Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.061964

ARTICLE

HNND: Hybrid Neural Network Detection for Blockchain Abnormal Transaction Behaviors

Jiling Wan, Lifeng Cao*, Jinlong Bai, Jinhui Li and Xuehui Du

Henan Province Key Laboratory of Information Security, Information Engineering University, Zhengzhou, 450000, China

*Corresponding Author: Lifeng Cao. Email: caolf302@sina.com

Received: 06 December 2024; Accepted: 04 March 2025; Published: 19 May 2025

ABSTRACT: Blockchain platforms with the unique characteristics of anonymity, decentralization, and transparency of their transactions, which are faced with abnormal activities such as money laundering, phishing scams, and fraudulent behavior, posing a serious threat to account asset security. For these potential security risks, this paper proposes a hybrid neural network detection method (HNND) that learns multiple types of account features and enhances fusion information among them to effectively detect abnormal transaction behaviors in the blockchain. In HNND, the Temporal Transaction Graph Attention Network (T2GAT) is first designed to learn biased aggregation representation of multi-attribute transactions among nodes, which can capture key temporal information from node neighborhood transactions. Then, the Graph Convolutional Network (GCN) is adopted which captures abstract structural features of the transaction network. Further, the Stacked Denoising Autoencode (SDA) is developed to achieve adaptive fusion of thses features from different modules. Moreover, the SDA enhances robustness and generalization ability of node representation, leading to higher binary classification accuracy in detecting abnormal behaviors of blockchain accounts. Evaluations on a real-world abnormal transaction dataset demonstrate great advantages of the proposed HNND method over other compared methods.

KEYWORDS: Blockchain security; abnormal transaction detection; network representation learning; hybrid neural network

1 Introduction

Blockchain is a distributed encrypted ledger composed of consensus mechanisms, smart contracts and other techniques, which provides a secure platform for transactions among non-trusting participants [1–5]. In the ongoing digital transformation, blockchain technology, characterized by its inherent decentralization, transparency, and cryptographic security [6,7], is increasingly recognized as a cornerstone for next-generation digital infrastructures [8]. In recent years, blockchain technology has been widely and extensively used in areas such as finance [9], healthcare [10–12] and the internet of things [13,14]. In which, virtual digital cryptocurrencies have emerged in large numbers and entered the financial market, making it possible for users to conduct P2P anonymous transactions and securely store assets in a distributed manner. However, the substantial market value of blockchain cryptocurrencies has attracted the attention of malicious actors, leading to attacks that pose potential security risks to the blockchain ecosystem.

In recent years, an increasing number of criminals have exploited the anonymity of blockchain to conduct illegal activities in internet and financial domains [15]. Deceptive practices have proliferated in blockchain networks, as attackers lure users into investing in Ponzi schemes by promising attractive future

returns. At present, a large number of nefarious blockchain accounts have been active and engaged in money laundering, phishing, and other unlawful behaviors [16,17]. According to a report from the SAFEIS Security Research Institute, the blockchain field experienced numerous security incidents in 2022 that caused significant economic damage on a global scale. These incidents included vulnerability exploits, data breaches, phishing attacks, and price manipulation, resulting in total losses exceeding $75.3 billion. These security concerns not only resulted in great financial setbacks for investors but also impeded the adoption and advancement of blockchain technology. Therefore, it is imperative to identify anomalous transactional behaviors in the blockchain network environment to ensure its regular and stable operation [18,19].

Blockchain technology employs a distributed ledger to publicly record transactions, ensuring transparency and easy access to data. The extensive availability of these data records offers sufficient samples for the detection of abnormal transaction behaviors. However, the large and anonymous user base poses challenges for anomaly detection. Hand-engineered features from the complex transaction network may overlook important information, such as temporal patterns and node interactions, thereby limiting the ability to accurately identify transaction behaviors. To address these limitations, network embedding-based detection methods have been developed. These methods combine various neural network modules to capture temporal and structural transaction features of malicious nodes [20]. Despite performing well when used individually, the integration among different modules sometimes fails to achieve optimal information sharing, which can instead lead to a decrease in detection accuracy. Therefore, by improving feature information fusion strategies, it is possible to more effectively uncover latent patterns within complex transaction networks, thereby enhancing the capability to identify abnormal transaction behaviors.

To overcome the above difficulties, this paper proposes a hybrid neural network detection method (HNND) with T2GAT, GCN and SDA to learn and optimize multiple types of node features to effectively detect abnormal transaction behaviors on the blockchain. Specifically, a T2GAT is devised to fully capture temporal information from the records of blockchain historical transactions, GCN is employed to learn network structure information, and SDA is proposed for adaptive fusion of different levels of features and improvement of node representation ability. These components collectively enhance the accuracy of abnormal transaction behavior detection on the blockchain. The main contributions of this paper are as follows:

1. A novel hybrid neural network detection method is developed to enhence blockchain abnormal transaction detection by learning node representations, such as general statistical features (GSFs), temporal transaction features (TTFs), and abstract structural features (ASFs).

2. SDA generates more robust embedding representations through the adaptive fusion of features learned from different modules to improve overall detection performance of the model.

3. Some extensive experiments on real-world blockchain abnormal transaction dataset are conducted and the experimental results show that HNND outperforms state-of-the-art methods on multiple evaluation metrics.

The remaining parts of this paper are organized as follows. Some related works are reviewed in Section 2. Section 3 presentes the HNND method in detail. Section 4 provides the details of the experiments followed by the conclusion in Section 5.

2 Related Work

Money laundering, fraud, and other illicit transactions on blockchain pose significant challenges and risks to the stable operation of these systems. Therefore, it is imperative to promptly identify such malicious transactions and accounts. For the abnormal transaction behaviors detection problem on Blockchain, there

are two main categories of existing methods: machine learning-based anomaly detection methods and network embedding-based anomaly detection methods.

2.1 Traditional Machine Learning-Base Blockchain Anomaly Detection Methods

Traditional machine learning-based blockchain anomaly detection methods automatically learn abnormal patterns in blockchain networks by leveraging datasets consisting of hand-engineered features. Researchers have extracted indicative features of transaction behaviors to judge whether blockchain exhibits abnormal activities. Farrugia et al. pointed out the three most influential features for detecting illicit Ethereum accounts [21] and Ibrahim et al. used attribute association evaluation to select six key transaction features from historical transactions [22]. These features include transaction amounts, balances, durations of activities, among others [23]. Moreover, Kumar et al. additionally developed a set of 18 features for smart contract accounts concerning contract creation and invocation [24].

From the perspective of graph neighborhood, Jourdan et al. leveraged discrete-time graphs to capture temporal features, centrality measures, and other transaction patterns of anonymous Bitcoin accounts [25]. Wu et al. introduced the concept of Attribute-Temporal Heterogeneous (ATH) motifs, and demonstrated the significance of ATH motif features in identifying transactions associated with Bitcoin mixing services [16]. Chen et al. captured the cascade statistical features of nodes and higher-order neighbors based on transaction graphs [26]. These transaction graph-based method consequently augmenting its capacity to detect illicit transaction nodes. However, conventional feature extraction methods are unable to uncover the deeper information on blockchain transaction network.

2.2 Network Embedding-Base Blockchain Anomaly Detection Methods

Blockchain anomaly detection applies some network embedding methods, such as Deepwalk [27], Node2Vec [28] and Graph Convolutional Networks (GCN) [29] to mine latent abstract features. These methods can capture network structural and attribute information from blockchain transaction graphs by transforming network structures into low-dimensional spaces to detect anomalous transaction behaviors. Yuan et al. proposed Node2Vec to uncover abnormal transaction activities [30]. It mainly defined an adjustable random walk strategy to obtain neighborhood vector of nodes. Wu et al. designed Trans2Vec, building upon Node2Vec, wherein the sampling process is not random but biased according to the most recent transaction between two nodes [31].

Chen et al. designed a graph neural network method called E-GCN, to detect anomalous account nodes [32]. The E-GCN learned structural features of transaction networks by drawing on techniques from VGAE [33] and DOMINANT [34]. Moreover, Sun et al. proposed the LSTM Transaction Tree Classifier (LSTM-TC) to identify mixed coin transactions in Bitcoin [35]. They extracted the temporal behavior features of transactions by constructing transaction trees. Some methods learned temporal behavior patterns in transaction sequences to enhance edge representation of nodes, utilizing an LSTM model [36] and a multi-head self-attention mechanism [17]. Specifically, Wang et al. employed time function encoding to capture periodicity features, enabling them to precisely discern the activity cycles and behavioral patterns of anomalous accounts [17]. Nonetheless, there have been few methods that learn graph-based node embedding of transaction networks from multiple angles to capture both temporal transaction information and network structural details. Meanwhile, existing methods inadequately address the issue of noise interference in the concatenation of embedding vectors across different levels.

Therefore, the method proposed in this paper facilitates the simultaneous learning of both node embedding and graph embedding for account transaction graphs. And it accounts for noise effects arising

from the concatenation of features at distinct hierarchical levels. Our method can improve precision and robustness in blockchain anomaly transaction detection.

3 The HNND Method

The blockchain anomaly detection method proposed in this paper employs a hybrid neural network to simultaneously learn diverse embedding representations that contain both temporal transaction patterns and network structural features. In the process of multi-level feature fusion, this method improves the generalization ability of the detection model by adaptively fusing features and adding noise interference to enhance feature robustness. This section introduces the proposed HNND method and analyzes from several aspects, including the overall framework, problem definition and data processing, features extraction (including GSFs, TTFs and ASFs), multi-feature fusion, and binary classification detection.

3.1 The HNND Framework

The hybrid neural network detection (HNND) framework for blockchain abnormal transaction behavior is presented in Fig. 1, which integrates the T2GAT, GCN, and SDA modules. HNND first constructs a temporal transaction multi-directed graph based on historical blockchain transaction data and randomly samples a certain number of subgraphs. It extracts general statistical features (GSFs), temporal transaction features (TTFs), and abstract structural features (ASFs) from each subgraph to judge whether an address exhibits abnormal behavior. In Fig. 1, the following content is mainly involved.

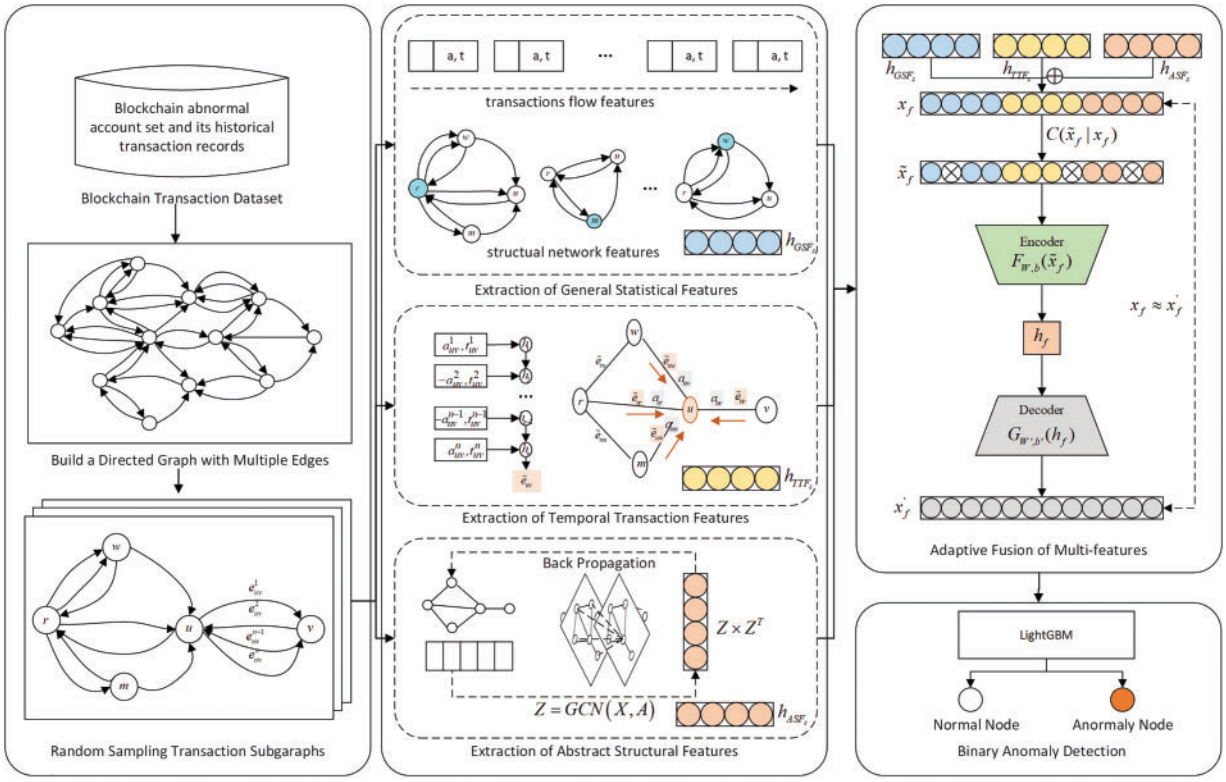

Figure 1: The HNND framework

- Data processing and sampling: It involves constructing and sampling transaction subgraphs based on blockchain transaction records, which reduces the difficulty and computational cost of learning large-scale networks.
- Extraction of GSFs: Aggregate functions are used for statistical analysis to extract four categories of statistical features: transaction features of nodes, basic structural features, regional features, and neighborhood features.
- Extraction of TTFs: The T2GAT model is utilized to learn the temporal transaction features of edges and bias-aggregate these features into the embedded representation of nodes.
- Extraction of ASFs: The GCN is employed to learn graph embedding representation vectors from the transaction subgraphs of nodes, which constitutes the abstract structural features.
- Multi-feature fusion: In the process of concatenating GSFs, TTFs, and ASFs, feature selection and weight allocation are automatically performed by SDA to eliminate data noise and learn key feature information at different levels.
- Binary classification detection: The HNND method uses a binary classifier to train the learned node representations to detect anomalies in the blockchain network.

3.2 Problem Definition and Data Processing

3.2.1 Problem definition

Based on the blockchain transaction records, a directed multi-edge transaction graph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, C)$ is constructed. $\mathscr{V} = \{v_1, \ldots, v_{N_v}\}$ represents the node set of blockchain addresses, where $\mathscr{V}$ is a $N_v \times d_v$ matrix, N_v denotes the number of nodes, d_v denotes the feature dimension of nodes. $\mathscr{E} = \{\varepsilon_1, \ldots, \varepsilon_L\}$ is the transaction set between addresses, where $\mathscr{E}$ represents the edge set of all transactions between each pair of nodes, and each edge has some attributes including transaction amount, time, and direction, etc. $\mathscr{E}$ is a $N_e \times d_e$ matrix, where N_e denotes the number of edges, and d_e denotes the size of the feature space of transaction edges. Since $\mathscr{E}$ can capture the relationship between different nodes, the adjacency matrix $\mathrm{A} \in \mathbb{R}^{N_v \times N_v}$ generated by matrix $\mathscr{E}$ can be employed to represent the correlation between addresses. Specifically, if there is a transaction between node v_i and node v_j, then $v_{ij} = 1$. Conversely, if there is no transaction between node v_i and node v_j, then $A_{ij} = 0$. $C \in \mathbb{R}^{N_v \times N_v}$ represents the corresponding labels of addresses. This paper aims to effectively learn node representations from a large-scale spatiotemporal transaction network graph, so it is necessary to perform embedding representation learning for all nodes. The embedding space for the nodes is denoted as $X_{\mathscr{E}} \in \mathbb{R}^{N_v \times d}$, where d represents the dimensionality of the feature representation.

3.2.2 Data Processing

Based on the acquired large amount of historical blockchain transaction data, a large-scale blockchain transaction network, namely MultiDiGraph (a directed graph with multiple edges), is constructed and continuously updated. In this transaction graph, nodes represent accounts, and edges represent transactions. There can be multiple edges between account nodes, and each edge carries attribute information related to the transaction, such as direction, timestamp, and amount.

To preserve the structural information of the graph while effectively reducing the complexity of learning large-scale networks, this paper adopts a random walk strategy to traverse neighbor relationships and extract sampled subgraphs with the same structural features. In this process, the MultiDiGraph is treated as a single-weighted undirected graph. A node in the graph is randomly selected as the starting node, and any of its neighboring nodes is reached with a certain probability until a subgraph of a fixed size is formed. This process is repeated to obtain a dataset of learnable sampled subgraphs.

3.3 *Hand-Engineered Extraction of GSFs*

After the construction of the transaction graph, node feature information is further extracted from the subgraphs of transactions. While preserving the original account information to minimize information loss, it lays a foundation for subsequent neural network learning to derive multiple types of features. Due to the anonymous characteristic of blockchain accounts, which precludes direct access to any profiling information about the nodes, our method mainly relies on the basic properties of nodes and their transaction edges within the graph to build account features. Meanwhile, the GSFs for the blockchain accounts are derived by various aggregation functions, effectively characterizing account behavioral patterns. The GSFs are outlined in Table 1.

Table 1: General statistical features (GSFs)

Categories	Num	Name	Symbol and calculation formula
Transaction features	FT1	sum_tx_amount	$Tx^{a}_{\text{sum}}(u) = \text{sum}(a_u)$
	FT2	max_tx_amount	$Tx^{a}_{\text{max}}(u) = \max(a_u)$
	FT3	min_tx_amount	$Tx^{a}_{\text{min}}(u) = \min(a_u)$
	FT4	avg_tx_amount	$Tx^{a}_{\text{avg}}(u) = \text{avg}(a_u)$
	FT5	std_tx_amount	$Tx^{a}_{\text{std}}(u) = \sigma(a_u)$
	FT6	entropy_tx_amount	$Tx^{a}_{H}(u) = H(a_u)$
	FT7	sum_tx_time_interval	$Tx^{t}_{\text{sum}}(u) = \text{sum}(t_u)$
	FT8	max_tx_time_interval	$Tx^{t}_{\text{max}}(u) = \max(t_u)$
	FT9	min_tx_time_interval	$Tx^{t}_{\text{min}}(u) = \min(t_u)$
	FT10	avg_tx_time_interval	$Tx^{t}_{\text{avg}}(u) = \text{avg}(t_u)$
	FT11	std_tx_time_interval	$Tx^{t}_{\text{std}}(u) = \sigma(t_u)$
	FT12	entropy_tx_time_interval	$Tx^{t}_{H}(u) = H(t_u)$
Structural features	FT13	in_degree	$D_{\text{in}}(u) = \lvert\{e_{vu} \mid v \in N_u\}\rvert$
	FT14	out_degree	$D_{\text{out}}(u) = \lvert\{e_{uv} \mid v \in N_u\}\rvert$
	FT15	total_degree	$D_{\text{total}}(u) = D_{\text{in}}(u) + D_{\text{out}}(u)$
Regional features	FT16	in_degree_ego_subgraph	$D_{\text{in}}(R_u) = \lvert\{e_{wu} \in E \mid w \notin V_u, v \in V_u\}\rvert$
	FT17	out_degree_ego_subgraph	$D_{\text{out}}(R_u) = \lvert\{e_{wu} \in E \mid w \in V_u, v \notin V_u\}\rvert$
	FT18	total_degree_ego_subgraph	$D_{\text{total}}(R_u) = D_{\text{in}}(R_u) + D_{\text{out}}(R_u)$
Neighborhood features	FT19	sum_in_degree_first_order	$D_{\text{sum\_in}}(N_u) = \{\text{sum}(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT20	max_in_degree_first_order	$D_{\text{max\_in}}(N_u) = \{\max(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT21	min_in_degree_first_order	$D_{\text{min\_in}}(N_u) = \{\min(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT22	avg_in_degree_first_order	$D_{\text{avg\_in}}(N_u) = \{\text{avg}(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT23	std_in_degree_first_order	$D_{\text{std\_in}}(N_u) = \{\sigma(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT24	entropy_in_degree_first_order	$D_{\text{entropy\_in}}(N_u) = \{H(D_{\text{in}}(v)) \mid v \in N_u\}$
	FT25	sum_out_degree_first_order	$D_{\text{sum\_out}}(N_u) = \{\text{sum}(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT26	max_out_degree_first_order	$D_{\text{max\_out}}(N_u) = \{\max(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT27	mout_out_degree_first_order	$D_{\text{min\_out}}(N_u) = \{\min(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT28	avg_out_degree_first_order	$D_{\text{avg\_out}}(N_u) = \{\text{avg}(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT29	std_out_degree_first_order	$D_{\text{std\_out}}(N_u) = \{\sigma(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT30	entropy_out_degree_first_order	$D_{\text{entropy\_out}}(N_u) = \{H(D_{\text{out}}(v)) \mid v \in N_u\}$
	FT31	sum_total_degree_first_order	$D_{\text{sum\_total}}(N_u) = \{\text{sum}(D_{\text{total}}(v)) \mid v \in N_u\}$
	FT32	max_total_degree_first_order	$D_{\text{max\_total}}(N_u) = \{\max(D_{\text{total}}(v)) \mid v \in N_u\}$
	FT33	min_total_degree_first_order	$D_{\text{min\_total}}(N_u) = \{\min(D_{\text{total}}(v)) \mid v \in N_u\}$
	FT34	avg_total_degree_first_order	$D_{\text{avg\_total}}(N_u) = \{\text{avg}(D_{\text{total}}(v)) \mid v \in N_u\}$
	FT35	std_total_degree_first_order	$D_{\text{std\_total}}(N_u) = \{\sigma(D_{\text{total}}(v)) \mid v \in N_u\}$
	FT36	entropy_total_degree_first_order	$D_{\text{entropy\_total}}(N_u) = \{H(D_{\text{total}}(v)) \mid v \in N_u\}$

The GSFs are extacted from basic attribute information such as transaction amounts and timestamps of accounts, as well as various structural information between graph nodes, by employing a set of aggregate functions. The aggregate functions set Z is showed in Formula (1). The GSFs can be classified into four

categories: transaction features, structural features, regional features, and neighborhood features.

$$Z = \{\text{sum}(x), \max(x), \min(x), \text{avg}(x), \sigma(x), H(x)\} \tag{1}$$

FT1~FT12: Transaction features. Transaction features are mainly related to transaction amounts and timestamps, and can reflect the intensity and scale of account fund flows. Each transaction e_{uv} between node u and node v has features such as amount, time interval, and direction. In which, the direction of the transaction flow indicates different transaction features, e.g., divide the transaction amount a_u of node u into received amount a_{vu} or sent amount a_{uv}. Finally, based on the aggregation function Z, statistical analysis is performed on the transaction amounts and timestamps of any given node u. And the transaction features $Tx_a(u)$ and $Tx_t(u)$ for the node u are obtained.

FT13~FT15: Structural features. Structural features include the in-degree, out-degree, and total-degree of nodes, which reflect the connection pattern and interaction relationship of accounts in the transaction graph. Multiple transaction edges exist between account nodes, $|e_{vu}|$ represents the number of transactions between node u and node v, and N_u denotes the set of first-order neighbors of node u. The structural features of node u are represented by the symbol $D(u)$.

FT16~FT18: Regional features. The ego network $R_u = (V_u, E_u)$ of node u consists of node u and its first-order neighbors $V_u = u \cup N_u$, and all edges are between these nodes. Regional features including the internal region information are obtained from the ego network of nodes, which reflect the spatial distribution and aggregation of account spaces. Based on the definition of the ego network, the regional features of node u are represented by the symbol $D(R_u)$.

FT19~FT36: Neighborhood features. These features obtain the external neighborhood information from neighboring nodes, representing the interaction between account neighbors. Finally, the aggregation function Z is used to calculate neighborhood features $D(N_u)$.

To sum up, the GSFs of network nodes are expressed as

$$GSFs = \left\{ Tx_z^{\text{a}}(u), Tx_z^{t}(u), D_z(u), D_z(R_u), D_z(N_u) \mid \text{z} \in Z \right\}. \tag{2}$$

3.4 Extraction of TTFs by T2GAT

The T2GAT is designed to learn the information of transaction sequences sorted by timestamps and interactions among all neighboring transactions of nodes. As shown in Fig. 2, a temporal sequence neural network model GRU is first utilized to learn the temporal relationships of all transaction sequences between each pair of nodes from which the edge representation vectors are obtained. The representation of the edges between the node and all its adjacent nodes is then learned in a weighted manner by using a Graph Attention Network (GAT) to obtain the node representation for the account.

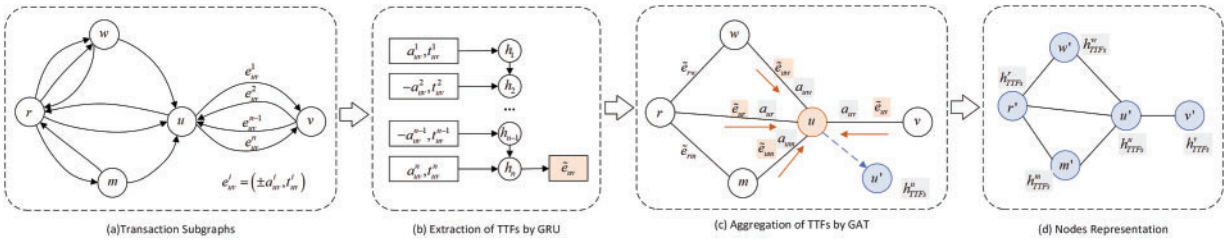

Figure 2: Extraction and aggregation of TTFs

3.4.1 Extraction of Edge Representations

To fully capture the temporal relationship and attribute information of account transactions, this paper introduces basic features such as transaction amount and direction as attributes of edges in the graph. Moreover, the GRU model is adopted to capture the behavioral patterns of multi-attribute transactions and obtain temporal transaction features.

The GRU model can address the issues of long-term dependency and gradient vanishing in Recurrent Neural Networks (RNNs) by introducing two gating mechanisms: the reset gate and the update gate [37]. Compared to LSTM, GRU has lower computational complexity and faster convergence speed. The basic structure of GRU consists of two gating units (i.e., reset gate and update gate) and a hidden state unit. Specifically, the reset gate r_t determines whether the previous hidden state unit h_{t-1} is ignored, which controls the extent to which previous state information is disregarded. While the update gate z_t determines whether the current hidden state unit $\tilde{h}_t$ needs to be updated with a new hidden state unit, which controls the extent to which current state information is retained. Eqs. (3)–(6) represent the principles of GRU's iterative update, respectively. In particular,

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1} + \mathrm{b}_r\right); \tag{3}$$

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1} + \mathrm{b}_z\right); \tag{4}$$

$$\tilde{h}_t = \mathrm{Tanh}\left(W_h x_t + U_h\left(r_t \odot h_{t-1}\right) + \mathrm{b}_h\right); \tag{5}$$

$$h_t = \left(1 - z_t\right) h_{t-1} + z_t \tilde{h}_t, \tag{6}$$

where W and U are weight matrices, b is the bias term, and σ is the Parametric Linear Unit (PReLU) activation function. It is noteworthy that the PReLU activation function addresses the hard saturation issue of Rectified Linear Unit (ReLU) in the negative interval by introducing a learnable parameter α. The expression for the PReLU is given by $f(x) = \max(0, x) + \alpha \min(0, x)$. In which, the learnable parameter α adjusts the slope of negative inputs to make the function adapt to the optimal negative slope for different features, potentially contributing to better convergence and generalization for anomaly detection. Therefore, the GRU model can not only extract attribute features from transaction data but also capture temporal relationship features between time-series transaction data.

As demonstrated in (a) and (b) of Fig. 2, this paper sorts all transaction edges $\varepsilon_{uv} = \left\{e_{uv}^1, e_{uv}^2, \ldots, e_{uv}^n\right\}$ of a node pair (u, v) by timestamp. And then these sorted edges are input into the GRU model to learn the temporal transaction information between the node pair (u, v), i.e., which is represented as edge representation vectors $\tilde{e}_{uv}$ of the node pair (u, v). Each transaction edge has three attributes: amount, time interval, and direction, with the directionality of the transaction represented by positive or negative sign.

3.4.2 Biased Aggregation of Edge Representations

Learning the temporal transaction features of nodes essentially involves aggregating all neighbor edge representation vectors of each node. To more effectively capture interaction relationships between nodes, this paper employs a Graph Attention Network (GAT) augmented with a multi-head attention mechanism. The GAT can facilitate biased learning of node representations, thereby focusing on significant interactions within the graph [29]. In this paper, the GAT model is utilized to dynamically assign different attention weights based on the importance of edge representations between nodes (as shown in (c) of Fig. 2). And the attention coefficients are employed to transform all the edge representation of a node into attribute features of that node (as shown in (d) of Fig. 2).

The attention coefficient of the edge representation between the node u and its neighbor node i is calculated and normalized as

$$a_{ui} = \frac{\exp\left(\mathrm{ReLU}\left(\beta^{\mathrm{T}}\left(Wh_{\tilde{\varepsilon}_u} \| Wh_{\tilde{\varepsilon}_i}\right)\right)\right)}{\sum_{k\in N_u}\exp\left(\mathrm{ReLU}\left(\beta^{\mathrm{T}}\left(Wh_{\tilde{\varepsilon}_u} \| Wh_{\tilde{\varepsilon}_k}\right)\right)\right)}, \tag{7}$$

where W represents the weight matrix, "$\|$" denotes the concatenation operation, β is a learnable parameter, N_u stands for the set of neighboring nodes of node u, and $\tilde{\varepsilon}_u = \{\tilde{e}_{uv}, \tilde{e}_{uw}, \ldots, \tilde{e}_{ur}\}$ denote all adjacent edges for node u. After performing a dot product operation between the concatenated vector $h_{\tilde{\varepsilon}_u}$ and β^T, a normalization is applied using the softmax function.

To enhance the model's robustness, the GAT employs a multi-head attention mechanism that integrates the outputs from several independent single-head attention processes. Each head computes a weighted sum of the edge representations for a node based on its attention coefficients. By averaging the outputs across all heads, the temporal transaction features h_{TTFs} of nodes is obtained by

$$h^u_{TTFs} = \frac{1}{K}\sum_{k=1}^{K}\sigma\left(\sum_{i\in N_u} a^{(k)}_{ui} W^{(k)} h_{\tilde{e}_{ui}}\right), \tag{8}$$

where K represents the number of attention heads in the GAT.

3.5 Extraction of ASFs by Improved-GCN

Abstract structural features (ASFs) are obtained by reconstructing the transaction graph structure using an improved Graph Convolutional Network (GCN). The extraction method and included information for the ASFs differ from those for the structural features in the GSFs. The structural features are extracted by aggregation functions to obtain structural information from multiple aspects. But the ASFs are based on the potential structural features extracted by the enhanced neural network GCN, which can capture higher-order neighbor features and underlying topological relationships between nodes.

The GCN model performs convolutional operations in the spectral domain, where each operation can aggregate an additional layer of features. The spectral convolution function is represented as

$$H^{(l+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \tag{9}$$

where $\tilde{A} = A + I_N$ is the adjacency matrix A augmented with self-loops, where I_N is the identity matrix, ensuring that each node also considers its own information. $\tilde{D}$ is the degree matrix of $\tilde{A}$, a diagonal matrix whose elements correspond to the degrees (number of connections) of the nodes in $\tilde{A}$. $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix of A. $H^{(l)}$ denotes the feature matrix of nodes at layer l. $W^{(l)}$ represents the trainable weight matrix used for linearly transforming node features at each layer. σ is ReLU activation function.

The transaction subgraph is fed into the GCN for a single-layer convolution operation $Z = \sigma\left(\widehat{A}XW^{(0)}\right)$ to learn the latent node representations. $\bar{A} = \sigma\left(Z \cdot Z^T\right)$ serves as an approximate estimation of the adjacency matrix A, where Z reconstructs the original graph structure. Anomalous nodes are identified by comparing the similarity between the reconstructed decoder and the original graph structure. The lower the similarity, the higher the likelihood of the node being anomalous. The reconstruction loss function is given by

$$\mathscr{L}_{\mathrm{GCN}} = \frac{\|\bar{A} - A\|_F^2}{n}, \tag{10}$$

where $\|\cdot\|_F^2$ denotes the L2 norm of a vector. By minimizing this loss function, a node representation h_{ASFs} that encapsulates abstract structural features of the transaction graph is learned.

3.6 Multi-feature Fusion by Stacked Denoising Autoencoder

Based on the aforementioned methods, three types of features are learned: general statistical feature vector h_{GSFs} extracted from the original transaction graphs, temporal transaction feature vector h_{TTFs} captured by the T2GAT model, and abstract structural feature vector h_{ASFs} obtained through the GCN model. The resulting feature vector x_f is yielded by directly concatenating the three feature vectors.

$$x_f = \text{contract}\left(h_{GSFs}, h_{TTFs}, h_{ASFs}\right).$$

However, there exists data noise in the process of concatenating feature vectors of various types, and different types of features contain information of varying importance. Therefore, a Stacked Denoising Autoencoder (SDA) is utilized to adaptively integrate features of multiple types. By learning the key information of features at different levels, node representations that are more robust and have enhanced generalization capabilities are obtained. Specifically, during the training of each autoencoder in the SDA, Gaussian noise is injected to enable the network to recover the original signal from noisy data, thus learning more robust features. Meanwhile, multiple denoising autoencoders are stacked in a hierarchical structure for layered learning. An adaptive fusion of feature vectors of various types is achieved by automatically allocating weights and selecting key information from features at different levels. Each denoising autoencoder (DA) within the SDA consists of an encoder and a decoder, which are presented as

$$h_f = F_{W,b}\left(\tilde{x}_f\right) = \varsigma\left(W\tilde{x}_f + b\right); \tag{11}$$

$$x'_f = G_{W',b'}\left(h_f\right) = W'h_f + b', \tag{12}$$

where W and W' are the weight matrices, b and b' are the biases, while the noise feature vector $\tilde{x}_f$ is derived from the introduction of a corruption process $C\left(\tilde{x}_f \mid x_f\right)$ into the input feature vector. By adding specific forms of noise or perturbations to the input data, the robustness and generalization capability of the model can be enhanced. ς denotes the non-linear activation function LeakyReLU. Compared to ReLU, the improved activation function LeakyReLU allows negative inputs to have a small positive gradient, so these negative values can be better utilized to enhance the model's performance.

The noisy feature vector a is obtained by introducing a corruption process c to the input feature vector. By adding specific forms of noise or perturbations to the input data, the robustness and generalization capability of the model can be enhanced. ς denotes the non-linear activation function LeakyReLU. Compared to the traditional ReLU activation function, the improved LeakyReLU activation function allows negative inputs to have a small positive gradient. Therefore, it can more effectively utilize negative value information, thereby contributing to improved model performance.

Each DA approximates the reconstruction by minimizing the reconstruction error $\mathscr{L}_{\text{DA}}\left(x_f, x'_f\right)$, which aims to replicate an output vector similar to the input feature vector x'_f. As presented in Fig. 3, the SDA consisting of two cascaded DA layers is designed to learn more robust node representations. This approach preserves the useful features at different levels to improve the robustness of the fused features.

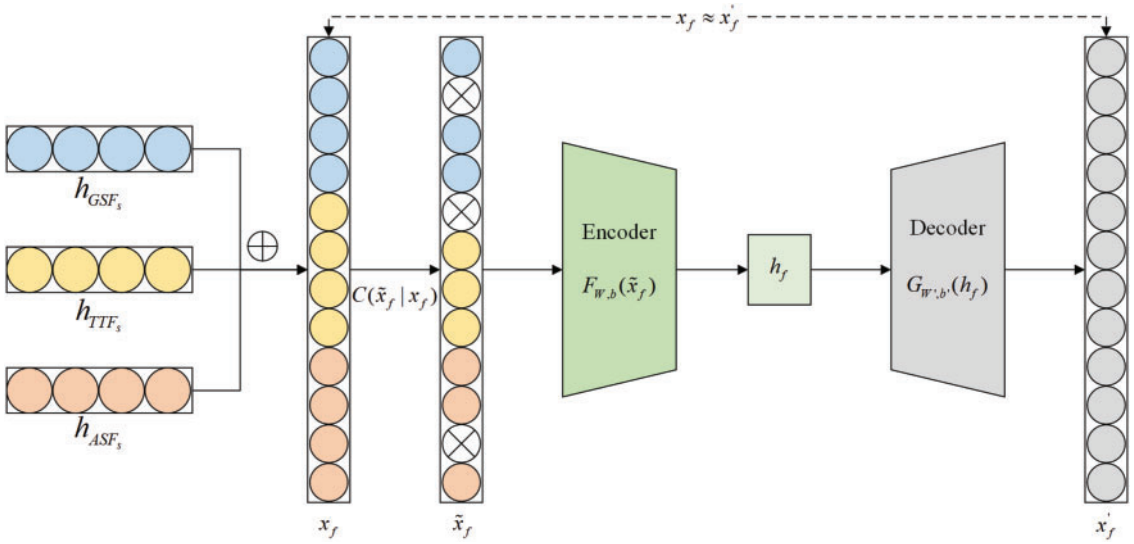

Figure 3: Adaptive fusion of multi-features

3.7 Binary Classification Detection

This study aims to construct a hybrid neural network based on T2GAT, GCN, and SDA, to learn temporal sequence relationships, abstract structural features, and critical information from different types of features. It facilitates obtaining a more comprehensive and complete node representation to improve the accuracy of the binary classifier LightGBM in detecting abnormal blockchain accounts. The detailed procedure of the proposed HNND detection method is presented in Algorithm 1.

Algorithm 1: The HNND algorithm

Input: Blockchain transaction set of addresses $\mathcal{T}_{\text{addrs}}$
Output: The prediction results of the algorithm Res_{class}
1: $\mathcal{G}(\mathcal{V}, \mathcal{E}) \leftarrow$ empty graph; $\mathcal{V} \leftarrow$ empty set of nodes; $\mathcal{E} \leftarrow$ empty set of edges;
2: **for** each transaction $e \in \mathcal{T}_{addrs}$ **do**
3: Edge subset $\hat{\mathcal{E}}_{ij} \leftarrow$ directed edge $e\left(v_i \rightarrow v_j\right)$ with transaction amount and time stamp;
4: Subset $\hat{\varepsilon}_{ij}$ of edge set $\mathcal{E}$ in $\mathcal{G}(\mathcal{V}, \mathcal{E}) \leftarrow$ all other edges of the pair nodes $< v_i, v_j >$;
5: Node set $\mathcal{V}$ in $\mathcal{G}(\mathcal{V}, \mathcal{E}) \leftarrow$ unique v_i and v_j;
6: Original features $h_{RF} \leftarrow$ transaction amount, direction, and time stamp of directed edge e;
7: The number of samples N, result $\leftarrow$ [];
8: **for** i in range (N) **do**
9: A random node $v_s \leftarrow \mathcal{V}, k$;
10: $\mathcal{G}' \leftarrow$ empty subgraph; $\mathcal{V}' \leftarrow v_s$;
11: **while** $|\mathcal{V}'| < k+1$ **do**
12: $v_r \leftarrow$ a random neighbor node of v_s in $\mathcal{V}'$;
13: Add v_r to the node set $\mathcal{V}'$ in $\mathcal{G}'(\mathcal{V}', \mathcal{E}')$;
14: Add all directed edges of a pair of nodes $< v_s, v_r >$ to the edge subset $\hat{\varepsilon}_{sr}$ of $\mathcal{E}'$;
15: **for** $\mathcal{G}_u'$ of each node u **do**
16: $Z = \{\text{sum}(x), \max(x), \min(x), \text{avg}(x), \sigma(x), H(x)\}$;

(Continued)

Algorithm 1 (continued)

17: $h_{GSFs} \leftarrow \{Tx_z^a(u), Tx_z^t(u), D_z(u), D_z(R_u), D_z(N_u) \mid z \in Z\}$;
18: $\widetilde{\mathscr{G}} \leftarrow$ empty subgraph;
19: **for** v_u in $\mathscr{V}'$ **do**
20: **for** for each neighbor node v_i of node v_u **do**
21: $\tilde{e}_{ui} \leftarrow$ GRU (all edges ε'_{ui} of $< v_u, v_i >$);
22: $\tilde{\varepsilon}_u \cdot$ add $(\tilde{e}_{ui})$, $\widetilde{\mathscr{V}}_u \cdot$ add (unique v_u and v_i);
23: **for** $\tilde{e}_{ui}$ in $\tilde{\varepsilon}_u$ **do**
24: $a_{ui} \leftarrow Attention\left(\tilde{e}_{ui}, \tilde{\mathscr{E}}_u\right)$;
25: $h'_u \leftarrow ReLU\left(\sum_{i=N\tilde{\varepsilon}_u} a_{ui} W h_u\right)$;
26: $h_{TTFs} \cdot$ add (h'_u);
27: $X \leftarrow$ Original Feature h_{RF}, $A \leftarrow \widetilde{\mathscr{G}}$;
28: $Z \leftarrow \mathrm{GCN}(X, A)$;
29: $\bar{A} \leftarrow \sigma\left(ZZ^T\right)$;
30: $h_{ASFs} \leftarrow \min \mathscr{L}_{\mathrm{GCN}}(A, \bar{A})$;
31: $x_f \leftarrow$ contract $(h_{GSFs}; h_{TTFs}; h_{ASFs})$;
32: $\tilde{x}_f \leftarrow C\left(\tilde{x}_f \mid x_f\right)$;
33: $x'_f \leftarrow \min \mathscr{L}_{\mathrm{SDA}}\left(x_f, \mathrm{SDAE}\left(\tilde{x}_f\right)\right)$;
34: $Res_{class} \leftarrow$ LightGBM $\left(x'_f\right)$;
35: **return** Res_{class}

In Algorithm 1, a blockchain transaction network is first constructed by yielding an account transaction directed graph with multiple edges, where transaction edges have timestamps and monetary values as properties (Lines 1–6). Then, a random walk strategy is utilized to extract sampled subgraphs with a sample size of N, and the size of the subgraphs k is customizable (Lines 7–14). Next, the aggregate function Z is used for feature engineering on the original transaction records to obtain general statistical features of the accounts (Lines 15–18). Furthermore, the T2GAT model is designed to learn node embedding in the transaction network (Lines 19–26). And the GRU model is employed for capturing the temporal relationships and attribute information of all transactions around each node (Lines 20–22). Moreover, the GAT model is adopted to bias-learn the transaction information around each node and transform it into a node embedding vector (Lines 23–25). The abstract structural features of the node transaction graph is learned by using the GCN model (Lines 27–30). Thereafter, the SDA model is developed to adaptively integrate multiple features at different levels and learn more robust features by adding noise (Lines 31–33). Finally, the LightGBM method is employed for anomaly account binary classification detection (Line 34).

4 Experiments

To evaluate the effectiveness of the HNND method, experiments are conducted on a real-world blockchain dataset of abnormal account transactions. The experiment aims at three aspects.

RQ1: Evaluate the effectiveness of the HNND method in detecting abnormal addresses on the blockchain transaction network.

RQ2: Evaluate the contribution of each component of HNND on the final performance of blockchain anomaly detection.

RQ3: Evaluate the robustness of the HNND method by changing transaction sequence lengths and attention sizes.

4.1 Dataset

4.1.1 Data collection

The historical transaction dataset of blockchain fraudulent accounts in [32] is used in the experiment. The dataset is constructed by harvesting fraudulent accounts (central nodes) from the Ethereum tag cloud on the authorized website Etherscan[1]. Then their first-order neighbors, second-order neighbors are extracted, and the transactions among them via APIs provided by the website. The dataset contains 2,973,382 nodes and 13,551,214 edges, including 1157 fraudulent accounts among the central nodes. Due to the large size of the original connected subgraph, a random walk strategy is used to sample subgraph datasets (i.e., Dateset 1, Dataset 2, Dataset 3) with different sizes (i.e., 20000, 30000, 40000). Each subgraph is sampled five times to ensure the validity of the method. The detailed information of experimental data is provided in Table 2.

Table 2: Statistics of experimental datasets

Dataset	Subgraph size	Labeled nodes	Edges	Average degree
Dateset 1	20000	78	839,721	83.9721
Dateset 2	30000	106	1,127,359	75.1573
Dateset 3	40000	143	1,341,266	67.0633

4.1.2 Data Cleaning

For the class imbalance problem in the dataset, this paper eliminates obvious normal addresses to build a more efficient model. Addresses with less than five or more than 1000 transactions are removed because these addresses could be wallets or other normal types of accounts, as confirmed by data analysis. In addition, duplicate account addresses are deleted, and the latest transaction records among them are kept. After data cleaning, the average number of remaining nodes in each subgraph is 18,504, 27,973 and 37,542, respectively. In the course of training the model, 80% of the dataset is used as the training data and the remaining 20% as the testing data.

4.2 Experimental Setup

This section provides a detailed introduction to the baseline methods, evaluation metrics, and implementation details, all of which ensure the accuracy and reliability of the experiments.

4.2.1 Baseline Methods

To illustrate the validity of the HNND method, it is compared with the following blockchain anomaly detection methods.

- Original features [21]: A set of basic account features, including eight features such as degree, strength, etc., and the number of neighboring nodes, which are utilized and directly fed into the LightGBM model.
- Statistical features [16,23,26]: A set of general statistical features (GSFs) is designed and extracted, which involves basic transactional features [23] and structural features [16,26] related to account transactions. The machine learning algorithms are then employed to implement the binary classification task for anomaly detection.
- Node2Vec [30]: Node2Vec is a network representation learning method that simulates proximity relationships between nodes through random walks. It generates a series of node sequences, which

[1] https://etherscan.io, accessed on 3 March 2025.

are then mapped into a low-dimensional vector space through the word embedding model to obtain node embeddings.

- E-GCN [32] and GAT [29]: The two methods perform graph convolutional operations to capture the topological structure information and feature information of nodes in the transaction graph, including the neighbor relations of nodes, the weights of edges, etc. They can generate more expressive node representations for node classification.
- SCSGuard [38] and LSTM [39]: The two methods employ sequence learning neural network to learn key information from temporal transactions data. It can effectively capture behavioral patterns in the transaction records of abnormal accounts, thereby identifying transactions that significantly deviate from normal behaviors.

4.2.2 Evaluation Metrics

The detection of abnormal transaction behaviors in the blockchain is essentially a binary classification problem. Hence, the confusion matrix is utilized to evaluate the accuracy of classification. Three classification metrics that assess model performance are introduced: (1) Precision, (2) Recall, and (3) F1-score to objectively evaluate the performance of the HNND method in detecting abnormal transactions on the blockchain. In which, (1) Precision refers to the proportion of samples predicted by the model as abnormal accounts that are truly abnormal, which reflects the accuracy of the model's prediction of abnormal accounts. (2) Recall indicates the proportion of all actual abnormal accounts that are predicted as abnormal by the model, which demonstrates the model's ability to identify all abnormal accounts. (3) F1-score is the harmonic mean of precision and recall, and it is used to evaluate the model's performance comprehensively.

In the context of blockchain abnormal account detection considered in this paper, the importance of recall outweighs precision. This is because the purpose of the detection model is to identify all abnormal accounts as many as possible, even if it may mistakenly label some normal accounts as abnormal. Since the potential harm caused by abnormal accounts is significant, overlooking some malicious accounts in the detection process could result in large economic and financial losses as well as system disruption. Meanwhile, mislabeling some normal accounts as abnormal may cause inconvenience to users, but this impact is acceptable compared to the consequences of missing abnormal accounts.

4.2.3 Implementation Details

All methods are implemented based on the PyTorch framework in the experiment, with the output dimension of all models set to 8. The walk length, window size, p, and q for the Node2Vec method are set to 20, 4, 0.25, and 0.4, respectively. The hidden layer size and learning rate for E-GCN are set to 3 and 0.001, and the Adam optimizer is chosen for model optimization. Additionally, the number of leaves and the learning rate for the LightGBM model are fixed at 50 and 0.03, respectively.

4.3 Experimental Evaluation

This paper conducts two groups of experiments and analyzes results from two aspects. One group of experiments evaluates the effectiveness of the proposed HNND method by comparing it with various baseline methods. The other group of experiments performs ablation studies by sequentially eliminating each module within HNND to determine the impact of the removed module on the detection results.

4.3.1 Effectiveness Results (RQ1)

The performance of all the comparison methods for blockchain abnormal account transaction behavior detection is evaluated in this subsection. And the results are presented in Table 3. The following conclusions can be drawn.

Table 3: The performance comparison of blockchain anomaly detection methods

Method	Dataset 1			Dataset 2			Dataset 3		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Original features	0.603	0.694	0.645	0.717	0.537	0.614	0.752	0.614	0.676
Statistical features	0.734	0.701	0.717	0.831	0.654	0.732	0.814	0.759	0.786
Node2Vec	0.586	0.673	0.626	0.680	0.697	0.688	0.734	0.659	0.694
E-GCN	0.713	0.635	0.672	0.738	0.679	0.707	0.746	0.684	0.714
GAT	0.738	0.604	0.664	0.747	0.692	0.718	0.708	0.717	0.712
SCSGuard	0.697	0.756	0.725	0.769	0.812	0.790	0.819	0.749	0.782
LSTM	0.766	0.703	0.733	0.836	0.773	0.803	0.732	0.873	0.796
Our HNND	**0.891**	**0.904**	**0.897**	**0.896**	**0.917**	**0.906**	**0.925**	**0.902**	**0.913**

- Our HNND method outperforms all other baseline methods in all three evaluation metrics. The HNND method achieves the best performance, with 92.5% precision, 90.2% recall, and 91.3% F1-score under Dataset 3. Deep learning methods exhibit comparable performance, with E-GCN and GAT achieving an F1-score of approximately 71%, while SCSGuard and LSTM achieving an F1-score of around 78%. Node2Vec demonstrates higher performance than the feature-based method, and its F1-score is 69.4%.
- Compared with the method based on the original features, the HNND method obtains about 23% higher values in all three evaluation metrics. Of all the compared methods, the method based on the original features performs the worst. It can be found that the parties based on the original features do not consider the timing and network structure of the blockchain data, the feature information obtained is very limited, and further data mining of the transaction information is lacking.
- Compared with the method based on statistical features, the F1-score of the HNND method improves by 12.7% on Dataset 3. Although the method based on statistical features fully utilizes the network structure information, it lacks learning from the temporal features of transaction data, so its detection effect is worse than that of the HNND method. Meanwhile, the method based on statistical features compared with the method based on original features increases by F1-score of 11%. The statistical method actually is a machine learning method, and can rely on aggregation functions to obtain statistical features containing structure information so its F1-score reaches 78.6%.
- The F1-score of Node2Vec is about 20% different from that of the HNND method. Node2Vec utilizes random walks to mine the neighborhood information of nodes, but it ignores learning the interaction relationships and transaction temporal information between nodes, leading to incomplete representation learning of nodes. The HNND method learns the feature information that Node2Vec fails to learn, so it exhibits the best detection performance. However, the difference in evaluation metrics between Node2Vec and Original feature-based methods is not significant. Though both can obtain rich network structure information through different methods, Node2Vec's representation learning method can capture potential structures and patterns in the network without the need for manual feature design.
- The detection performance of the network representation method based on deep learning is greatly improved. E-GCN, GAT, SCSGuard, and LSTM all obtain good detection results. Compared with these deep learning methods, the three evaluation metrics of HNND increase by about 10%. Specifically, E-GCN and GAT can capture the topological structure information of the nodes in the transaction graph to

generate a more expressive node representation. SCSGuard and LSTM can provide long-term memory and effectively capture the behavior patterns in the account temporal transaction data. The two types of methods have different focuses on feature extraction, and there is still much room for improvement in learning features. Our HNND method combines the advantages of both E-GCN and GRU methods and utilizes a hybrid neural network for representation learning. It fully exploits the temporal relationship in the transaction sequence and the interaction pattern information between the network nodes, thereby obtaining a more efficient node embedding representation.

4.3.2 Ablation Experiment (RQ2)

This subsection verifies the effectiveness of the TTFs extraction module (HNND/t), the ASFs extraction module (HNND/s), and the SDA module (HNND/d) in the proposed HNND method, respectively, as shown in Fig. 4. The following observations are obtained:

- Relative to HNND, the effectiveness of HNND/t significantly declines, with F1-scores that are 21.5%, 17.7%, and 19.6% less on Datasets 1–3, respectively. The main reason is that the eliminated T2GAT can fully extract the temporal pattern of the transaction interaction between nodes, and the bias aggregation learning can obtain expressive node embedding representation. This result indicates that it is crucial to fully learn the timely order features of transaction attributes among nodes in the transaction graph for the detection of abnormal account behaviors.
- The F1-score of HNND/s is only 9.4% lower than that of HNND on Dataset 3. It can be found that, to some extent, the extracted abstract structure features are not as important as the temporal transaction features, but they can also reflect the information of the topological network and further enrich the representation of nodes.
- The slightly lower performance of HNND/d compared to HNND demonstrates that SDA can improve the detection performance of anomalous transactions by denoising and adaptively integrating features obtained from different modules. The comparison results of the E-GCN and T2GAT modules with HNND/t and HNND/s also support this conclusion. And final fully implemented HNND method can reach F1-score of 91.3% on Dataset 3.
- The complete HNND method is superior to other ablation models in the three evaluation metrics on Dataset 1, Dataset 2, and Dataset 3, respectively, which proves that each model can provide effective improvement and enable HNND to achieve the best performance in the binary classification of abnormal transaction behaviors.

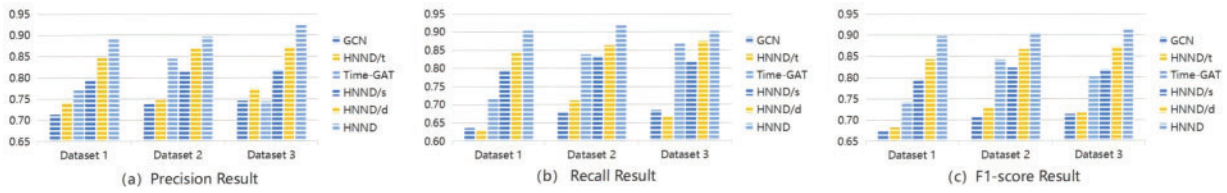

Figure 4: Precision, Recall, and F1-scores of HNND and its variants

4.3.3 Sensitivity Analysis (RQ3)

The robustness of the HNND method is evaluated by investigating its stability and reliability across varying transaction sequence lengths and attention sizes.

Figure 5 presents three metrics scores for the HNND model across varying transaction sequence lengths. The findings can be summarized as follows:

- Up to a certain point, the HNND model demonstrates improved performance on all datasets as the length of sequences fed into the GRU increases. This suggests that within an optimal range, longer sequences provide richer behavioral features.
- Occasionally, shorter sequences yield superior results, as they focus exclusively on critical transaction patterns and avoiding redundant information.
- Model performance deteriorates when sequence lengths exceed a threshold, potentially due to the introduction of noise or overfitting issues from excessively long sequences.
- Beyond approximately 15 time steps, while there are minor fluctuations, the evaluation metrics generally stabilize. This indicates that the model has effectively learned the primary behavioral patterns in the transaction sequences, with minimal gains from further lengthening the sequences.
- According to F1-scores in Fig. 5c, model performance improves with larger dataset sizes, highlighting the benefit of more high-quality data in capturing transaction behavior accurately.

These results underscore the importance of temporal transaction behavior features and demonstrate the robustness and adaptability of the HNND model in handling sequences of varying lengths.

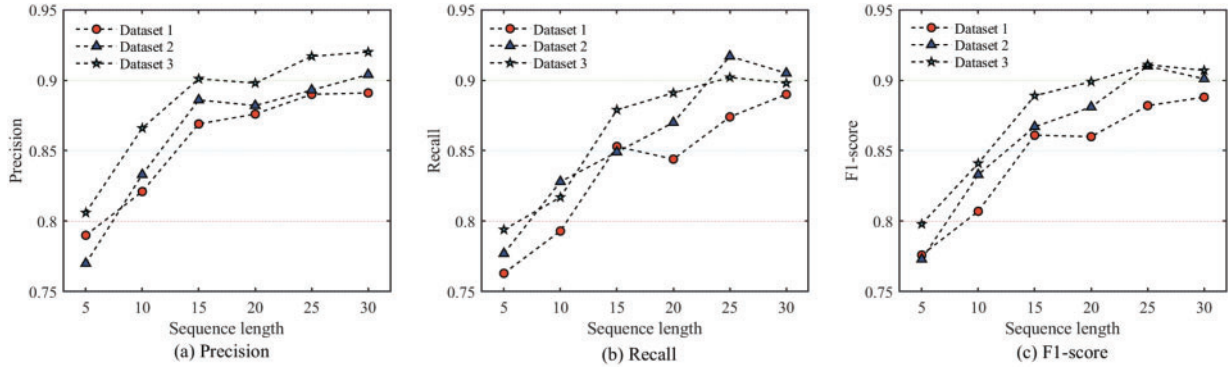

Figure 5: Sensitivity analysis of HNND with different sequence lengths

Figure 6 illustrates the model's performance across three evaluation metrics when varying the attention sizes of GAT. Key observations include:

- The HNND model exhibits minimal fluctuation in all evaluation metrics across different datasets and attention size settings. This stability is likely due to the multi-head attention mechanism of GAT, which provides sufficient temporal transaction features for nodes and allows each attention head to dynamically adjust the weight distribution of edge representations.
- Even with a smaller attention size (e.g., sizes = 2), HNND achieves relatively strong performance. This indicates that the model possesses good stability and maintains generalization capabilities on unseen data.

These findings highlight the robustness of the HNND model in learning node features, particularly regarding the dimensions of TTFs (i.e., the attention sizes of GAT).

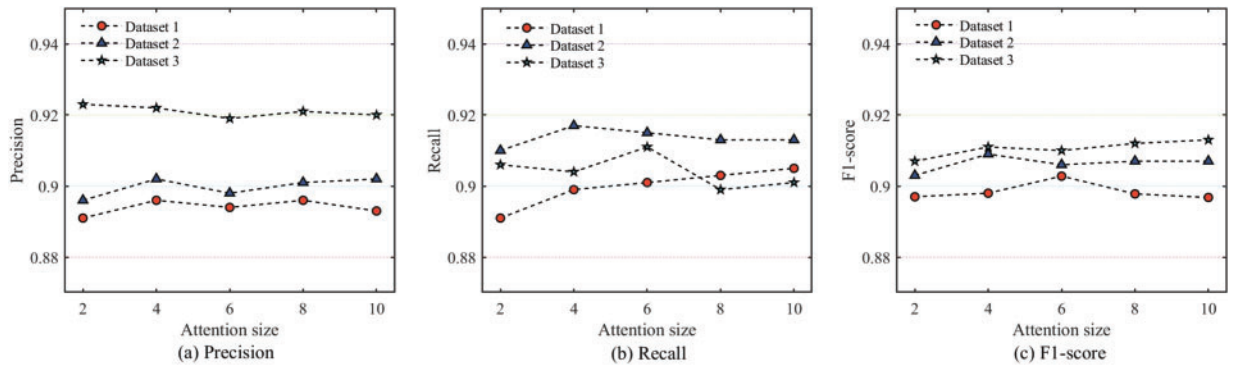

Figure 6: Sensitivity analysis of HNND with different attention sizes

5 Conclusion

To address the challenge of detecting anomalous transaction behaviors in blockchain application networks, this study introduces a hybrid neural network-based detection method, called HNND. The proposed method utilizes T2GAT and GCN to effectively capture temporal patterns (TTFs) and structural features (ASFs) from transaction graphs, respectively. Subsequently, these abstract features are adaptively fused with hand-engineering features (GSFs) using SDA, resulting in enhanced node representations that significantly improve anomaly detection performance. In addition, our experimental analysis reveals that the detection performance of HNND is sensitive to the length of transaction sequences, indicating a limitation in handling varying sequence lengths. Future research should focus on enhancing the model's robustness by improving its capability to process variable-length sequences. Potential directions for improvement include developing mechanisms for dynamically adjusting to sequence lengths or refining feature extraction techniques to better identify salient behavioral features within transaction sequences.

Acknowledgement: We would like to express our gratitude to the editors and reviewers for their detailed review and insightful advice.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Jiling Wan; data collection: Jiling Wan, Lifeng Cao, Jinlong Bai; analysis and interpretation of results: Jiling Wan, Jinlong Bai, Jinhui Li; draft manuscript preparation: Jiling Wan, Lifeng Cao, Jinlong Bai, Jinhui Li, Xuehui Du. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data used in this paper can be requested from the corresponding author upon request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Eyal I, Gencer AE, Sirer EG, Van Renesse R. Bitcoin-NG: a scalable blockchain protocol. In: Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation (NSDI); Berkeley, CA, USA: USENIX Association; 2016 Mar. p. 45–59.
2. Gilad Y, Hemo R, Micali S, Vlachos G, Zeldovich N. Algorand: scaling byzantine agreements for cryptocurrencies. In: Proceedings of the 26th Symposium on Operating Systems Principles (SOSP); 2017 Oct. p. 51–68.

3. Han Y, Li C, Li P, Wu M, Zhou D, Long F. Shrec: bandwidth-efficient transaction relay in high-throughput blockchain systems. In: Proceedings of the 11th ACM Symposium on Cloud Computing (SOCC); Virtual Event, USA: ACM; 2020 Oct. p. 238–52.
4. Lewenberg Y, Sompolinsky Y, Zohar A. Inclusive block chain protocols. In: Böhme R, Okamoto T, editors. Proceedings of the Financial Cryptography and Data Security (FC); Curacao, Berlin/Heidelberg: Springer; 2015. p. 528–47.
5. Li C, Li P, Zhou D, Yang Z, Wu M, Yang G et al. A decentralized blockchain with high throughput and fast confirmation. In: Proceedings of the USENIX Annual Technical Conference (USENIX ATC); Virtual Event: USENIX Association; 2020. p. 515–28.
6. KokorisKogias E, Jovanovic P, Gasser L, Gailly N, Syta E, Ford B. OmniLedger: a secure, scale-out, decentralized ledger via sharding. In: Proceedings of the IEEE Symposium on Security and Privacy (SP); San Francisco, CA, USA: IEEE; 2018. p. 583–98.
7. Naumenko G, Maxwell G, Wuille P, Fedorova A, Beschastnikh I. Erlay: efficient transaction relay for bitcoin. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS); London, UK: ACM; 2019 Nov. p. 817–31.
8. Yang P, Ren J. A blockchain-based data auditing scheme with key-exposure resistance for IIoT. Sci China Inf Sci. 2024 Jan;67(2):129102. doi:10.1007/s11432-023-3828-7.
9. Chen H, Wei N, Wang L, Fawzy Mohamed Mobarak W, Ali Albahar M, Shaikh ZA. The role of blockchain in finance beyond cryptocurrency: trust, data management, and automation. IEEE Access. 2024;12(37):64861–85. doi:10.1109/ACCESS.2024.3395918.
10. Krishankumar R, Dhruva S, Ravichandran KS, Kar S. Selection of a viable blockchain service provider for data management within the internet of medical things: an MCDM approach to Indian healthcare. Inf Sci. 2024 Mar;657(4):119890. doi:10.1016/j.ins.2023.119890.
11. Oksuz O. A system for storing anonymous patient healthcare data using blockchain and its applications. Comput J. 2024 Jan;67(1):18–30. doi:10.1093/comjnl/bxac155.
12. Salim MM, Yang LT, Park JH. Privacy-preserving and scalable federated blockchain scheme for Healthcare 4.0. Comput Netw. 2024 Jul;247:110472. doi:10.1016/j.comnet.2024.110472.
13. Damaševičius R, Misra S, Maskeliūnas R, Nayyar A. Convergence of blockchain and internet of things: integration, security, and use cases. Front Inform Technol Electron Eng. 2024 Oct;25(10):1295–1321. doi:10.1631/FITEE.2300215.
14. Joseph Antony A, Singh K. A blockchain-based public key infrastructure for IoT-based healthcare systems. Comput J. 2024 Apr;67(4):1531–7. doi:10.1093/comjnl/bxad079.
15. Chen H, Pendleton M, Njilla L, Xu S. A survey on ethereum systems security: vulnerabilities, attacks, and defenses. ACM Comput Surv. 2020 Jun;53(3):1–43. doi:10.1145/3391195.
16. Wu J, Liu J, Chen W, Huang H, Zheng Z, Zhang Y. Detecting mixing services via mining bitcoin transaction network with hybrid motifs. IEEE Trans Syst Man Cybern: Syst. 2022 Apr;52(4):2237–49. doi:10.1109/TSMC.2021.3049278.
17. Wang L, Xu M, Cheng H. Phishing scams detection via temporal graph attention network in ethereum. Inf Process Manag. 2023 Jul;60(4):103412.
18. Anita N, Vijayalakshmi M. Blockchain security attack: a brief survey. In: Proceedings of the 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT); Kanpur, India: IEEE; 2019 Jul. p. 1–6.
19. Wang Z, Jin H, Dai W, Choo K-KR, Zou D. Ethereum smart contract security research: survey and future research opportunities. Front Comput Sci. 2020 Oct;15(2):152802. doi:10.1007/s11704-020-9284-9.
20. Ressi D, Romanello R, Piazza C, Rossi S. AI-enhanced blockchain technology: a review of advancements and opportunities. J Netw Comput Appl. 2024;225:103858. doi:10.1016/j.jnca.2024.103858.
21. Farrugia S, Ellul J, Azzopardi G. Detection of illicit accounts over the ethereum blockchain. Expert Syst Appl. 2020 Jul;150:113318.

22. Ibrahim RF, Mohammad Elian A, Ababneh M. Illicit account detection in the ethereum blockchain using machine learning. In: Proceedings of the International Conference on Information Technology (ICIT); Amman, Jordan: IEEE; 2021 Jul. p. 488–93.
23. Poursafaei F, Hamad GB, Zilic Z. Detecting malicious ethereum entities via application of machine learning classification. In: Proceedings of the 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS); Cham: Springer; 2020 Sep. p. 120–7.
24. Kumar N, Singh A, Handa A, Shukla SK. Detecting malicious accounts on the ethereum blockchain with supervised learning. In: Dolev S, Kolesnikov V, Lodha S, Weiss G, editors. Proceedings of the Cyber Security Cryptography and Machine Learning (CSCML); Cham: Springer; 2020. p. 94–109.
25. Jourdan M, Blandin S, Wynter L, Deshpande P. Characterizing entities in the bitcoin blockchain. In: Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW); Singapore: IEEE; 2018 Nov. p. 55–62.
26. Chen W, Guo X, Chen Z, Zheng Z, Lu Y. Phishing scam detection on ethereum: towards financial security for blockchain ecosystem. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI); Yokohama, Yokohama, Japan: IJCAI Organization and AAAI Press; 2020 Jul. p. 4506–12.
27. Perozzi B, AlRfou R, Skiena S. DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14); New York, NY, USA: ACM; 2014 Aug. p. 701–10.
28. Grover A, Leskovec J. Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16); San Francisco, CA, USA: ACM; 2016 Aug. p. 855–64.
29. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. In: Proceedings of the International Conference on Learning Representations (ICLR); Vancouver, BC, Canada ; 2018 Feb.
30. Yuan Q, Huang B, Zhang J, Wu J, Zhang H, Zhang X. Detecting phishing scams on ethereum based on transaction records. In: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS); Seville, Spain: IEEE; 2020 Oct. p. 1–5.
31. Wu J, Yuan Q, Lin D, You W, Chen W, Chen C, et al. Who are the phishers? Phishing scam detection on ethereum via network embedding. IEEE Trans Syst Man Cybern: Syst. 2022 Feb;52(2):1156–66. doi:10.1109/TSMC.2020.3016821.
32. Chen L, Peng J, Liu Y, Li J, Xie F, Zheng Z. Phishing scams detection in ethereum transaction network. ACM Trans Internet Technol. 2021 Feb;21(1):1–16. doi:10.1145/3450630.
33. Ahn SJ, Kim M. Variational graph normalized autoencoders. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM); New York, NY, USA: ACM; 2021 Oct. p. 2827–31.
34. Ding K, Li J, Bhanushali R, Liu H. Deep anomaly detection on attributed networks. In: Proceedings of the SIAM International Conference on Data Mining (SDM); Philadelphia, PA, USA: SIAM; 2019 May. p. 594–602.
35. Sun X, Yang T, Hu B. LSTM-TC: bitcoin coin mixing detection method with a high recall. Appl Intell. 2022 Jan;52(1):780–93. doi:10.1007/s10489-021-02453-9.
36. Li S, Gou G, Liu C, Hou C, Li Z, Xiong G. TTAGN: temporal transaction aggregation graph network for ethereum phishing scams detection. In: Proceedings of the ACM Web Conference (WWW); New York, NY, USA: ACM; 2022 Apr. p. 661–9.
37. Chung J, Gülcehre C, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. 2014.
38. Hu H, Bai Q, Xu Y. SCSGuard: Deep scam detection for ethereum smart contracts. In: Proceedings of the IEEE Conference on Computer Communications Workshops (WKSHPS); New York, NY, USA: IEEE; 2022 May. p. 1–6.
39. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997 Nov;9(8):1735–80. doi:10.1162/neco.1997.9.8.1735.