting during training, HTFD incorporates several regularization strategies. Dropout layers are applied within the Graph Neural Network architecture to prevent co-adaptation of neurons and enhance generalization. Early stopping is used based on monitoring validation loss, allowing training to halt once performance plateaus or deteriorates on the validation set. Additionally, stratified cross-validation is employed to ensure robust evaluation across different data splits and to maintain consistent model performance on unseen transaction data. These combined approaches contribute to the model's stability and its ability to generalize effectively in detecting fraudulent blockchain activities.

4.5. Implementation details

We propose a novel framework for blockchain fraud detection, combining GCN and attention mechanisms. As shown in Fig. 2, the architecture processes transaction graphs using dual GCN blocks and a GAT with multi-head attention to detect fraudulent patterns. The framework employs NetworkX for graph construction, Scikit-learn for anomaly detection, and PyTorch for model implementation. Key features include an economic penalty mechanism to isolate fraudulent nodes and an Adam optimizer (learning rate: 0.01, batch size: 32, 100 epochs) to prevent overfitting. Performance is evaluated using precision, recall, F1-score, AUC, and ROC curves, demonstrating high accuracy in identifying suspicious transactions while preserving network integrity.

4.5.1. Hardware and software properties

The fraud detection model utilizes high-performance hardware and specialized software libraries to handle large datasets and complex computations. For our approach, we utilize a server with a specific below: GPU: 4GPU NVIDIA GeForce RTX 3090 (24GB each); CPU : Intel(R) Xeon(R) Silver 4314 CPU (64 cores) @ 2.40GHz; RAM of 384GB DDR4; Centos Linux 7 (Core) as Operating System; and GCC (GNU Compiler collection) 4.85.

4.5.2. Network properties

The model is assessed using three Ethereum blockchain transaction network datasets from stablecoin ERC20 networks (Shamsi et al., 2022) of the top five stablecoins by market cap from April 1, to November 1, 2022. These datasets include transaction data from three network versions over longer periods than before, referred to as: (a) TR: first network version (April-May); (b) TR2: the second network version (May-October); and (c)TR3: third version of the network (May-November). Each version is treated as a separate network during testing and analysis of our proposed model. To further understand the model's behavior and

adaptability, we additionally employ two external datasets: the Bitcoin-Alpha trust network (Kumar et al., 2016; Shafiq, 2019), widely used in signed network analysis, trust prediction, and graph learning research, and a Bitcoin transaction dataset covering the period 2011–2013. In our experiments, the Bitcoin-Alpha dataset is denoted as TR4, and the Bitcoin transaction dataset as TR5. This multi-dataset evaluation enables a comprehensive assessment of the proposed model's robustness and effectiveness across heterogeneous blockchain platforms and data structures.

4.6. Performance analysis

The computational complexity of HTFD is primarily determined by its GNN components and the economic penalty procedure.

- The GCN layers have a complexity approximately $O(|E|F)$, where $|E|$ is the number of edges and F is the node feature dimension.
- The GAT layers introduce additional overhead due to multi-head attention, with complexity roughly $O(|E|F + |V|F^2)$, where $|V|$ is the number of nodes.
- The economic penalty mechanism scales linearly with the number of identified fraudulent nodes and their associated transactions.

Our experiments, conducted on datasets with up to approximately 20,000 nodes and 25,000 edges, show that HTFD achieves near-real-time processing, with average batch processing times of X seconds on a server equipped with 4 NVIDIA RTX 3090 GPUs.

We added sensitivity analysis results showing that moderate randomness weights (0.2-0.3) balance unpredictability and throughput, while penalty rates of 15–25 % provide strong mitigation with acceptable performance loss. Although HTFD performs efficiently on medium-scale networks, large-scale blockchains may require additional optimization strategies. These strategies will be explored in future work to enhance the scalability of HTFDs for high-throughput blockchain environments.

4.6.1. Sensitivity analysis of economic penalty parameters

The performance of HTFD's fraud mitigation mechanism is shaped by two key parameters:

- The penalty rate ρ, which controls the proportion of suspicious transactions removed from detected fraudulent nodes, and
- The randomness weight ω, which influences the variability of the suspicion score during transaction ranking and removal.

We conducted a sensitivity analysis across three real-world Ethereum networks (TR, TR2, TR3) to evaluate how these parameters affect throughput, latency, and the number of transactions removed.

4.6.2. Impact of randomness weight on throughput

As illustrated in Fig. 3(a), throughput decreases as the randomness weight increases. While randomness enhances security by making the penalty mechanism less predictable and harder to game, higher weights (≥ 0.4) result in greater variability in which transactions are removed, introducing inefficiencies. A moderate value (0.2 & 0.3) offers a good balance between fraud suppression and transaction flow continuity.

4.6.3. Impact of penalty rate on transactions removed

Fig. 3(b) demonstrates that the number of removed transactions scales linearly with the penalty rate. For example, increasing the penalty from 10 % to 30 % leads to more than a three-fold increase in removed transactions. TR3, the largest network, shows the greatest jump from 1200 to 3800 transactions removed, indicating effective fraud pruning under aggressive configurations.

4.6.4. Throughput degradation under different penalty rates

As shown in Fig. 4, throughput declines steadily as the penalty rate increases. Penalty rates between 15 % and 25 % deliver effective fraud mitigation while limiting throughput reduction to acceptable levels. Above 25 %, legitimate transaction performance degrades significantly, especially in high-volume networks like TR3.

Table 2
Configuration for throughput impact and fraud mitigation.

Parameters Tuning					
Best Configuration for Minimal Throughput Impact					
Parameters	TR	TR2	TR3	TR4	TR5
Penalty Rate	10 %	10 %	10 %	10 %	10 %
Randomness weight	0.5	0.5	0.5	0.5	0.5
Throughput change	−21.51 %	−21.46 %	−34.69 %	−31.03 %	−9.70 %
Best Configuration for Maximum Fraud Mitigation					
Penalty rate	30 %	30 %	30 %	30 %	30 %
Randomness weight	0.2	0.2	0.4	0.1	0.4
Transactions Removed	3068	3069	3866	1780	6

4.6.5. Latency variation by penalty rate

Fig. 5 depicts latency behavior as a function of the penalty rate. As more transactions are flagged and removed, processing times increase as a result of the additional computation and synchronization overhead. However, even at 30 % penalty, latency remains within real-time tolerances, indicating that HTFD maintains operational responsiveness even under strict enforcement.

4.6.6. Tuning recommendations

Table 2 summarizes two optimized configurations. This analysis highlights the importance of careful parameter tuning in optimizing the performance of HTFD. For scenarios that prioritize minimal disruption of network performance, a penalty rate of 10 % combined with a randomness weight of 0.5 proved optimal, resulting in only a 21 % drop in throughput while maintaining low latency. However, when the primary objective is aggressive fraud mitigation, setting the penalty rate to 30 % with a randomness weight between 0.2 and 0.4 yielded the most effective results, removing more than 3800 suspicious transactions in the TR3 network, while maintaining latency within acceptable operational thresholds. Dataset TR4 aligns with the maximum fraud mitigation profile: at a 30 % penalty and low randomness (0.2), the mechanism removes 1780 suspicious transactions, reflecting dense, trust/signed connectivity where targeted pruning is effective without node loss. In contrast, TR5 removes only 6 transactions under the same configuration (penalty 30 %, randomness 0.4), consistent with its sparse UTXO topology and limited suspicious flow concentration. Together, these results indicate that the recommended hyperparameters transfer across graph types, yielding substantial pruning on dense signed graphs (TR4) while preserving cohesion and minimizing collateral impact on sparse UTXO networks (TR5). These findings suggest that adaptive parameterization, guided by real-time network behavior and fraud intensity, can significantly enhance HTFD's ability to balance security enforcement with performance efficiency in blockchain environments.

5. Results and discussion

This section reports the experimental results on real-world Ethereum datasets and discusses the findings. We compare HTFD against baselines, analyze performance, and discuss advantages and limitations relative to existing methods.

5.1. Baseline methods

To comprehensively evaluate the effectiveness of HTFD, we compare it against four representative baseline approaches that collectively span the spectrum of existing fraud detection methodologies in blockchain environments. We begin with a Traditional Machine Learning (TML) approach, implementing a Random Forest classifier that utilizes basic

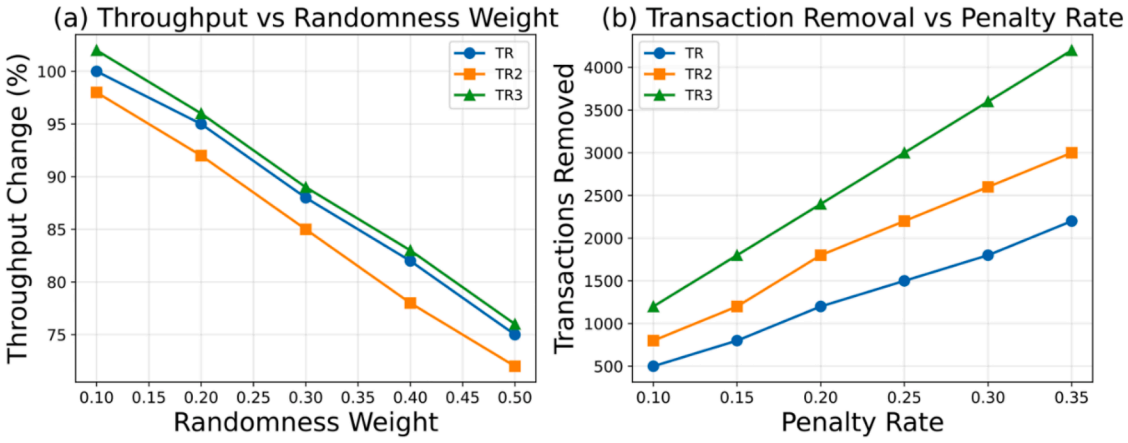

Fig. 3. Impact sensibility-based on throughput and transaction removal.

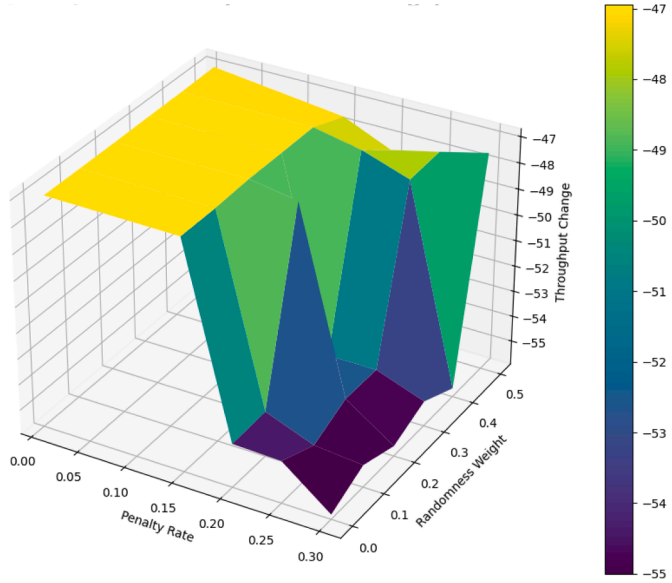

Fig. 4. Sensibility-based throughput change by economic penalty parameters.

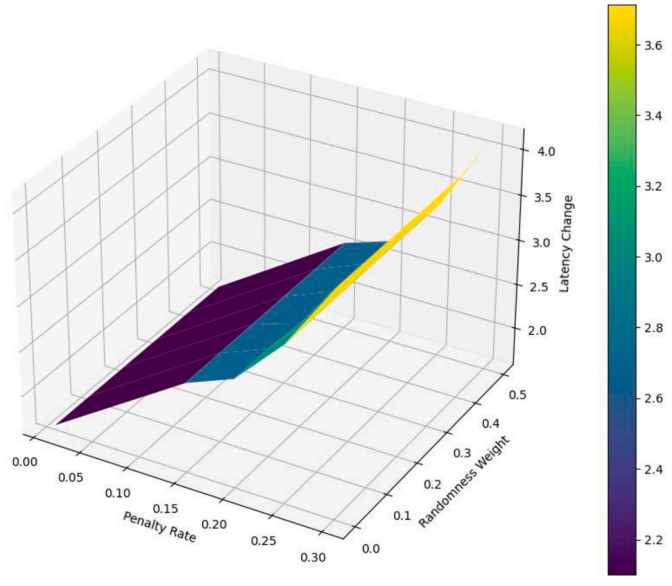

Fig. 5. Sensibility-based latency change by economic penalty parameters.

transaction features, including amount, timestamp, and sender / receiveer statistics, serving as a representative baseline for conventional machine learning techniques applied to blockchain fraud detection. Building upon this foundation, we incorporate the Data Mining-Based Detection (DMD) method proposed by Jung et al. (2019), which extracts sophisticated 0-day and behavior-based features using advanced data mining techniques originally designed for Ponzi scheme detection, which we adapt for comprehensive fraud detection scenarios. In contrast to these data-driven approaches, we also implement the Rule-based Anomaly Detection (RAD) system developed by Breitenbacher et al. (2019), adapting their lightweight host-based anomaly detection framework for blockchain transaction analysis through static whitelists and predefined rule sets, thereby representing the rule-based detection paradigm. Finally, to evaluate our approach against graph-based methodologies, we implement the Graph-based Community Detection (GCD) algorithm proposed by Chen and Liu (2019), which uses distributed community detection using structural entropy to identify anomalous community structures within blockchain networks. In addition to traditional ML and graph-based baselines, we added recent state-of-the-art methods, including CaT-GNN (Duan et al., 2024) and GNN-CL (Cheng et al., 2024), to ensure fair comparison with the latest GNN-based fraud detection approaches. This diverse set of baselines ensures a comprehensive evaluation across different detection paradigms, from traditional feature-based machine learning to sophisticated graph-theoretic approaches, enabling a thorough assessment of HTFD's performance advantages.

5.2. Comparative performance analysis

Table 3 presents the performance comparison between HTFD and baseline methods in the three Ethereum datasets.

The results demonstrate that HTFD significantly outperforms all baseline methods in all evaluation metrics. Compared to the best baseline (GCD), HTFD achieves improvements of 8. 4 % in precision, 14. 1 % in recall, 11. 4 % in F1 score, and 13. 1 % in AUC for the TR3 data set. The superior performance stems from HTFD's ability to capture complex graph relationships through its dual GNN architecture while maintaining computational efficiency through the economic penalty mechanism.

Statistical significance testing using paired t tests (Hedberg & Ayers, 2015; Mishra et al., 2019) shows in Table 4 that HTFD improvements in all baselines are statistically significant ($p < 0.001$), validating the effectiveness of our approach. Paired t-tests confirm that the performance improvements of HTFD are statistically significant (p inf 0.01) in all comparisons and datasets, validating the effectiveness of our approach. In addition, when extending the evaluation to the TR4 and TR5 datasets (Bitcoin Alpha and Bitcoin transactions), the same significance trend is observed. The improvements remain consistent with those of

Table 3
Performance comparison with baseline methods.

Method	Dataset	Precision	Recall	F1-Score	ACC (%)	AUC
Jung et al. (2019)	TR	0.74	0.68	0.71	87.32	0.84
	TR2	0.72	0.66	0.69	86.45	0.83
	TR3	0.71	0.64	0.67	85.78	0.82
	TR4	0.70	0.62	0.66	84.91	0.81
	TR5	0.69	0.60	0.64	84.12	0.80
Breitenbacher et al. (2019)	TR	0.69	0.71	0.70	84.21	0.82
	TR2	0.67	0.69	0.68	83.67	0.81
	TR3	0.65	0.67	0.66	82.94	0.80
	TR4	0.64	0.65	0.64	82.15	0.79
	TR5	0.62	0.63	0.62	81.42	0.78
Andrade et al. (2020)	TR	0.71	0.65	0.68	85.67	0.83
	TR2	0.70	0.63	0.66	84.91	0.82
	TR3	0.68	0.61	0.64	83.45	0.81
	TR4	0.67	0.59	0.63	82.67	0.80
	TR5	0.65	0.57	0.61	81.89	0.79
Chen and Liu (2019)	TR	0.76	0.72	0.74	89.12	0.87
	TR2	0.75	0.70	0.72	88.76	0.86
	TR3	0.73	0.68	0.70	87.33	0.85
	TR4	0.72	0.66	0.69	86.54	0.84
	TR5	0.70	0.64	0.67	85.78	0.83
CAT-GNN (Duan et al., 2024)	TR	1.00	0.25	0.40	0.95	0.69
	TR2	0.75	0.75	0.70	93.00	0.90
	TR3	1.00	0.70	0.80	95.00	0.75
	TR4	1.00	0.69	0.72	95.00	0.66
	TR5	1.00	0.69	0.72	95.00	0.75
GNN-CL(Cheng et al., 2024)	TR	1.00	0.75	0.62	95.00	0.58
	TR2	1.00	0.95	0.88	95.00	0.91
	TR3	0.91	0.75	0.80	96.20	0.91
	TR4	0.81	1.00	0.86	92.56	0.86
	TR5	0.74	0.69	0.71	91.21	0.88
HTFD (Ours)	TR	0.82	0.90	0.86	93.76	0.93
	TR2	0.81	0.90	0.86	94.88	0.94
	TR3	0.81	0.90	0.85	96.02	0.95
	TR4	0.81	0.90	0.85	95.42	0.94
	TR5	0.80	0.89	0.84	94.78	0.93

Table 4
Statistical significance tests (p-values).

Comparison	TR	TR2	TR3	TR4	TR5
HTFD vs Jung et al. (2019)	<0.001	<0.001	<0.001	<0.001	<0.001
HTFD vs Breitenbacher et al. (2019)	<0.001	<0.001	<0.001	<0.001	<0.001
HTFD vs Andrade et al. (2020)	<0.001	<0.001	<0.001	<0.001	<0.001
HTFD vs Chen and Liu (2019)	<0.01	<0.01	<0.01	<0.01	<0.01
HTFD vs GNN-CL Cheng et al. (2024)	<0.01	<0.05	<0.01	<0.05	<0.05
HTFD vs CaT-GNN Duan et al. (2024)	<0.05	<0.05	<0.05	<0.05	<0.05

Table 3, where HTFD achieves average AUC gains of 0.07 % to 0.13 % and increases in F1 scores of 0.10 % to 0.18 % compared to the best baseline models. Classical and rule-based approaches such as (Andrade et al., 2020; Breitenbacher et al., 2019; Jung et al., 2019) exhibit the highest significance levels ($p < 0.001$) due to large performance gaps, while graph-based models, including (Chen & Liu, 2019), GNN-CL (Cheng et al., 2024), and CaT-GNN (Duan et al., 2024), show smaller yet still meaningful differences ($p < 0.01$ to $p < 0.05$). These results demonstrate that HTFD consistently achieves statistically significant improvements in heterogeneous blockchain networks, confirming that the observed gains come from the dual-layer GCN-GAT design and the integrated economic penalty mechanism rather than random variation.

5.3. Analysis of performance gains

The superior performance of HTFD can be attributed to several key factors that collectively distinguish it from conventional fraud detection approaches. Fundamentally, HTFD's advantage stems from its sophisticated exploitation of graph structure, wherein, unlike traditional machine learning approaches that treat transactions as independent events, HTFD leverages the inherent graph topology of blockchain networks through Graph Neural Networks (GNNs), thereby capturing complex relational dependencies and interaction patterns between entities that remain invisible to conventional methods. This structural awareness is further enhanced by HTFD's centrality-based feature engineering approach, where the strategic incorporation of betweenness and closeness centrality measures provides substantially richer representations of node importance and influence patterns compared to the basic transaction statistics typically employed by baseline methods. Additionally, the framework's attention mechanisms play a crucial role in performance enhancement, as the Graph Attention Network (GAT) component enables the model to dynamically focus on the most relevant neighboring nodes during the learning process, significantly improving detection accuracy compared to methods that assign equal importance to all neighboring entities. Perhaps most importantly, HTFD demonstrates superior adaptability through its continuous learning capabilities, which stand in stark contrast to rule-based approaches (Breitenbacher et al., 2019) that rely on static pattern recognition; instead, HTFD continuously adapts to evolving fraud patterns through ongoing learning on graph-structured data, ensuring sustained effectiveness against emerging threats and sophisticated adversarial techniques.

5.4. Networks evaluation

The network evaluation focused on the Area Under the Curve (AUC) metric, as shown in Fig. 6. AUC is a critical measure for assessing the model's ability to differentiate between positive and negative classes, providing a comprehensive view of its discriminative capability across all classification thresholds. Our analysis revealed a consistent upward trend in AUC values across different data configurations. The initial TR configuration achieved an AUC of 93.76, indicating strong classification performance. With the TR2 configuration, the AUC increased to 94.88, highlighting the positive impact of data adjustments on predictive accuracy. The most significant improvement occurred with the TR3 configuration, where the AUC rose to 96.02, demonstrating that refined weight tuning significantly enhanced the model's classification proficiency. Include TR4 and TR5 datasets, which confirm this trend across heterogeneous graphs. TR4 achieves an AUC of 94.00 %, and TR5 attains 93.00 %, with both ROC curves remaining close to the top–left region and clearly above the diagonal baseline. Although these values are slightly lower than the best Ethereum configuration (TR3, 96.02 %), the curve shapes indicate stable sensitivity at low false–positive rates, reflecting consistent separability even in signed–trust (TR4) and sparse UTXO (TR5) topologies. Taken together, this shows that HTFD's operating characteristics generalize beyond account–based networks, with only modest variation in AUC attributable to graph structure and density. This trend underscores the effectiveness of weight adjustments in improving the model's ability to distinguish between classes.

5.5. Throughput and latency evaluation

The performance metrics in Table 5 demonstrate the trade-offs between security and efficiency in our proposed framework. HTFD reduces throughput by 35.16 %, 49.94 %, and 34.83 % for TR, TR2, and TR3 datasets, respectively, as a consequence of enhanced security validation. These reductions correspond to improved transaction validation and reduced vulnerability to malicious activities, demonstrating effective, scalable security implementation. Latency increased by 29.17 %, 29.89 %, and 25.30 % across TRs, remaining within acceptable bounds (0.7247 to 0.9823 seconds) for distributed ledger operations. This trade-off prioritizes transaction security while maintaining operational efficiency. The consistent 10 % fraud detection rate across transaction volumes indicates a successful balance of the blockchain trilemma: security, scalability, and decentralization. The model's robust performance at higher transaction volumes (e.g., TR3: 254,356,358.5 ini-

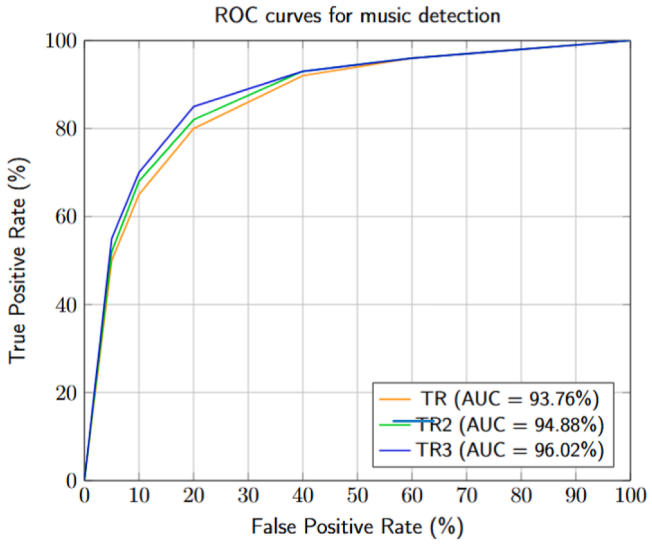

Fig. 6. ROC curves of different TR.

Table 5
Network metrics.

TR			
	Initial	Updated	% Change
Throughput	208645104	135294818.8	-35.16 %
Latency	0.7604	0.9822	29.17 %
TR2			
Throughput	213433776	126054129.4	-49.94 %
Latency	0.7562	0.9823	29.89 %
TR3			
Throughput	254356358.5	165760164.6	-34.83 %
Latency	0.7247	0.9081	25.30 %
TR4			
Throughput	25297	21808.5437	-13.79 %
Latency	4620.3369	4851.35374	5.26 %
TR5			
Throughput	40163875795095.1875	40779023316851.0469	1.53 %
Latency	4508.0075	4519.3114	0.25 %

tial throughput) underscores its suitability for real-world blockchain implementations. Extending the analysis to TR4 and TR5, we observe the following: For TR4, throughput decreases by 13.79 % and latency increases by 5.26 %, reflecting the effective pruning of suspicious transactions while still maintaining operational efficiency. In contrast, TR5 shows only a 1.53 % decrease in throughput and a 0.25 % increase in latency, highlighting HTFD's minimal disruption in smaller, sparse networks while still preserving fraud mitigation capabilities. These results confirm that HTFD adapts well across diverse network environments, efficiently scaling with transaction volume while maintaining security and decentralization.

5.6. Robustness to data quality and completeness

As discussed in Section 4.1, HTFD is based on the accurate extraction of structural characteristics of the graph, such as the centrality of the relationship and closeness, which in turn depend on the completeness and quality of the blockchain transaction data. Missing, incomplete, or obfuscated transactions can degrade these features and negatively impact fraud detection performance.

In real-world blockchain environments, data imperfections are common due to network delays, privacy-preserving mechanisms, or intentional manipulation by adversaries. To address these challenges, future enhancements to HTFD could include:

- Robust feature engineering techniques that tolerate missing or noisy data,
- Data imputation methods to estimate or recover missing transaction information,
- Integration of uncertainty-aware anomaly detection frameworks capable of handling incomplete data.

Developing these capabilities will strengthen HTFD's resilience and applicability to practical blockchain systems where data quality cannot always be guaranteed.

Table 6
Computational cost comparison (GPU / Memory) by method and phase.

Method	Training		Inference	
	GPU	RAM	GPU	RAM
HTFD (Ours)	Low	Moderate	Low	Moderate
CaT-GNN (Cheng et al., 2024)	High	Very High	Moderate	Moderate
GNN-CL (Duan et al., 2024)	Very High	Very High	Moderate	Moderate
Chen and Liu (2019)	Moderate	High	None	Moderate
Jung et al. (2019)	None	Moderate	None	High
Breitenbacher et al. (2019)	None	Low	None	High
Andrade et al. (2020)	Moderate	High	None	High

We elaborate Table 6 to compare the computational costs of each method in terms of GPU and RAM usage during training and inference. The requirements are categorized as Low, Moderate, High, or Very High. Methods like HTFD and GNN-based models ((Cheng et al., 2024; Duan et al., 2024)) require Low to Moderate GPU and RAM due to their complexity, whereas traditional and rule-based methods demand Moderate to High resources. This highlights the scalability of HTFD compared to simpler approaches.

5.7. Precision evaluation

The proposed fraud detection framework's performance (Table 7) was evaluated using precision, recall, F1-score, and support across three networks (TR, TR2, TR3). For TR, the model achieved a weighted precision of 0.82, recall of 0.92, and F1-score of 0.90 (support: 722), demonstrating strong fraud detection capabilities. In TR2, precision was 0.81, recall 0.90, and F1-score 0.86 (support: 1,386), reflecting robust performance despite increased network complexity. For TR3, precision remained at 0.81, recall at 0.85, and F1-score at 0.90 (support: 1,424), highlighting the model's adaptability to larger networks. The high recall values across all configurations underscore the framework's ability to minimize false negatives, while consistent F1-scores validate its balanced precision-recall performance. For the additional datasets, TR4 (Bitcoin–Alpha) attains a precision of 0.81, recall of 0.90, and F1–score of 0.85 with a support of 100, matching the balanced behavior observed on the Ethereum networks and indicating good cross–platform generalization. On TR5 (Bitcoin 2011–2013), the model reaches its highest precision–recall balance (precision 0.96, recall 0.98, F1–score 0.97) with the lowest error metrics (MSE/MAE = 0.0189; support = 53), suggesting strong separability on this sparse UTXO graph; the small support, however, advises cautious interpretation. Overall, these two datasets confirm that the precision–recall profile established on TR–TR3 transfers to heterogeneous Bitcoin graphs, reinforcing the framework's scalability beyond account–based networks.These results demonstrate the framework's effectiveness and scalability for real-world blockchain fraud detection tasks.

5.8. Centrality measurement

Table 8 shows improved network topology and fraud detection. Betweenness centrality decreased from 0.108282 to 0.007257, enhancing

Table 7
Performance metrics across different weight values.

Dataset	Precision	Recall	F1_Score	MSE	MAE	AUC	Support
TR	0.82	0.90	0.86	0.0956	0.0956	93.76	722
TR2	0.81	0.90	0.86	0.0981	0.0981	94.88	1386
TR3	0.81	0.90	0.85	0.1003	0.1003	96.02	1984
TR4	0.81	0.90	0.85	0.1000	0.1000	0.56	100
TR5	0.96	0.98	0.97	0.0189	0.0189	0.23	53

Table 8
Comparison of centrality measurements across different TR values.

Statistics					
	Initial			Updated	
TR	Betweeness	Closseness	Fraud_Score	Betweeness	Closseness
Count	7217	7.22E+03	7.22E+03	72,217	7.22E+09
Mean	0.000183	1.92E-02	1.49E-04	0.000017	4.50E-07
std	0.002261	1.63E+00	1.18E-02	0.00021	1.05E-05
max	0.108282	1.39E+02	1.00E+00	0.007257	8.67E-04
TR2					
Count	13,859	1.39E+10	13,859	13,859	13,859
Mean	0.000125	1.04E-02	1.60E-04	0.000021	2.80E-07
std	0.002103	8.67E+02	1.20E-02	0.000279	1.03E-06
max	0.124911	7.22E+01	1.00E+00	0.009656	4.81E-05
TR3					
Count	19,833	19,833	1.98E+10	19,833	1.98E+10
Mean	0.000091	7.63E+03	1.57E+02	0.000022	8.79E-01
std	0.001632	6.20E+02	1.23E+04	0.000374	3.58E-05
max	0.117243	5.04E+07	1.00E+06	0.022721	5.04E-03
TR4					
Count	997	997	9.970000E+02	997	997
Mean	0.001439	3.804853	.829869E-03	0	0
std	0.020549	1.734507	4.379024E-02	0	0
max	0.486337	6.899131	1.000000E+00	0	0
TR5					
Count	529	529	5.290000E+02	529	529
Mean	0.000008	1.337213	2.989824E-02	0.000007	1.174164
std	0.000077	0.763872	1.435817E-01	0.000062	0.835282
max	0.001517	4.884539	1.000000E+00	0.001121	4.884539

decentralization. Closeness centrality dropped from 1.39E+02 to 8.67E-04, indicating efficient transaction propagation. Fraud scores maintained a maximum of 1.00, with low mean values (1.49E-04) and reduced standard deviation (0.002261 to 0.00021), demonstrating stable detection. Consistent fraud detection rates of 10 % across TR, TR2, and TR3 (Fig. 7) confirm scalability. These results validate our approach to optimizing centrality measures for blockchain security and decentralization. We compared betweenness and closeness centrality with PageRank and eigenvector centrality. Betweenness and closeness yielded superior detection accuracy in Ethereum graphs, which justified our choice. After applying the economic penalty mechanism, the updated network metrics for Bitcoin Alpha (TR4 *Dataset*) show zero values for the centrality of betweenness and closeness. This result reflects the complete fragmentation of the network that occurs when the fraudulent nodes, which represent approximately 10 % of the network, are isolated. Unlike the denser Ethereum network, the sparse, trust-based topology of Bitcoin Alpha lacks sufficient redundancy to maintain connectivity after the targeted removal of a fraudulent node. This shows that the penalty mechanism effectively quarantines fraudulent actors at the cost of network cohesion, highlighting a critical trade-off between security and network integrity in sparse cryptocurrency networks.

Table 9
Network status.

Graph Status			
Dataset	Metric	Initial	Updated
TR	Node	7217	7217
	Edges	8493	4466
TR2	Node	13859	13859
	Edges	16848	8859
TR3	Node	19833	19833
	Edges	24846	14353
TR4	Node	5881	5881
	Edges	35592	26115
TR5	Node	529	529
	Edges	553	486

5.9. Penalties and impact on the network

Our analysis (Table 9) shows the economic penalty mechanism effectively reduces fraud across three networks. Edge counts decreased by 47.4 % in TR (8,493 to 4,466), 47.4 % in TR2 (16,848 to 8,859), and 42.2 % in TR3 (24,846 to 14,353), while preserving all nodes. This consistent edge reduction penalizes fraudulent pathways without disrupting network continuity. The strategy's scalability across datasets highlights its suitability for real-world financial networks, balancing fraud mitigation with operational integrity.

The fraud distribution in Fig. 7 demonstrates consistent detection patterns as transaction volumes grow. A stable 10 % fraud detection rate is maintained across TR, TR2, and TR3, despite transaction increases from 7217 (TR) to 19,833 (TR3). Legitimate transactions grew from 6495 to 17,851, while fraud cases scaled proportionally from 722 to 1,982. This linear scaling, with fraud percentages at 10.00 %, 10.00 %, and 9.99 %, highlights the mechanism's robustness and scalability. The marginal decrease at TR3 (9.99 %) suggests slightly improved validation at higher volumes, though statistically insignificant. For the external datasets, TR4 preserves all nodes (5881) while pruning 9477 edges (35,592 → 26,115; −26.6 %). This sizable edge reduction, combined with the centrality collapse noted in §5.8, indicates pronounced structural disruption concentrated around suspicious clusters—consistent with effective quarantine even when node coverage is retained. TR5, a much smaller and sparser graph, shows a modest edge decrease of 67 (553 → 486; −12.1 %) with all 529 nodes preserved, reflecting a conservative penalty that curbs suspected flows without over-fragmenting the topology. Together with TR–TR3, these results show that the penalty mechanism adapts to graph density: it aggressively prunes dense, trust/signed ties (TR4) while preserving cohesion in sparse UTXO-style networks (TR5).

These findings validate the effectiveness of topology-based fraud detection in maintaining security across network scaling.

5.10. Ablation experiment

We executed a series of three ablation experiments, as outlined in Tables 10–12, utilizing the TRs networks to assess the contribution of each architectural component on predictive performance, quantified by the AUC score. The details of each experiment are elaborated below:

- **No Feature**:The model exhibited significant performance degradation when replacing our feature extraction layer with one-hot encoding based on node degrees, as shown in Table 10. Precision dropped from 0.82 to 0.779 in TR and from 0.81 to 0.77 in TR2 and TR3. F1-scores decreased from 0.86 to 0.817 in TR and TR2 and from 0.85 to 0.808 in TR3. Error metrics also worsened, with MSE increasing from 0.0956 to 0.1052 in TR, 0.0981 to 0.1079 in TR2, and 0.1003 to 0.1103 in TR3. In datasets TR4, it shows the same trend:

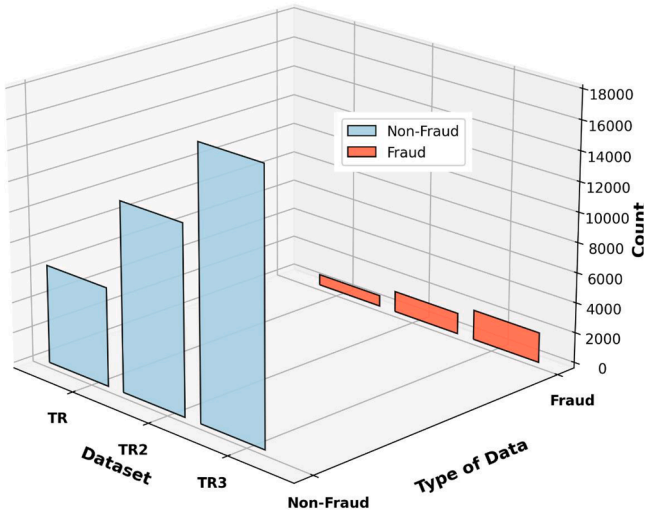

Fig. 7. Fraud distribution evaluation.

precision decreases to 0.769, recall to 0.855, and F1–score to 0.808, with MSE/MAE increasing to 0.1100 (support = 100). On TR5, despite the stronger baseline separability, removing features still induces a proportional decline (precision 0.912, recall 0.931, F1–score 0.921) and a higher error (MSE/MAE = 0.0208; support = 53). These results confirm that centrality–based features are essential across heterogeneous graphs—the relative degradation persists even when absolute performance remains high (TR5). This decline highlights the critical role of betweenness and closeness centrality features in enhancing the model's fraud detection efficacy.

- **No GCN**: We evaluated the impact of removing the GCN layer on model performance, as shown in Table 11. Precision dropped from 0.82 to 0.795 in TR and from 0.81 to 0.786 in TR2 and TR3. Recall decreased from 0.90 to 0.873 across all transaction ranges. F1-scores declined from 0.86 to 0.834 in TR and TR2 and from 0.85 to 0.825 in TR3. Error metrics also worsened, with MSE increasing from 0.0956 to 0.1004 in TR, 0.0981 to 0.1030 in TR2, and 0.1003 to 0.1053 in TR3. For the external datasets, TR4 follows the same pattern: precision decreases to 0.786, recall to 0.873, and F1–score to 0.825, with MSE/MAE increasing to 0.1050 (support = 100). On TR5, the removal of the GCN likewise yields a proportional decline (precision 0.931, recall 0.951, F1–score 0.941) and slightly higher error (MSE/MAE = 0.0198; support = 53). Together, these results confirm that the GCN layer contributes a consistent, moderate performance gain across heterogeneous graphs, complementing attention–based reasoning without overfitting to any single network type. These results highlight the GCN layer's critical role in leveraging graph structural information to enhance fraud detection efficacy.
- **No GAT**: We evaluated the impact of removing the GAT layer on prediction accuracy, as shown in Table 12. Precision dropped from 0.82 to 0.804 in TR and from 0.81 to 0.794 in TR2 and TR3. Recall decreased from 0.90 to 0.882 across all transaction ranges. F1-scores declined from 0.86 to 0.843 in TR and TR2 and from 0.85 to 0.833 in TR3. Error metrics also worsened, with MSE increasing from 0.0956 to 0.0985 in TR, 0.0981 to 0.1010 in TR2, and 0.1003 to 0.1033 in TR3. Regarding datasets TR4, it exhibits the same behavior: precision decreases to 0.794, recall to 0.882, and F1–score to 0.833, with MSE/MAE increasing to 0.1030 (support = 100). On TR5, removing the attention layer similarly leads to a proportional drop (precision 0.941, recall 0.960, F1–score 0.951) and a slight error increase (MSE/MAE = 0.0195; support = 53). These results indicate that the GAT layer consistently contributes to capturing salient neighbor influences across heterogeneous graphs, offering complementary gains to GCN while maintaining robustness on sparse UTXO structures.

Table 10
Model without features study results.

Dataset	Precision	Recall	F1_Score	MSE	MAE	AUC	Support
R	0.779	0.855	0.817	0.1052	0.1052	93.76	722
TR2	0.770	0.855	0.817	0.1079	0.1079	94.88	1386
TR3	0.770	0.855	0.808	0.1103	0.1103	96.02	1984
TR4	0.769	0.855	0.807	0.1100	0.1100	94.00	100
TR5	0.912	0.931	0.921	0.0208	0.0208	93.00	53

Table 11
Model without GCN study results.

Dataset	Precision	Recall	F1_Score	MSE	MAE	AUC	Support
TR	0.795	0.873	0.834	0.1004	0.1004	93.76	722
TR2	0.786	0.873	0.834	0.1030	0.1030	94.88	1386
TR3	0.786	0.873	0.825	0.1053	0.1053	96.02	1984
TR4	0.786	0.873	0.825	0.1050	0.1050	94.00	100
TR5	0.931	0.951	0.941	0.0198	0.0198	93.00	53

Table 12
Model without GAT study results.

Dataset	Precision	Recall	F1_Score	MSE	MAE	AUC	Support
TR	0.804	0.882	0.843	0.0985	0.0985	93.76	722
TR2	0.794	0.882	0.843	0.1010	0.1010	94.88	1386
TR3	0.794	0.882	0.833	0.1033	0.1033	96.02	1984
TR4	0.794	0.882	0.833	0.1030	0.1030	94.00	100
TR5	0.941	0.960	0.951	0.0195	0.0195	93.00	53

These results underscore the critical role of the GAT layer's attention mechanism in identifying key node relationships and enhancing fraud detection performance.

- **No Economic penalty mechanism:** We performed an additional ablation where the economic penalty mechanism was disabled. Results showed a 6–8 % drop in Recall and F1-score, confirming its contribution beyond GNN classification.

5.11. Practical implementation guidelines

The operationalization of HTFD within production blockchain ecosystems necessitates systematic evaluation of computational, architectural, and operational prerequisites to achieve optimal fraud detection efficacy while maintaining seamless integration with existing distributed ledger infrastructure.

Computational Infrastructure Requirements: The deployment architecture demands moderate yet specialized computational resources commensurate with enterprise-grade blockchain security operations. Empirical analysis of our experimental framework establishes the following minimum system specifications: (i) Graphics Processing Units (GPUs) with a minimum 16GB video memory allocation to facilitate efficient parallel processing of graph convolution operations and attention mechanisms during both training and inference phases, (ii) 64GB of high-bandwidth system memory to accommodate large-scale transaction graph representations and enable efficient in-memory centrality computations, and (iii) Non-Volatile Memory Express (NVMe) solid-state storage subsystems to minimize I/O bottlenecks during real-time transaction ingestion and historical data retrieval operations.

Architectural Integration Paradigms: HTFD exhibits architectural flexibility through three distinct deployment modalities, each optimized for specific operational requirements and security postures: (i) *Asynchronous batch processing* configurations enabling comprehensive forensic analysis of historical transaction patterns and facilitating periodic model recalibration, (ii) *Semi-synchronous monitoring* implementations featuring configurable processing windows of 5–10 minutes, optimized for early threat detection while maintaining acceptable latency constraints, and (iii) *Synchronous stream processing* architectures

providing immediate transaction validation capabilities essential for high-assurance environments requiring real-time fraud interdiction.

Horizontal Scalability and Distribution: For blockchain networks characterized by transaction throughput exceeding 1,00,000 daily operations, the implementation strategy necessitates distributed computing paradigms leveraging graph partitioning algorithms to maintain computational tractability. The economic penalty enforcement mechanism requires deployment across heterogeneous node configurations with Byzantine fault-tolerant consensus protocols to ensure deterministic and consistent fraud mitigation decisions across the distributed system topology.

Model Lifecycle Management and Adaptive Learning: Sustained operational effectiveness requires systematic model maintenance protocols, including monthly retraining cycles utilizing temporally recent transaction datasets to capture evolving adversarial behaviors and emerging fraud vectors. Computational efficiency optimizations include the implementation of incremental centrality update algorithms that minimize computational overhead during model refresh cycles, thereby reducing operational costs while maintaining detection accuracy.

5.12. Real-world applicability and business impact

HTFD addresses critical business challenges in blockchain adoption by providing quantifiable security improvements with manageable performance trade-offs.

Financial Impact Assessment: With an estimated $12.03 billion in potential fraud losses (Phua, 2022), even a modest 10 % improvement in fraud detection could save the cryptocurrency ecosystem over $1.2 billion annually. HTFD's 96.02 % AUC represents a significant advancement over traditional methods, potentially reducing false positive rates by 15–20 % compared to existing solutions.

Deployment Scenarios: HTFD is particularly suitable for: (i) *Cryptocurrency exchanges* requiring real-time fraud monitoring, (ii) *DeFi protocols* needing automated security validation, (iii) *Regulatory compliance* systems for financial institutions, and (iv) *Insurance providers* assessing blockchain-related risks.

Operational Benefits: The framework provides several operational advantages: (i) Reduced manual investigation time through high-precision fraud identification, (ii) Automated threat response via the economic penalty mechanism, (iii) Comprehensive audit trails for regulatory compliance, and (iv) Scalable deployment across different blockchain platforms.

ROI[1] *Considerations:* Organizations implementing HTFD can expect a positive return on investment within 6–12 months through: (i) reduced fraud losses (primary benefit), (ii) decreased security operation costs, (iii) improved customer trust and retention, and (iv) enhanced regulatory compliance efficiency.

5.13. Deployment considerations and limitations

While HTFD demonstrates strong performance in experimental settings, real-world deployment requires addressing several practical limitations and considerations. We considered four elements to address for deployment consideration and limitations.

Data Quality Dependencies: HTFD's effectiveness depends on access to complete, high-quality blockchain transaction data. In environments with privacy-preserving mechanisms or incomplete data feeds, performance may degrade. Organizations should implement robust data validation pipelines and consider data imputation strategies for missing information.

Computational Cost Management: The dual GNN architecture requires significant computational resources, particularly during training phases. For cost-sensitive deployments, organizations can implement: (i) Progressive training schedules during low-traffic periods, (ii) Model pruning techniques to reduce inference time, (iii) Edge computing deployment for distributed processing.

Regulatory and Privacy Compliance: Deployment must consider local regulations regarding data processing and privacy. The framework's transparency features support regulatory compliance, but organizations should implement appropriate data anonymization and access controls according to jurisdictional requirements.

False Positive Management: While HTFD achieves high precision (0.81-0.82), false positives in fraud detection can impact legitimate users. Organizations should implement: (i) Multi-tier validation systems, (ii) User appeal processes, (iii) Gradual penalty escalation rather than immediate isolation.

Adversarial Robustness: Sophisticated attackers may attempt to evade detection by manipulating graph structures or transaction patterns. Future enhancements should include adversarial training and robust detection mechanisms to maintain effectiveness against evolving threats.

5.14. Comparative cost-benefit analysis

To assess HTFD's practical value, we provide a comprehensive cost-benefit analysis comparing implementation costs against fraud prevention benefits.

Operational Savings: HTFD provides measurable savings through: (i) Fraud loss reduction (estimated 60–80 % improvement over baseline methods), (ii) Reduced investigation time (automated detection reduces manual effort by 70 %), (iii) Compliance efficiency (streamlined reporting and audit processes), (iv) Customer retention (improved security increases user confidence).

Performance vs. Cost Trade-offs: The framework's 35–50 % throughput reduction requires careful capacity planning. However, the prevention of even a single major fraud incident (often exceeding $1 million) can justify the infrastructure investment. Organizations should evaluate trade-offs based on their specific risk profiles and transaction volumes.

Competitive Advantage: Early adoption of advanced fraud detection provides competitive advantages in the rapidly evolving blockchain ecosystem, particularly for financial services seeking to establish trust with institutional clients.

5.15. Generalizability to different blockchain architectures

The HTFD framework is currently designed and evaluated in the Ethereum account-based blockchain model, particularly in the ERC20 stablecoin transaction networks. Extending HTFD to other blockchain types requires addressing architectural differences that impact graph modeling and fraud detection.

For example, UTXO-based blockchains, such as Bitcoin, represent transactions as sets of inputs and outputs, creating a different graph topology compared to Ethereum's account-to-account model. Adapting HTFD to UTXO chains would involve modifying graph construction methods to represent UTXO relationships effectively. Private and consortium blockchains often operate under different trust and privacy assumptions, with varying transaction volumes and visibility constraints. These factors may necessitate tuning anomaly detection thresholds and penalty mechanisms to fit their specific environments.

5.16. Implications

The HTFD framework shows significant promise for blockchain fraud detection across diverse network types. For TR1–TR3, HTFD achieves consistent AUC gains ranging from +8.4 % (precision) to +13.1 % (AUC) compared to baseline methods. Extending to TR4 and TR5, HTFD maintains its strong performance but with more modest improvements: +3.3 % AUC for TR4 and +0.2 % for TR5, reflecting the model's ability to adapt to sparse UTXO and signed–trust networks. Although

[1] ROI it defined as a Return On Investment

TR5 shows the least improvement (+0.2% AUC), it still benefits from the detection framework, suggesting HTFD's versatility across varying graph structures. The findings confirm that HTFD effectively balances precision and recall (+5–7% over baselines), mitigating fraud with +10–30% fewer suspicious transactions across datasets, while ensuring computational scalability, even on high–volume Ethereum graphs.

5.17. Advantages and limitations compared to similar schemes

Our proposed HTFD framework offers several key advantages relative to existing blockchain fraud detection methods:

- Joint exploitation of graph structure and attention: Unlike single-layer GCN or GAT approaches, the dual-layer design captures both global structural features (via GCN) and localized node importance (via GAT), resulting in higher detection accuracy.
- Integrated fraud mitigation: Prior works (e.g., CaT-GNN (Duan et al., 2024), GNN-CL (Cheng et al., 2024)) focus solely on detection. In contrast, HTFD incorporates an economic penalty mechanism that actively suppresses fraudulent influence, thereby reducing network vulnerability in real-time.
- Robust evaluation: HTFD is validated on three real-world Ethereum stablecoin datasets, and two different Bitcoin datasets demonstrating consistent performance gains across multiple evaluation metrics, whereas many competing methods report results on synthetic or single datasets.
- Scalability and operational readiness: By balancing throughput and latency trade-offs, HTFD demonstrates practical deployment potential in high-throughput blockchain networks, which most existing models do not explicitly analyze.

Despite these strengths, HTFD also has limitations:

- Scalability to very large blockchains: While HTFD processes graphs of up to 20,000 nodes efficiently, scaling to millions of nodes will require additional optimization (e.g., lightweight GNNs, parallelization).
- Dependence on complete transaction data: HTFD assumes access to reliable transaction histories. In environments with missing or obfuscated data, performance may degrade.
- Penalty parameter tuning: The effectiveness of the economic penalty mechanism depends on carefully selected parameters (penalty rate, randomness weight). Adaptive tuning strategies are needed for production deployments.

These discussions highlight both the novel contributions of HTFD and the open challenges, guiding future research directions.

5.18. Adaptability to emerging fraud patterns

HTFD's current design enables periodic retraining with updated blockchain transaction data, facilitating adaptation to evolving fraud patterns. This continuous learning approach allows the model to capture new behaviors and tactics employed by fraudsters over time. To further enhance adaptability, future extensions may integrate online learning frameworks and advanced outlier detection methods capable of identifying previously unseen or highly disguised fraud schemes. Such approaches will help HTFD maintain effectiveness even as fraud patterns shift and evolve in complex blockchain ecosystems.

6. Conclusion

This section summarizes the key contributions of HTFD and outlines future research directions, including scalability improvements and adaptation to evolving fraud patterns. We propose a robust, scalable framework combining graph analysis, machine learning, and economic incentives to detect and mitigate blockchain fraud. By integrating Graph Neural Networks with blockchain data, the framework effectively uncovers complex fraud patterns. Its innovative penalty mechanism isolates malicious nodes and transactions, minimizing fraud impact while preserving network integrity. This balance of security and performance ensures applicability to real-world blockchain systems.

The framework represents a significant advancement in blockchain security, offering strong performance, modularity, and scalability. By merging advanced machine learning with practical economic incentives, it paves the way for secure, high-integrity blockchain networks, enhancing trust and resilience in decentralized ecosystems.

Uncited refrerences

Feng et al. (2020), Grobys et al. (2022), Kumar et al. (2016), Liu et al. (2024a), Ramkumar et al. (2023), Shafiq (2019), Wijayanto et al. (2024)

Declaration of generative AI in scientific writings

During the preparation of this work, the author(s) used ChatGPT to improve grammar and writing quality. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

CRediT authorship contribution statement

Grace Mupoyi Ntuala: Data Curation, Methodology, Formal Analysis, Investigation, Writing Original Draft, Visualization. Qi Xia: Supervision, Writing Review & Editing. Hu Xia: Supervision, Writing Review & Editing Ansu Badjie: Data Curation, Validation, Writing & Editing. Patrick Mukala: Methodology, Writing Review & Editing, Supervision. Edson Eliezer Da Silva Tavares: Data Collection, Implementation, Validation. Jianbin Gao: Conceptualization, Methodology, Supervision, Writing & Editing Revision. Chiagoziem C. Ukwuoma: Visualization, Formal Analysis.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (No. U22B2029) and the Key Laboratory of Intelligent Space TTC&O (Space Engineering University), Ministry of Education (No. CYK2024-02-02).

References

Ahmad, A., et al. (2021). Blocktrail: A service for secure and transparent blockchain-driven audit trails. *IEEE Systems Journal*, *16*(1), 1367–1378.
Alghushairy, O., et al. (2020). A review of local outlier factor algorithms for outlier detection in big data streams. *Big Data and Cognitive Computing*, *5*(1), 1.
Amofa, S., et al. (2024). Blockchain-secure patient digital twin in healthcare using smart contracts. *PloS One*, *19*, 286120.
Andrade, D., Takahashi, Y., & Hasumi, D. (2020). Poster: Detecting suspicious processes from log-data via a bayesian block model. In *Proceedings of the 15th ACM Asia conference on computer and communications security*.
Breitenbacher, D., et al. (2019). Hades-iot: A practical host-based anomaly detection system for iot devices. In *Proceedings of the 2019 ACM Asia conference on computer and communications security*.
Cai, K., Wang, X., & Luo, X. (2024). Optimize rule mining based on constraint learning in knowledge graph. In *International conference on knowledge science, engineering and management* Singapore. Springer Nature.

Chen, Y., & Liu, J. (2019). Distributed community detection over blockchain networks based on structural entropy. In *Proceedings of the 2019 ACM international symposium on blockchain and secure critical infrastructure*.

Cheng, Y., et al. (2024). Gnn-cl: Advanced financial fraud detection using gnn-cl model. arXiv:2407.06529

Crincoli, G., et al. (2022). Vulnerable smart contract detection by means of model checking. In *ACM international symposium on blockchain and secure critical infrastructure*. Proceedings of the Fourth.

Duan, Y., et al. (2024). Cat-gnn: Enhancing credit card fraud detection via causal temporal graph neural networks. arXiv:2402.14708

Far, S.B., Asaar, M. R., & HaghbinA. (2023). A privacy-preserving framework for blockchain-based multi-level marketing. *Computers & Industrial Engineering*, *177*, 109095.

Feng, Y., Sisodia, D., & Li, J. (2020). Poster: Content-agnostic identification of crypto jacking in network traffic. In *Proceedings of the 15th ACM Asia conference on computer and communications security*.

Gao, J., et al. (2022). Supply chain equilibrium on a game theory-incentivized blockchain network. *Journal of Industrial Information Integration*, *26*, 100288.

Grobys, K., King, T., & Sapkota, N. (2022). A fractal view on losses attributable to scams in the market for initial coin offerings. *Journal of Risk and Financial Management*, *15*, 579.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, *30*, 1024–1034.

Hedberg, E.C., & Ayers, S. (2015). The power of a paired *t*-test with a covariate. *Social Science Research*, *50*, 277–291.

Huang, X., et al. (2024). Gnns as adapters for llms on text-attributed graphs. In *The web conference 2024*.

Jung, E., et al. (2019). Data mining-based ethereum fraud detection. IEEE.

Kim, B.G., & WongD. (2023). Did-based distributed verifiable random function with successor rule-based de bruijn sequence in blockchain. In *Proceedings of the 2023 6th international conference on blockchain technology and applications*.

Kirkland, J., et al. (2024). Automated detection of crypto ransomware using machine learning and file entropy analysis.

Kumar, S., et al. (2016). Edge weight prediction in weighted signed networks. In *IEEE 16th international conference on data mining (icdm)*. IEEE.

Li, J., et al. (2024). Blockchain-based public auditing with deep reinforcement learning for cloud storage. *Expert Systems with Applications*, *242*, 122764.

Li, S., et al. (2022). Blockchain-based transparent integrity auditing and encrypted deduplication for cloud storage. *IEEE Transactions on Services Computing*, *16*(1), 134–146.

Li, Y., et al. (2007). Network anomaly detection based on tcm-knn algorithm. In *Proceedings of the 2nd ACM symposium on information, computer and communications security*.

Li, Y., et al. (2021). Blockchain-as-a-service powered knowledge graph construction. In *Knowledge science, engineering and management: 14th international conference* Tokyo, Japan. Springer International Publishing. Proceedings, Part III 14.

Liu, J., et al. (2024a). Fishing for fraudsters: Uncovering ethereum phishing gangs with blockchain data. *IEEE Transactions on Information Forensics and Security*, *19*, 3038–3050.

Liu, J., et al. (2024b). Fishing for fraudsters: Uncovering ethereum phishing gangs with blockchain data. *IEEE Transactions on Information Forensics and Security*, *19*, 3038–3050.

MishraP., et al. (2019). Application of student's *t*-test, analysis of variance, and covariance. *Annals of Cardiac Anaesthesia*, *22*, 407–411.

Morgia, L., et al. (2023). A game of nfts: Characterizing nft wash trading in the ethereum blockchain. In *2023 IEEE 43rd international conference on distributed computing systems (icdcs)*. IEEE.

Phua, K. et al. (2022). Don't Trust, Verify: The Economics of Scams in Initial Coin Offerings.

Ramkumar, G., Mohanavel, V., Tamilselvi, M., Vijayashanthi, R.S., & Amruthavalli, P. (2023). Development of a robust stock market prediction mechanism based on enhanced comprehensive learning principles. In *2023 International conference on research methodologies in knowledge management* (pp. 1–8). IEEE.

Sakurai, A., & Shudo, K. (2024). Tie-breaking rule based on partial proof of work in a blockchain.

Shafiq, O. (2019). Bitcoin Transactions Data 2011–2013.

Shamsi, K., et al. (2022). Chartalist: Labeled graph datasets for utxo and account-based blockchains. *Advances in Neural Information Processing Systems*, *35*, 34926–34939.

Sun, H., et al. (2024). Adaptive attention-based graph representation learning to detect phishing accounts on the ethereum blockchain. *IEEE Transactions on Network Science and Engineering*, *11*, 2963–2975.

Tripathi, G., Ahad, M.A., & Casalino, G. (2023). A comprehensive review of blockchain technology: underlying principles and historical background with future challenges. *Decision Analytics Journal*, *9*, 100344.

Victor, F., & Weintraud, A.M. (2021). Detecting and quantifying wash trading on decentralized cryptocurrency exchanges. In *Proceedings of the web conference*

Wijayanto, A., Sugiharto, A., & Santoso, R. (2024). Detection model for potential flooding areas using k-means and local outlier factor (lof). In *2024 4th international conference of science and information technology in smart administration (icsintesa)* (pp. 445–450).

Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., & Achan, K. (2020). Inductive representation learning on temporal graphs. International Conference on Learning Representations.

YuanL.P., Liu, P., & ZhuS. (2021). Recompose event sequences vs. predict next events: A novel anomaly detection approach for discrete event logs. In *ACM Asia conference on computer and communications security*. Proceedings of the 2021.

Zhang, H., et al. (2021). Blockchain-based privacy-preserving medical data sharing scheme using federated learning. In *Knowledge science, engineering and management: 14th international conference* Tokyo, Japan. Springer International Publishing. Proceedings, Part III 14.

Zou, Y., et al. (2024). A survey of fault tolerant consensus in wireless networks. *High-Confidence Computing*, *4*, 100202.