UDC 004.89:004.75:336.71 **doi:** https://doi.org/10.20998/3083-6298.2025.03.10

Vladyslav Prosolov[1], Oleksandr Kushnerov[2], Vladyslav Sokol[3], Ruslan Trofymenko[3]

[1] Kharkiv National University of Radio Electronics, Kharkiv, Ukraine.
[2] Sumy State University, Sumy, Ukraine
[3] National Technical University "Kharkiv polytechnic institute", Kharkiv, Ukraine.

GRAPH AND TEMPORAL NEURAL MODELS FOR PROACTIVE IDENTIFICATION OF FRAUDULENT ACCOUNTS IN THE ETHEREUM BLOCKCHAIN

Abstract. Topicality. Fraudulent activities on the Ethereum blockchain pose a substantial risk to decentralized finance and require capable models not only to respond to already detected abuses but also to identify suspicious accounts proactively before losses escalate. **The subject of study** is the application of graph and temporal neural models to the task of classifying Ethereum accounts as benign or fraudulent, considering the structural relationships between addresses and the temporal dynamics of transactions. **The purpose of this** article is to develop and experimentally evaluate a neural architecture based on a multilayer perceptron as a baseline component for the subsequent integration of graph and temporal mechanisms, and to analyze its performance on the open Ethereum Fraud Detection dataset, which features a high-class imbalance. **The following results** were obtained. A baseline deep model for binary account classification was constructed using feature preprocessing, stratified data splitting, class weight balancing, L2 regularization, Dropout, and early stopping, which enabled the achievement of an ROC AUC value of approximately 0.98 under conditions of a pronounced dominance of the safe class. A detailed analysis of the confusion matrix and the precision, recall, and F1 metrics demonstrated an acceptable trade-off between reducing false positives and minimizing the proportion of missed fraudulent accounts, which is critical for real-world financial scenarios. **Conclusion.** The results indicate that a properly designed baseline neural model on tabular features can ensure high-quality proactive identification of fraudulent Ethereum accounts and serve as a starting point for further integration of graph and temporal architectures aimed at improving interpretability and robustness to the evolution of malicious behavior patterns.

Keywords: Ethereum blockchain; fraud; graph neural networks; temporal models; risk identification; machine learning; transactions.

Introduction

Problem relevance. Fraud in blockchain has long ceased to be an isolated incident at the periphery of the market. It has evolved into a systemic threat to centralized exchanges, DeFi platforms, and mass retail investors. Aggregate losses are consistently measured in tens of billions of dollars, forming a distinct class of financial threats [1; 12]. As the capitalization of cryptoassets grows, not only the scale but also the organization of schemes changes: they combine technical exploits of smart contracts, phishing campaigns, long-prepared rug pull projects, and the use of gaps in regulatory frameworks across jurisdictions [1; 11; 12].

In this context, the blockchain is increasingly viewed as a critical component of financial infrastructure. Its reliability depends not only on the correct implementation of cryptographic mechanisms but also on the ability to detect suspicious transactions promptly, with minimal delays, approaching real-time [1; 9; 12]. This issue is particularly significant for the Ethereum ecosystem, where the base protocol guarantees the authenticity of signatures and the consistency of state but does not address the legitimacy of the motives behind monetary flows [5; 12]. The detection of toxic addresses, the construction of wallet risk profiles, and the identification of fraudulent chains are delegated to external analytics services and exchange solutions, which in practice often react only after funds have passed through cross-chain bridges, mixing services, and cascades of centralized platforms [5; 9; 12].

The combination of address pseudonymity and transaction transparency creates a complex situation. Formally, industrial-scale transaction graphs are accessible to everyone, but this does not guarantee the detection of complex fraudulent patterns [1; 12]. The public ledger stimulates the development of graph and temporal analysis methods, yet adversaries use the same data. Adaptation of schemes to common heuristics takes the form of splitting flows into small amounts, multi-step address carousels, and multi-stage cross-chain paths that blur topological and temporal signals [5; 12]. This gives rise to a continuous adaptation process between model developers and actors seeking ways to bypass heuristics. The most successful models maintain an advantage for a limited period, after which fraudsters change their evasion logic to minimize visible traces. [2; 6; 7; 12].

Classical anomaly detection methods based on threshold rules or simple statistical filters were designed for scenarios with centralized identification and stable behavioral profiles. In blockchain, analysts operate only with addresses, volumes, and timestamps, and are forced to compensate for the lack of context through aggressive feature engineering [1; 8; 12]. Performance improves only up to a point: multilayered fraud strategies adjust local behaviors to appear normal, and anomalies manifest only at the global level of the money flow pattern, which naturally leads to the formalization of graph and temporal models [2; 3; 7; 12].

A separate challenge is the extreme class imbalance typical of financial monitoring on open transaction graphs. The share of confirmed malicious nodes represents a small fraction of the dataset, and the simple proportion of correctly classified addresses is not an informative metric. A model that always optimistically classifies addresses as legitimate achieves more than 99

percent formally correct decisions but fails to detect rare events that are critical for security. Therefore, the metrics must be sensitive to the minority class: recall, F1, PR AUC, and the share of incidents at a constrained FP level [1; 2; 3; 8]. As a result, the task shifts from a binary decision to allocating analytical resources to the most suspicious segments of the graph, which aligns with the requirements of modern financial monitoring [1; 9].

At the same time, there is a growing demand for models that jointly capture graph topology and temporal dynamics while respecting constraints on computational resources and latency that are acceptable for practical deployment in the Ethereum network [2; 3; 7; 10]. Graph neural networks with temporal attention mechanisms, temporal GNN architectures, and attention-based models, such as TempoKGAT, demonstrate the ability to detect complex spatiotemporal patterns of fund flow more accurately under severe class imbalance [3; 7; 10]. At the same time, their practical value for Ethereum still requires further validation on realistic data streams, considering memory constraints, latency, and integration with exchange and regulatory services, which makes this research direction one of the key priorities [1; 2; 5; 9; 12].

Literature review. In the contemporary literature on blockchain ecosystem security, a persistent gap is emphasized between formally correct cryptographic implementation and actual protection against fraud. For Ethereum, this issue is particularly pronounced: core mechanisms guarantee the authenticity of transaction signatures but do not address the motives or admissibility of an operation. In the absence of a strong regulatory framework, a significant share of protective functions shifts to external analytics tools and informal practices [1; 7; 12]. Additionally, smart contracts, which have increased the platform’s attractiveness for developers, have simultaneously opened new avenues for exploits, logic errors, and opaque financial flows [1; 7; 12]. Classical approaches to fraud detection have historically focused on features tightly linked to an identifiable user and have relied on models such as logistic regression, decision trees, or ensembles of these models. In blockchain systems, such signals are either missing, fragmented, or weakly trusted. The lack of centralized identification limits analytics to abstract addresses and transactions, where auxiliary features are derived from the graph's structure or its history of cash flows [1; 12].

A particular focus is placed on studies that reconstruct latent address behavior using only public blockchain data through feature engineering [5; 11; 12]. Such methods include aggregates like mean or median balance, transaction counts, volume distributions, interaction frequencies with counterparties, and centrality metrics in the graph. They yield a substantial performance gain over naive models but quickly approach a ceiling in complex schemes where anomalies are visible only through the coordination of multiple addresses [1; 5; 11; 12].

Research in detection is increasingly oriented toward exploiting the structure of relationships among addresses. Graph neural networks alter the modeling perspective: a node is no longer treated as an isolated point, but as part of a complex network where information circulates among neighbors [2; 7; 6]. Convolution-based architectures propagate information through several topological layers, and neighbor sampling enables scaling to large graphs where complete aggregation is infeasible [2; 7]. Graph attention mechanisms provide additional flexibility, allowing the model to assign distinct weights to specific types of connections, which is crucial in graphs with numerous low-informative edges [2; 6; 10; 12].

Most current graph implementations rely on the assumption of structural stationarity, where the graph is fixed at a given moment, and internal connections are treated uniformly, regardless of transaction time. For Ethereum networks, such a simplified view risks distorting the genuine dynamics of cryptoassets. Moreover, many models assume that the graph or its substantial part fits into main memory, which is problematic for systems with millions of addresses and edges [2; 7; 9]. In parallel, temporal models are emerging that treat sequences of events as the primary object [3; 7; 8]. Recurrent architectures and their variants have long been the standard for time series tasks, allowing the accumulation of information about past states and their use when interpreting new signals [3; 8]. In blockchain analytics, this enables the observation of the evolution of transaction rates, volumes, or counterparties, while attention mechanisms allow the model to focus on critical episodes rather than the averaged profile [1; 7; 8].

Temporal approaches, however, poorly capture the position of an address within the broader network configuration; the operation sequence of an isolated node, even with heavy activity, may mask its role in a complex network attack [3; 2; 7]. This creates a methodological gap: graph models provide a better description of topology but oversimplify the temporal dimension, whereas sequential models focus on behavioral dynamics while largely ignoring structure [1; 3; 7]. In response to these limitations, a line of hybrid architectures is emerging that combines structural and temporal components [2; 3; 7; 10]. One strategy is to analyze graph snapshots at different time points sequentially, process them using graph networks, and feed the resulting representations into a recurrent or attention-based block to model dynamics [3; 7; 10]. Another approach integrates temporal information directly into the graph structure by annotating nodes and edges with timestamps and adapting state update rules to the temporal nature of the data [7; 10]. In parallel, work is underway on distributed training and graph temporal methods that distribute computation across nodes to improve scalability under challenging conditions [2; 6; 8; 9].

An important practical issue of hybrid solutions is the large number of parameters and resource demands, complex hyperparameter tuning, and an elevated risk of overfitting on small or extremely imbalanced datasets [1; 2; 3; 5]. Experience from real systems shows that ensembles of simpler models, where each component detects a specific aspect (static features, local connectivity, or temporal dynamics), often provide more

controllable and stable behavior than highly complex universal architectures [1; 3; 5; 8].

The overall review suggests that, despite the breadth of modern tools, a notable shortage of studies remains that explicitly focus on balancing model complexity with stable performance in production systems [1; 2; 5; 9]. Issues of scalability, resource consumption, interpretability of results, and decision latency are often secondary to improving offline metrics on test sets, which opens a research space for building hybrid architectures with a transparent and manageable structure that combines the relevant properties of graph and temporal models while controlling the number of parameters [1; 2; 7; 6; 10].

In this context, **the purpose of the research** is to propose and experimentally validate a hybrid neural architecture that combines expressive graph-based aggregation with a careful treatment of the temporal structure of transactions in the Ethereum network, while maintaining a manageable number of parameters and ensuring suitability for deployment under real-world blockchain workloads.

1. Methodology

The study is based on the Kaggle Ethereum Fraud Detection dataset, which comprises 9,842 Ethereum addresses labeled as fraudulent [4]. Each record represents an address as a vector of 45 features, including transaction volumes, the number of interactions with unique counterparties, temporal statistics, and derived structural network characteristics, which is consistent with contemporary approaches to anomaly detection in blockchain graphs [5; 11; 12]. The feature matrix has the form $X \in \mathbb{R}^{N\times d}$, where $N = 9842$ is the number of addresses and $d = 45$ is the feature space dimension – the label vector $y \in \{0,1\}^N$, where $y_i = 1$ denotes a fraudulent address. The dataset exhibits a strong class imbalance, with approximately 82 percent of records being legitimate and 18 percent fraudulent, which aligns with trends in financial fraud detection systems, where genuine events dominate [1; 2; 3]. Denoting the number of legitimate examples by N_0, and the number of fraudulent examples by N_1, one obtains

$$\frac{N_1}{N_0 + N_1} \approx 0.18 \quad (1)$$

$$\frac{N_0}{N_0 + N_1} \approx 0.82 \quad (2)$$

which implies that most addresses do not exhibit suspicious activity, while significant financial losses are generated by a small subset of fraudsters [1; 2; 12].

Preprocessing comprised several stages. First, all features were standardized using the StandardScaler. For each feature j, the mean μ_j and standard deviation $\sigma_j$were computed, and the transformation

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \quad (3)$$

was applied to achieve zero mean and unit variance along each coordinate, as recommended for deep models in financial data applications [1; 2; 3]. Such normalization is critical for deep networks when individual features have different scales, since this complicates optimizer convergence [1; 2]. The dataset was then split into three subsets: 70% for training, 15% for validation, and 15% for testing. This is formalized as three disjoint index sets $I_{train}, I_{val}, I_{test}$, with stratification, so that the proportion of the positive class in each subset approximately matches the global ratio N_1/N [1; 2; 3; 8]. The next step was class balancing through weights. For class frequencies $f_0 = N_0/N$і $f_1 = N_1/N$ в the weights were computed as

$$w_0 = \frac{1}{f_0} \quad (4)$$

$$w_1 = \frac{1}{f_1} \quad (5)$$

and, after normalization, for this sample $w_0 \approx 1.0$, $w_1 \approx 5.5$. The importance of weighting the rare class has been confirmed in studies on imbalanced data problems [2; 6; 9; 11].

The model architecture was chosen as a dense neural network (Dense NN). Unlike LSTM or GRU architectures, which require complete sequences, the data here are represented as aggregated features constructed from the interaction graph [1; 2; 5; 12]. This makes it possible to map the input address vector in the feature space via

$$f_\theta: \mathbb{R}^d \to ,[0][1] \quad (6)$$

where $f_\theta(x_i)$ is the estimated probability that an address belongs to the fraudulent class. An explicit graph model would require an adjacency matrix $A \in \{0,1\}^{N\times N}$, which for large N mposes memory and parameter count constraints and leads to typical overfitting risks [2; 3; 7]. Structural information is already incorporated through features such as the number of neighbors, counterparty types, balances, and centrality measures, which $\vec{x}_i$ is the result of a preliminary mapping of the transaction graph into the feature space [2; 5; 12]. Regularization in the model is based on a combination of L2 penalty $\lambda \parallel \theta \parallel_2^2$ and dropout for hidden layers, which significantly constrains parameter growth and minimizes the risk of overfitting, especially for small, imbalanced samples [1; 2; 3]. Dropout enables the network to distribute information and temporarily suppress weight coadaptation, while the L2 penalty maintains a flat weight distribution, a standard practice in financial fraud and cybersecurity tasks [1; 2; 3].

Training is performed using Adam, an optimizer that adapts the step size for each parameter based on the first and second moments of the gradient. An initial learning rate of $\eta_0 = 1 \times 10^{-4}$ provides stable convergence for most standard architectures [1; 2]. To solve the task of classifying fraudulent addresses, the loss function is set to binary cross-entropy, which is the classical choice for

two-class problems with significant imbalance [1; 2; 3]. The loss for a single example is given by

$$\ell_i = -(y_i \log\, p_i + (1 - y_i)\log\,(1 - p_i)) \quad (7)$$

and the total loss with class weights is

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} w_{y_i}\,\ell_i. \quad (8)$$

Confident misclassifications receive higher penalties, which is crucial in settings where missed fraudulent addresses carry an exceptionally high cost [1; 2; 3; 8].

The mini-batch size varied from 128 to 1024. Initially, smaller batches introduced noise and helped escape local minima, while increasing the batch size in later epochs improved training efficiency [1; 2; 3]. An early stopping mechanism was used, terminating training when the validation loss failed to improve for more than 15 epochs, combined with a learning rate reduction when the loss plateaued [1; 2, 3]. Accuracy is not very informative for such a dataset,

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (9)$$

because it does not account for missed fraudulent cases, the evaluation focuses on precision, recall, F1 score, area under the ROC curve, and the behavior along the precision-recall curve, following recommendations for fraud detection models [1; 2; 3; 8]. These metrics are defined as

$$\text{Precision} = \frac{TP}{TP + FP} \quad (10)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (11)$$

$$F1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (12)$$

Here, the ROC AUC reflects ranking quality across all thresholds, while the precision-recall curve is more indicative for class-imbalanced datasets and serves as the basis for selecting the operating threshold [1; 2; 3].

2. Model Architecture

In financial fraud detection tasks, there is a natural temptation to employ highly complex models, such as GRU stacks with attention mechanisms, graph convolutional layers, and residual connections, primarily since recent work on financial fraud and blockchain anomalies reports accuracy values close to 99 percent on massive datasets [1; 2; 6]. On small datasets, the situation changes, because the same architectures, without strong regularization, tend to memorize training examples and implement mappings $f(x_i) \approx y_i$ essentially only for elements of the training set, with minimal generalization to new addresses, which is a typical issue for deep models in financial fraud problems [1; 2; 3].

In the present setting, the dataset size is limited, so the architectural complexity is tightly controlled. If the number of model parameters is denoted by $P(\theta)$, тhen each additional dense layer with n_{in} inputs and n_{out} outputs contributes

$$P_{dense} = n_{in} \cdot n_{out} + n_{out} \quad (13)$$

that is, new degrees of freedom, and when $P(\theta) \gg N$ the model is almost inevitably driven toward overfitting, as documented in deep learning surveys for fraud detection [1; 2]. Considering this, a sequential fully connected architecture with two hidden layers is deliberately chosen instead of convolutional or recurrent structures, which are more justified for strictly sequential or explicitly graph-based input representations [1; 2; 3; 7]. Model interpretability is just as important as numerical metrics. When the model flags a particular address as fraudulent, practical deployment requires at least an approximate answer to the question why this address was chosen, which aligns with explainability requirements in financial systems and regulatory approaches to fraud [1; 2; 6]. Deep, multi-branch models with graph layers and complex attention mechanisms, such as SEFraud or ASA GNN, are significantly harder to explain to stakeholders and regulators. In contrast, a two-layer dense network remains sufficiently transparent to support post-hoc interpretation methods, such as SHAP or gradient-based attribution, which have already been described in the context of financial fraud analysis [1; 6].

The architecture graph is linear. The input layer receives 45 normalized features, formally a vector $x \in \mathbb{R}^{45}$, which corresponds to aggregated transactional, network, and temporal characteristics, as in typical datasets for fraud and blockchain anomalies [4; 5; 12]. The first hidden layer is dense with 128 neurons, and its number of parameters is

$$P_1 = (45 \cdot 128) + 128 = 5888,$$

while the ReLU activation $ReLU(z) = \max(0, z)$ allows established practice in tabular and graph-oriented financial fraud tasks [1; 2; 3]. Batch normalization is then applied, which for an activation vector $h \in \mathbb{R}^{128}$ transforms each component as

$$\hat{h}_j = \frac{h_j - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}, y_j = \gamma_j \hat{h}_j + \beta_j \quad (14)$$

where μ_j and σ_j^2 are mini-batch estimates of mean and variance, and γ_j, β_j act as learnable scale and shift parameters, stabilizing the activation distribution and permitting more aggressive learning rate settings, as recommended in deep learning studies on fraud detection [2; 3]. Dropout with probability $p = 0.5$ is applied after normalization. For each component y_j, a random mask $m_j \sim Bernoulli(1 - p)$ is sampled, and during training

$$\tilde{y}_j = \frac{m_j}{1 - p} y_j \quad (15)$$

is used, meaning that roughly half of the neurons are randomly zeroed, while the remaining ones are rescaled to preserve the expected activation level. This approach distributes information across many neurons and reduces the risk of overfitting, as advised for small and imbalanced financial datasets [1; 2; 3].

The second hidden layer is also dense, but with 64 neurons, thus reducing the representation dimension; its parameter count is

$$P_2 = (128 \cdot 64) + 64 = 8256,$$

and ReLU is again used as the activation function. This is followed by another batch normalization layer and Dropout with the same probability $p = 0.5$, forming a regularization symmetric block that matches conservative training practices in fraud detection [1; 2; 3]. The output layer contains a single neuron with sigmoid activation

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}} \quad (16)$$

where $z = w^{\top}h + b$, a $h \in \mathbb{R}^{64}$ is the output of the second hidden layer, and the number of parameters in the output layer is

$$P_3 = 64 \cdot 1 + 1 = 65.$$

The total number of model parameters is computed as

$$P_{total} = P_1 + P_2 + P_3 + P_{BN} \quad (17)$$

where P_{BN} accounts for the parameters of both batch normalization layers (scales and shifts). In this configuration, the total number of parameters is 14977, which can be regarded as a manageable level for a training set of one to two thousand examples, typical of public fraud and blockchain anomaly datasets [1; 2; 4].

The choice of a relatively aggressive Dropout value $p = 0.5$ requires additional justification. In image or text tasks, typical values range from 0.2 to 0.3. However, after the 70/15/15 split, only about 1,500 training examples remain, with roughly 270 belonging to the positive class, creating an extreme imbalance [1; 2; 3]. Under these conditions, the ratio $\frac{P_{total}}{N_{train}}$ is fairly large, and the model tends to memorize primarily positive examples; Dropout $p = 0.5$ combined with an L2 penalty with a coefficient $\lambda = 0.01$ adds the term

$$\mathcal{L}_{reg}(\theta) = \lambda \sum_k \theta_k^2 \quad (18)$$

to the loss function, constraining weight growth and steering the model toward solutions with smaller norms $\| \theta \|_2$, in line with recommended anti-overfitting strategies in fraud detection [1; 2; 3]. Batch normalization additionally serves a regularizing role, since μ_j and σ_j^2 are estimated at the mini-batch level, and each training step injects slight noise into the activation distribution. Combined with early stopping, which halts training when $\mathcal{L}_{val}$ it does not improve for 15 epochs, and with learning rate reduction on validation loss plateaus, this yields a conservative training regime [1; 2; 3]. The combination of Dropout, L2 regularization, batch normalization, early stopping, and adaptive learning rate reduction represents a deliberate compromise in which the model does not reach maximal training accuracy. Instead, it achieves more stable and higher F1, ROC AUC, and PR AUC values on validation and test sets, as recommended in surveys on fraud and blockchain anomaly detection [1; 3; 2; 8].

The initial 45 features can be naturally divided into three groups that capture different aspects of address behavior. Transactional features describe the number of incoming and outgoing operations and the average and extreme transfer amounts; network features encode the structure of the local neighborhood in the interaction graph, including the number of unique counterparties, aggregated partner balances, and simple centrality indicators; temporal features capture the distribution of transactions over time and the irregularity of activity, consistent with feature construction practices for Ethereum and Bitcoin fraud [4; 5; 12]. Formally, all of them are represented by a vector $x \in \mathbb{R}^{45}$, but each subgroup lies in its own subspace $\mathbb{R}^{d_{tx}} \oplus \mathbb{R}^{d_{net}} \oplus \mathbb{R}^{d_{time}}$, reflecting different information types and allowing the model to indirectly account for local topology and temporal patterns encoded in the feature space [4; 6; 7; 12].

3. Results Of Training and Analysis

The training metrics of the model exhibit behavior typical of deep learning algorithms applied to financial transaction datasets with a strong class imbalance. The loss and accuracy curves follow a characteristic pattern for tasks in which most examples belong to the safe class, while fraudulent examples may have structurally distinctive features. The rapid improvement phase is short, but the effect of regularization is pronounced. Aggressive Dropout and L2 settings restrain overfitting and preserve stability even on a limited sample, although this does not guarantee the same level of data realism in a production environment.

The dynamics of learning metrics are shown in Fig. 1. The three plots reveal a consistent trend. The loss decreases to operating values around 0.35, validation accuracy stabilizes at approximately 0.95, and the AUC increases monotonically to 0.9788, which falls within the range interpreted as excellent discriminative performance. In practice, such behavior is typical for configurations where the number of parameters exceeds the data volume and regularization methods (such as Dropout, L2, and batch normalization) are applied stringently. The validation loss behavior after the 60th epoch, along with occasional fluctuations in accuracy, does not indicate critical overfitting; however, early stopping could help save computational resources.

These metrics are not universal and always depend on the quality of the dataset. In this case, the dataset is already cleaned and prefiltered, which compensates for some limitations of the simpler architecture. The area

under the ROC curve (AUC) is more critical here than raw accuracy, because the final decision is made with a variable classification threshold, and system behavior is adjusted to the user's specific risk tolerance.

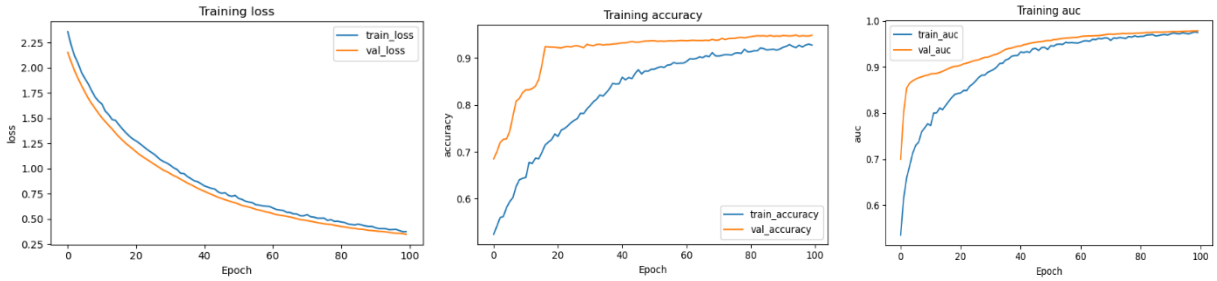

Fig. 1. Dynamics of model training performance metrics

The confusion matrix reveals a clear structure of classification outcomes on the test set, presented in Fig. 2. The model correctly classified 1507 legitimate addresses as safe (TN) and 361 as fraudulent (TP), while 26 legitimate addresses were flagged as suspicious (FP) and 75 fraudulent addresses were missed (FN). False positive errors maintain a more acceptable risk profile; their cost is limited to temporary user inconvenience and an additional marker for KYC or manual checks, and many industrial fraud prevention systems treat a low single-digit FP rate as operationally acceptable. False negatives, by contrast, pose systemic risk because they correspond to direct financial losses or undetected attacks. Out of 436 fraudulent addresses, 75 remained undetected, resulting in a false negative rate of approximately 17.2 percent. This level is high for a standalone system but acceptable as a first layer in a multistage control cascade where external heuristics and regulatory instruments provide subsequent blocking.

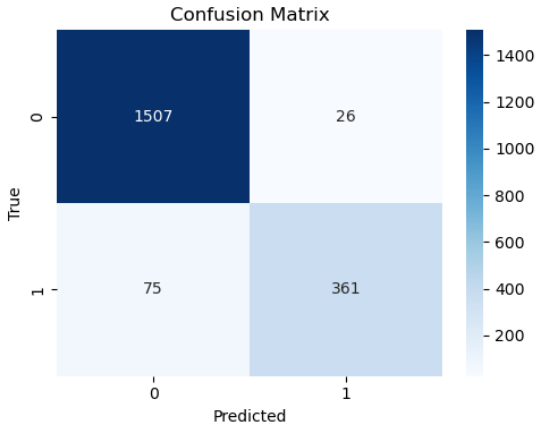

Fig. 2. Confusion matrix of classification outcomes

Precision Recall curve, located in the upper part of the value range, confirms model robustness under threshold variation; for recall above 0.8, precision remains higher than 0.9, which allows adjustment of the operating threshold without substantial quality degradation and keeps FP within bounds that are acceptable for primary screening tasks.

Qualitative assessment of the model is expressed through ROC and Precision Recall curves, shown in Fig. 3. The ROC curve preserves an almost ideal steepness in the region of low false positive rates, maintaining the actual positive rate above 0.90 at FPR below 0.03. At the same time, an AUC of 0.9788 indicates high-quality risk ranking and nearly optimal class separation under this metric.

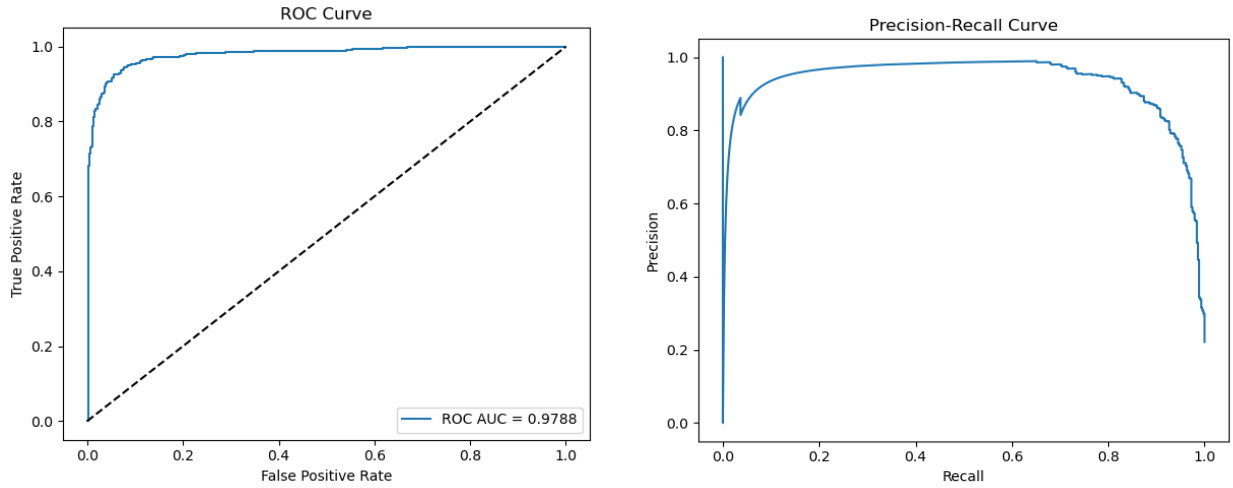

Fig. 3. Curves for evaluating classification quality

The detailed classification report highlights nuances in the region of class imbalance. For legitimate addresses, precision, recall, and F1 remain consistently high, whereas for fraudulent addresses, recall is lower, at 0.828, which is a direct consequence of the architectural compromise. Aggressive regularization with a Dropout

Rate of 0.5 effectively constrains overfitting but prevents achieving the optimal recall for the minority class. Reducing Dropout to 0.3 increases training accuracy, yet validation AUC decreases, and the number of FNs grows, once again confirming the lack of parameter portability between test and production samples. The model operates as part of a cascade and can guarantee practical robustness against fraud only when combined with additional heuristics and external monitoring.

Discussion of results

The obtained AUC value of 0.9788 on a sample of approximately 2000 Ethereum addresses indicates a high discriminative capacity of the dense neural model under consideration; however, it more accurately delineates an upper bound for this specific task and dataset rather than establishing a universal deployment standard. For Kaggle like financial datasets with carefully selected and aggregated features, it confirms the model's ability to exploit patterns already embedded in the feature space. However, the question of detecting novel, weakly labeled anomalies remains open. Outside a controlled environment, where fraud structures evolve and are not promptly reflected in the training data, such an architecture may lose part of its effectiveness precisely because of the limitations of the underlying feature space. In studies employing full graph neural networks that operate with explicit adjacency matrices, multi-layer message passing, and contrastive learning, AUC scores in the 0.92–0.95 range are typical on unfiltered on-chain data. These values are formally lower than those obtained in this work, but they are achieved under stronger noise, fragmented labeling, and less aggressive preprocessing. Therefore, a direct numerical comparison is not appropriate without a unified feature engineering scheme. Furthermore, graph-based models differ substantially in computational complexity. While the dense model considered here, with roughly 15000 parameters, has linear complexity $O(N)$ and processes examples independently, typical GCN or GAT variants scale as $O(N + E)$ or worse, depending on the edge density, and require recursive neighbor queries and subgraph storage in memory. For systems operating in near real time, where each millisecond of latency is critical, this creates a considerable barrier to deployment.

For a small, heavily preprocessed sample, the experiments demonstrate that a compact, dense model can be as effective as more complex architectures, provided that structural and temporal information have already been distilled into informative, aggregated features. However, these observations cannot be directly applied to real-world on-chain streams in production environments, where millions of addresses, dynamic graph structures, new node types, and evolving interaction patterns are present. Under such conditions, the advantage of graph-based approaches, which can model deep topology and inter-address dependencies, may become more pronounced.

The specificity of the obtained results lies not only in the AUC level but also in the nearly symmetric precision for both classes, 0.95 for legitimate and 0.93 for fraudulent addresses, at a class ratio of about 3.5 to 1. This case is representative of problems dominated by everyday examples, where the precision of the minority class typically degrades if the imbalance is not considered or the class weights are poorly chosen. In the presented configuration, optimal tuning of class weights and regularizers partially compensates for the small sample size, maintaining sensitivity to fraudulent patterns without sharply increasing the number of false positives. The confusion matrix confirms that the model retains an acceptable FP profile for primary screening while keeping the proportion of missed fraudulent addresses at a level compatible with a cascaded defense scenario in which subsequent analysis layers further narrow the risk perimeter.

An indirect graph representation, that is, a feature set derived from the transaction graph without an explicit adjacency matrix, enables a robust solution for a small dataset without losing the basic transactional context. Structural characteristics, such as neighbor counts, counterparty profiles, aggregated balances, and simple centrality measures, together with temporal statistics, including activity distributions over time and burst patterns, transform the original graph into a tabular space in which the proposed neural architecture operates effectively. From a resource perspective, this is often a more balanced approach than forcing GNN or sequence models on a few thousand addresses or short transaction chains, especially when memory budgets and allowable latency are constrained. Another advantage is transparency: analyzing input feature weights and model behavior at the level of feature groups simplifies interpretability for stakeholders and regulators compared with profound message passing networks.

The temporal dimension considerably limits the use of AUC as a universality indicator. A model trained on historical blockchain activity snapshots inevitably encounters distribution drift when processing new transaction streams. This concerns both the emergence of new fraud patterns and the gradual change in the behavior of legitimate users and infrastructure nodes. In financial anti-fraud systems, an AUC drop of 8–10 percentage points over several months, without retraining, is already considered a warning signal, regardless of the initial performance level. From this perspective, the current dense network configuration serves as an optimal first-level filter for a relatively narrow time window and a specific subgraph; however, it cannot be considered stable in the absence of update mechanisms.

The simplicity of the architecture and modest memory requirements, in contrast to the need to keep large graphs in RAM for GNNs, create favorable conditions for a strategy of regular retraining on fresh data. Periodic fine-tuning on updated samples allows the model to adapt to distribution drift without drastically increasing structural complexity or inference costs. At the same time, such a tabular architecture cannot serve as a standalone solution for systemic defense against evolving attacks. A rational strategy is to employ it as one layer in a multi-level system where heuristic rules, signatures of known schemes, aggregated graph features, and online adaptation mechanisms adjust weights on

current data and sustain the overall robustness of the system.

Conclusions

The regularized dense neural model with carefully balanced classes demonstrates performance comparable to more complex architectures for the task of detecting fraudulent Ethereum addresses, provided that rich, semi-manually aggregated features are available. The high AUC, together with balanced precision and recall for both classes, indicates that the model can capture complex, nonlinear combinations of transactional, network, and temporal characteristics without additional graph-based or sequence blocks. The strength of the approach lies not only in the numerical metrics but also in the absence of clear signs of overfitting on a small and imbalanced dataset. A flexible class weighting scheme, combined with Dropout and L2 regularization, provides a productive balance between model complexity and generalization ability, reducing the risk of over-sensitivity to individual addresses or local structures.

At the same time, the main limitations of the tabular architecture are linked to insufficient exploitation of the explicit transaction graph topology and detailed temporal dynamics. These aspects are crucial for identifying multihop laundering schemes, cascading attacks, composite cross-chain paths, and scenarios where anomalies manifest only at the level of global fund flow patterns. Moving to a hybrid model that augments aggregated features with compact graph embeddings and sequential or contextual vectors from Transformer architectures can shift the balance between complexity and quality toward better capture of spatio-temporal dependencies. This approach enables the combination of the interpretability and resource efficiency of the tabular model with the expressiveness of graph and temporal mechanisms.

Further development of the solution is best aligned with adaptive training regimes. Introducing federated or at least regular online fine-tuning would transform the static dense model into a component that gradually adapts to new fraud patterns, distribution drift, and protocol changes. In this configuration, the tabular architecture serves as a lightweight and interpretable first-level filter. At the same time, more complex graphs and temporal blocks handle a subset of high-risk addresses, providing deeper contextual analysis.

The current results should be viewed as a reference for model behavior on a particular historical snapshot and Ethereum subgraph rather than as a universal guarantee of future performance. High AUC values and balanced precision indicate that the model is ready for integration into multi-level analytics for a blockchain service, but this is feasible only under continuous monitoring of distribution drift, periodic validation on updated samples, and scheduled parameter updates. In practice, robustness and security are ensured not by a single classifier with record AUC, but by a flexible ecosystem of filters in which each component, from simple heuristics and signatures to graph and temporal models, can both individually and jointly respond to environmental changes and evolving risk patterns. Within such an ecosystem, the dense neural model considered here serves as a complexity-controlled and interpretable base module on top of which hybrid architectures for proactive identification of fraudulent accounts in the Ethereum blockchain can be constructed.

References

1. Chen, Y., Zhao, C., Xu, Y., Nie, C., & Zhang, Y. (2025), “Year-over-year developments in financial fraud detection via deep learning: A systematic literature review”, arXiv https://doi.org/10.48550/arXiv.2502.00201
2. Cheng, D., Zou, Y., Xiang, S. & et al. (2025), “Graph neural networks for financial fraud detection: a review. *Front. Comput. Sci.* 19, 199609, https://doi.org/10.48550/arXiv.2411.05815
3. Cheng, Y., Guo, J., Long, S., Wu, Y., Sun, M., & Zhang, R. (2024, August), “Advanced financial fraud detection using GNN-CL model” *International Conference on Computers, Information Processing and Advanced Education (CIPAE)* (pp. 453-460), https://doi.org/10.48550/arXiv.2407.06529
4. Ethereum Fraud Detection Dataset. Kaggle: Your Machine Learning and Data Science Community, https://www.kaggle.com/datasets/vagifa/ethereum-frauddetection-dataset.
5. Gu, Z, Dib, O. (2025), “Enhancing fraud detection in the Ethereum blockchain using ensemble learning”, *PeerJ Computer Science,* https://doi.org/10.7717/peerj-cs.2716
6. Li, K., Yang, T., Zhou, M., Meng, J., Wang, S., Wu, Y., ... & Tong, Y. (2024, August), “Sefraud: Graph-based self-explainable fraud detection via interpretative mask learning”, *30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 5329-5338). https://doi.org/10.48550/arXiv.2406.11389
7. Longa, A., Lachi, V., Santin, G., Bianchini, M., Lepri, B., Lio, P., ... & Passerini, A. (2023), “Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities”, arXiv preprint arXiv:2302.01018. https://doi.org/10.48550/arXiv.2302.01018.
8. Mukhamadiyev A, Nazarov F, Yarmatov S, & Cho J. (2025), “Intelligent Algorithms for the Detection of Suspicious Transactions in Payment Data Management Systems Based on LSTM Neural Networks”, *Sensors,* vol. 25(21), pp. 66-83, https://doi.org/10.3390/s25216683.
9. Real-Time Financial Fraud Detection Using Adaptive Graph Neural Networks and Federated Learning. (2025), *International Journal of Management and Data Analytics (IJMADA),* vol. 5(1), pp. 98-110. https://doi.org/10.5281/zenodo.15107110.
10. Sasal, L., Busby, D., & Hadid, A. (2024, December), “Tempokgat: A novel graph attention network approach for temporal graph analysis” *International Conference on Neural Information Processing* (pp. 212-226). Singapore: Springer Nature Singapore, https://doi.org/10.48550/arXiv.2408.16391
11. Shaizat, M., & Mussiraliyeva, S. (2025), “Enhanced identification of illicit bitcoin transactions through genetic algorithm-based feature selection”, *Eastern-European Journal of Enterprise Technologies,* 4(9 (136), pp. 34-42, https://doi.org/10.15587/1729-4061.2025.335630.
12. Shevchuk R, Martsenyuk V, Adamyk B, Benson V, & Melnyk A. (2025), “Anomaly Detection in Blockchain: A Systematic

Review of Trends", *Challenges, and Future Directions. Applied Sciences,* vol. 15(15), p. 83-90, https://doi.org/10.3390/app15158330

Received (Надійшла) 27.10.2025
Accepted for publication (Прийнята до друку) 11.11.2025

ВІДОМОСТІ ПРО АВТОРІВ/ ABOUT THE AUTHORS

Просолов Владислав Валерійович – аспірант, асистент кафедри безпеки інформаційних технологій, Харківський національний університет радіоелектроніки, Харків, Україна;
Vladyslav Prosolov – PhD student, assistant at the Department of Information Technology Security, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine;
e-mail: vladyslav.prosolov@nure.ua; ORCID Author ID: https://orcid.org/0009-0002-1276-4828.

Кушнерьов Олександр Сергійович – доктор філософії, старший викладач кафедри економічної кібернетики, Сумський державний університет, Суми, Україна;
Oleksandr Kushnerov – PhD, Senior Lecturer of the Economic Cybernetics Department, Sumy State University, Sumy, Ukraine;
e-mail: o.kushnerov@biem.sumdu.edu.ua; ORCID Author ID: https://orcid.org/0000-0001-8253-5698;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=58318301800.

Сокол Владислав Євгенович – кандидат технічних наук, докторант кафедри кібербезпеки Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Vladyslav Sokol – PhD, doctoral student of Cyber Security Department National Technical University "Kharkiv polytechnic institute", Kharkiv, Ukraine;
e-mail: Vladyslav.sokol@gmail.com; ORCID Author ID: https://orcid.org/0009-0009-9446-2049;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=56290096700.

Трофименко Руслан Валентинович – аспірант кафедри кібербезпеки Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Ruslan Trofymenko – PhD student of Cyber Security Department National Technical University "Kharkiv polytechnic institute", Kharkiv, Ukraine;
e-mail: Ruslan.Trofymenko@cs.khpi.edu.ua; ORCID Author ID: https://orcid.org/0009-0001-3114-2269.

ГРАФОВІ ТА ЧАСОВІ НЕЙРОННІ МОДЕЛІ ДЛЯ ПРОАКТИВНОЇ ІДЕНТИФІКАЦІЇ ШАХРАЙСЬКИХ ОБЛІКОВИХ ЗАПИСІВ У БЛОКЧЕЙНІ ETHEREUM

В. В. Просолов, О. С. Кушнерьов, В. Є. Сокол, Р. В. Трофименко

Анотація. Актуальність. Шахрайські активності у блокчейні Ethereum становлять суттєвий ризик для децентралізованих фінансів і потребують моделей, здатних не лише реагувати на вже виявлені зловживання, а й проактивно ідентифікувати підозрілі облікові записи до ескалації збитків. **Предметом дослідження** є застосування графових і часових нейронних моделей до задачі класифікації акаунтів Ethereum на доброчесні та шахрайські з урахуванням структурних зв'язків між адресами й динаміки транзакцій у часі. **Метою статті** є розроблення та експериментальна оцінка нейронної архітектури на основі багатошарового персептрона як базового компонента для подальшої інтеграції графових і часових механізмів, а також аналіз її ефективності на відкритому датасеті Ethereum Fraud Detection з високим класовим дисбалансом. **Були отримані наступні результати.** Побудовано базову глибинну модель для бінарної класифікації облікових записів із використанням попередньої обробки ознак, стратифікованого поділу вибірки, балансування ваг класів, регуляризації L2, Dropout і ранньої зупинки, що дало змогу досягти значення ROC AUC близько 0.98 за умов значної переваги "безпечного" класу. Детальний аналіз матриці неточностей та метрик precision, recall, F1 показав прийнятний компроміс між зменшенням хибнопозитивних спрацювань і мінімізацією частки пропущених шахрайських акаунтів, що є критичним для реальних фінансових сценаріїв. **Висновок**. Результати свідчать, що коректно спроєктована базова нейронна модель на табличних ознаках здатна забезпечити високу якість проактивної ідентифікації шахрайських облікових записів в Ethereum та слугувати відправною точкою для подальшої інтеграції графових і часових архітектур, орієнтованих на покращення інтерпретованості й стійкості до еволюції шаблонів зловмисної поведінки.

Ключові слова: блокчейн Ethereum; шахрайство; графові нейронні мережі; часові моделі; ідентифікація ризиків; машинне навчання; транзакції.