IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 June 2024, accepted 23 June 2024, date of publication 26 June 2024, date of current version 9 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3419152

RESEARCH ARTICLE

Graph Anomaly Detection With Disentangled Prototypical Autoencoder for Phishing Scam Detection in Cryptocurrency Transactions

JUNHA KANG AND SEOK-JUN BUU, (Member, IEEE)
Department of Computer Science, Gyeongsang National University, Jinju 52828, South Korea

Corresponding author: Seok-Jun Buu (sj.buu@gnu.ac.kr)

This work was supported by Pohang Iron and Steel Company (POSCO) Technical Research Laboratories under Grant 2024P023.

ABSTRACT As the popularity of cryptocurrencies grows, the threat of phishing scams on trading networks is growing. Detecting unusual transactions within the complex structure of these transaction graphs and imbalanced data between Benign and Scams remains a very important task. In this paper, we present Disentangled Prototypical Graph Convolutional Autoencoder, which is optimized for detecting anomalies in cryptocurrency transactions. Our model redefines the approach to analyzing cryptocurrency transactions by treating them as edges and accounts as nodes within a graph neural network enhanced by autoencoders. The DP-GCAE model differentiates itself from existing models by implementing disentangled representation learning within its autoencoder framework. This innovative approach allows for a more nuanced capture of the complex interactions within Ethereum transaction graphs, significantly enhancing the ability of the model to discern subtle patterns often obscured in imbalanced datasets. Building upon this, the autoencoder employs a triplet network to effectively disentangle and reconstruct the graph. Reconstruction is used as input to Graph Convolutional Network to detect unusual patterns through prototyping. In experiments conducted on real Ethereum transaction data, our proposed DP-GCAE model showed remarkable performance improvements. Compared with existing graph convolution methods, the DP-GCAE model achieved a 37.7 percent point increase in F1 score, validating the effectiveness and importance of incorporating disentangled learning approaches in graph anomaly detection. These advances not only improve the F1-score of identifying phishing scams in cryptocurrency networks, but also provide a powerful framework that can be applied to a variety of graph-based anomaly detection tasks.

INDEX TERMS Graph autoencoder, anomaly detection, triplet network, graph neural network, representation learning.

I. INTRODUCTION

The advent and proliferation of cryptocurrencies have dramatically transformed the financial landscape, offering benefits like independent transaction methods, rapid transfer speeds, and lower fees [1], [2], [3]. However, inherent characteristics of cryptocurrencies such as anonymity and decentralization [4], [5], [6], while providing numerous advantages, also leave these networks vulnerable to sophisticated phishing scams, posing significant challenges in ensuring secure transactions [7], [8], [9], [10]. This escalating threat underscores the urgent need for advanced and reliable fraud detection methods in cryptocurrency transactions.

Recognizing this issue, our research focuses on addressing the inherent vulnerabilities in cryptocurrency transactions. The transparency of the blockchain technology, particularly within the Ethereum network, allows anyone to access all transaction records [11]. This accessibility provides a complete dataset for researching various Ethereum users, offering an invaluable resource for our study. The primary

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa M. Fouda.

challenge lies in the difficulty of distinguishing between legitimate and fraudulent transactions within the complex and often imbalanced data of transaction graphs [12], [13]. Traditional graph analysis methodologies reveal inadequacies in effectively identifying these subtle differences, thereby highlighting the necessity for a more sophisticated approach. Particularly, the patterns of fraudulent transactions often mimic those of legitimate ones, making detection of scams challenging. Developing advanced analytical tools capable of detecting and analyzing these similarities is a crucial research task in this field. Figure 1, Table 1 shows transaction patterns according to benign and scam nodes. The transaction patterns of benign nodes (Figures d, e, f) and scam nodes (Figures a, b, c) are very similar. Similarity in transaction patterns between benign and scam nodes is difficult to detect, making the proposed methodology very important to solve this problem.

We address the escalating challenge of scams within cryptocurrency transaction networks. One of the critical issues in these networks is the subtle distinction between benign and scam transactions, which are often masked by the complex nature of the transaction graphs. To address nuanced challenge, we have developed the Disentangled Prototypical Graph Convolutional Autoencoder (DP-GCAE) model. The DP-GCAE is specifically designed to detect anomalous activities in these complex networks. The model treats transactions as edges and accounts as nodes, utilizing graph neural networks and autoencoders to process and analyze the data. The core mechanism of DP-GCAE involves an autoencoder that disentangles the graph's features through a triplet network and then reconstructs the graph by processing the features of nodes and edges. The reconstructed graph, processed through the autoencoder, then serves as input to a Graph Convolutional Network (GCN). Then, the model utilizes a prototyping approach to detect unusual patterns, excelling in distinguishing between legitimate and fraudulent transactions within the Ethereum network. This approach is particularly effective in handling the complexities of the transaction graph, making the DP-GCAE model a potent tool for identifying scams in cryptocurrency transactions.

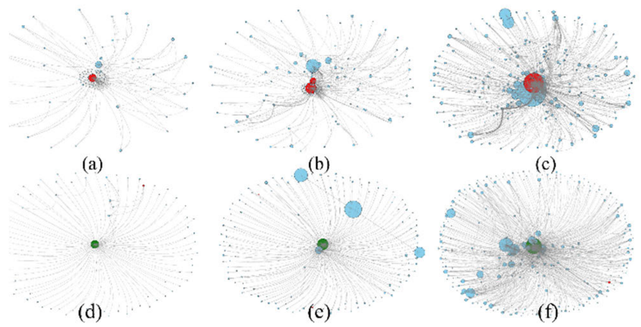


FIGURE 1. Scam and benign patterns according to similar number of nodes and edges.

Our study addresses the inherent vulnerabilities in cryptocurrency transactions. A key obstacle is the complexity of distinguishing between legitimate and fraudulent transactions amidst the often-imbalanced data characteristic of transaction graphs.

TABLE 1. Distribution of scam and normal nodes and edges.

Scam Benign	(a) (d)	(b) (e)	(c) (f)
	Node, edge distribution		
Nodes	100-150	150-200	600-650
Edges	50-80	100-150	300-350

The contributions of our work can be summarized as follows:

1) Advanced Disentanglement: The DP-GCAE's innovative use of disentangled prototyping approaches with triplet networks has significantly improved the ability of the model to identify and classify scams in cryptocurrency networks, marking a substantial advancement in graph-based anomaly detection methods.
2) Empirical Validation: Demonstrated through exhaustive testing with real Ethereum transaction data, the DP-GCAE model exhibits remarkable improvements over traditional detection methods. It proves its efficacy in analyzing and classifying transactions within extensive cryptocurrency networks, underscoring its practical relevance and superior performance.

The rest of paper is structured as follows: Section II provides an overview of related research in graph data analysis, focusing on the trends and methodologies prevalent in this field. Section III details our proposed model, DP-GCAE, explaining its design and functionalities. In Section IV, we compare the performance of our DP-GCAE model against various conventional methods, showcasing its efficacy in anomaly detection within cryptocurrency transactions. Finally, Section V offers a discussion on our findings and conclusions drawn from this research.

II. RELATED WORKS

In the area of transaction security within blockchain networks, research trends have described two main defense mechanisms: anomaly detection and classification. However, despite these advances, existing methodologies often struggle with the high complexity and evolving nature of fraud in blockchain networks, particularly in Ethereum transactions. This complexity frequently results in false positives or negatives, undermining the effectiveness of traditional methods. Table 2 summarizes existing defense mechanisms and methods for graph data.

Anomaly detection techniques, such as Graph Autoencoder (GAE) [14], base their structure on autoencoders that map input data into a latent space before reconstructing it back into the original space. This approach calculates anomaly factors by comparing the original and reconstructed features of each node, identifying deviations from standard patterns. Additionally, methods like the Heterogeneous

Transaction Subnet-based Graph Convolutional Network (HTSGCN) [15] considers the heterogeneity of Ethereum transaction data, using different weight matrices based on transaction types and directions. This method employs GCN layers to aggregate features from neighboring nodes, capturing the complex relationships inherent in Ethereum transactions.

TA-Struc2Vec [16] employs a multi-layer weighted graph to encode structural similarities between nodes, using the Dynamic Time Warping (DTW) algorithm for similarity assessment. This technique builds a structural context for each node, allowing the model to learn patterns that are indicative of fraud. Dual-Augment Graph Neural Network (DAGNN) [17] operates on a dual-path framework incorporating Disparity Augment (DA) and Similarity Augment (SA) paths. The DA path highlights differences, while the SA path emphasizes similarities between nodes. This model combines information from both paths to enhance anomaly detection. Ego-graph embedding [18] is designed for Ethereum transaction data, using individual account-centric ego-graphs and relabeling strategies based on Ethereum transaction attributes like transaction amounts, counts, and directions. This method focuses on the local network structure around each node.

Similarly, Local and Global Aware Memory-Based Graph Neural Network (LGM-GNN) [19] generates relation-aware embeddings for nodes and fuses local and global information through its memory networks. This approach allows the model to maintain a comprehensive view of the graph structure, which is crucial for detecting complex fraud patterns.

Moreover, advancements in graph-based anomaly detection and classification have been made with Neural Meta-Graph Search (NGS) [20], optimizes message passing structures for graph-based fraud detection by using meta-graphs to finely control information aggregation across various relations. NGS's meta-graph approach also improves interpretability, offering insights into detected anomalies. Similarly, the S_HGTNs model [21] has shown notable efficacy in detecting anomalous activities within Ethereum smart contracts. This approach constructs Heterogeneous Information Networks from complex smart contract data, extracting features based on meta paths and generating node embeddings with a transformer network.

The Explainable fraud detection framework [22] innovatively uses Multi-modal Information Graphs (MIG) and example-based explanations to assist in understanding detected frauds. This method indexes and extracts similar and diverse fraud subgraphs from past cases, providing a new perspective on fraud detection in complex networks like social media. Addressing the challenge of imbalanced data distributions in multi-relation graphs, the IMINF model [23] introduces an Imbalanced and Interactive Learning Framework that uses feature mapping modules and relationship aggregation to distinguish fraudulent actors effectively. IMINF's interactive learning component allows it to continuously improve its performance as it encounters new data. The MS_HGNN model [24] takes a hybrid approach to online transaction fraud detection, focusing on mitigating the imbalance issues inherent in graph-based data. This model combines hybrid sampling techniques with reinforcement learning-based reward/punishment mechanisms to optimize relational sampling weights.

GTN2Vec [25], designed to detect money laundering activities within the Ethereum network, integrates gas prices and timestamps as new weights to adjust random walk sampling tendencies. This method leverages these transaction-specific features to better capture the behavior patterns of money laundering nodes, enhancing the accuracy of the detection process. For phishing scam detection, the SIEGE model [26] employs self-supervised incremental deep graph learning to process and analyze large-scale Ethereum transaction data. It utilizes spatial and temporal pre-text tasks to learn high-quality node embeddings that capture the intricate details of phishing scams.

In addition, TTAGN [27] utilizes LSTM, attention mechanisms, and structural enhancements to detect phishing addresses. This framework is tailored for analyzing transaction patterns and interactions within the Ethereum blockchain. Lastly, the PDTGA method [28] implements temporal graph attention mechanisms specifically designed for phishing detection within Ethereum. This method focuses on the temporal characteristics of nodes and edges, allowing it to accurately identify phishing accounts by understanding how fraudulent activities evolve over time. These findings extend our understanding of the diverse methods and approaches employed for anomaly detection and classification within Ethereum and similar complex networks, highlighting the continual evolution and sophistication of these models.

In summary, existing methods have provided a solid foundation for the detection and classification of anomalies within blockchain networks. However, there remains an evident gap in crafting a comprehensive solution equipped to dynamically adapt to the ever-changing nature of blockchain fraud. While some related works have approached aspects of this challenge, they often lack an integrated framework that combines anomaly detection with classification in a manner that adapts to the shifting patterns of fraud.

Our research addresses this gap by proposing a singular, adaptive framework that unites these two critical facets. By implementing an Autoencoder for the initial detection of anomalies, followed by a GCN for robust classification, our hybrid method is designed to meet the complex challenges presented by the security of Ethereum transactions. To conclude, our study augments the existing corpus of research by providing a nuanced perspective on the synergy between anomaly detection and classification in blockchain networks. Our comparative analysis of various methods and their performance on diverse datasets reveals the integral role of an adaptive, integrated approach [29]. This strategy does not merely enhance the F1-Score for anomaly detection in Ethereum transactions but also charts a course for future

TABLE 2. Summary of defense mechanism and methods for graph data.

Defense Mechanism	Method	Dataset	Performance	year
Classification	Graph Neural Network [30]	Ethereum transaction	F1-Score 0.95	2021
Anomaly Detection	Graph Auto-encoder [31]	QCD boost-ed jet	AUC 0.84	2021
Anomaly Detection	GAE [14]	Benchmark	AUC 0.96	2022
Classification	DAGNN [17]	Amazon, YelpChi	F1-Score 91.47, 70.61	2022
Classification	Ego-graph Embedding [18]	Ethereum transaction	F1-Score 0.82	2022
Classification	NGS [20]	Amazon, YelpChi	F1-Score 0.9228 (±0.0046)	2022
Classification	S_HGTNs [21]	Ethereum transaction	F1-Score 0.8900 (±1.430)	2022
Classification	HTSGCN [15]	Ethereum transaction	F1-Score 0.85	2023
Classification	TA-Struc2Vec [16]	Ethereum transaction	F1-Score 0.74	2023
Classification	LGM-GNN [19]	Amazon, YelpChi	AUC 96.57, 82.83	2023
Classification	Explainable fraud detection [22]	Books, Amazon, YelpChi	F1-Score 0.9000	2023
Classification	IMINF [23]	Amazon, YelpChi	F1-Score 94.86	2023
Classification	MS_HGNN [24]	Amazon, YelpChi	AUC 0.9430	2023
Classification	GTN2Vec [25]	Ethereum transaction	F1-Score 0.9590	2023
Classification	SIEGE [26]	Ethereum transaction	F1-Score 0.720 ± 0.01	2023
Classification	TTAGN [27]	Ethereum transaction	F1-Score 0.8200	2023
Classification	PDTGA [28]	Ethereum transaction	F1-Score 0.8423	2023

innovations in the domain of blockchain security. Our work is pivotal as blockchain technology evolves, contributing to the critical discourse on safeguarding the integrity and safety of digital transactions at an international scale.

III. PROPOSED METHOD

A. OVERVIEW OF THE PROPOSED METHOD

Our proposed model, DP-GCAE, offers a novel approach to detecting anomalies in cryptocurrency trading networks. DP-GCAE integrates the concepts of disentangled representation learning with a prototyping network, utilizing a unique triplet network architecture. This integration is key to achieving a more nuanced and sophisticated analysis of Ethereum transaction data, enabling the model to effectively distinguish between benign and scam transactions.

In DP-GCAE, the Ethereum network is represented as a graph $G = \{V, E\}$, where V denotes accounts ($V = \{v_1, v_2, \ldots, v_n\}$), with each v_i representing an individual account, and E denotes transactions ($E = \{e_1, e_2, \ldots, e_m\}$). The architecture of the model, as shown in Figure 2, processes transaction data through two critical stages: disentanglement and prototype formation. The disentanglement stage uses the triplet network to separate the key features of the graph data, which helps in isolating the characteristics of scam transactions from benign ones.

This process of disentangling is crucial for focusing on relevant patterns within the data. The prototype formation stage of DP-GCAE then uses these disentangled features to create prototypes that are representative of both normal and anomalous transaction patterns. These prototypes aid in effectively classifying transactions by comparing new data points with the established patterns.

B. DISENTANGLED GRAPH CONVOLUTIONAL AUTOENCODER

1) AUTOENCODER-BASED ANOMALY DETECTION MECHANISM

The Graph Convolutional Autoencoder (GCAE), a core component of our DP-GCAE model, is specifically designed to process graph-structured data. It leverages the unique properties of graphs to effectively encode relational information, playing a pivotal role in anomaly detection within cryptocurrency transaction networks. The autoencoder operates by encoding the input data into a latent space of lower dimensions, focusing on capturing the essential characteristics of the graph data. During the encoding phase, the autoencoder identifies and distills significant features from the transaction data, preserving vital structural information that is critical in understanding the nature of transactions within the Ethereum network.

In Figure 3, GCN use the same parameters and optimization function in the encoder and decoder. GCN in both the encoder and decoder phases offers feature consistency, which is essential for maintaining the integrity of the graph's structural information throughout the encoding and decoding processes. Moreover, it enhances learning efficiency by allowing the model to symmetrically apply learned transformations, thus improving the overall model performance by more effectively leveraging the graph's inherent properties. Once the data is encoded into the latent space, the decoding phase begins. Here, the autoencoder attempts to reconstruct the original graph structure from the distilled features. This phase is crucial as it tests the ability of the model to accurately capture and represent the data's inherent characteristics.

The main goal of reconstruction process is to minimize the error between the original and reconstructed data, as represented by the optimized objective function in Equation (1). θ represents the parameters of the autoencoder, while $f_\theta(x_i)$ denotes the reconstructed data output by the model. The objective is to minimize the sum of squared differences between the original data x_i and the $f_\theta(x_i)$.

$$\begin{aligned}\theta^* &= argmin_\theta \sum_{x_i \in X_{train}} L_D(\theta; x_i) \\ &= argmin_\theta \sum_{x_i \in X_{train}} ||f_\theta(x_i) - x_i||^2 \end{aligned} \tag{1}$$

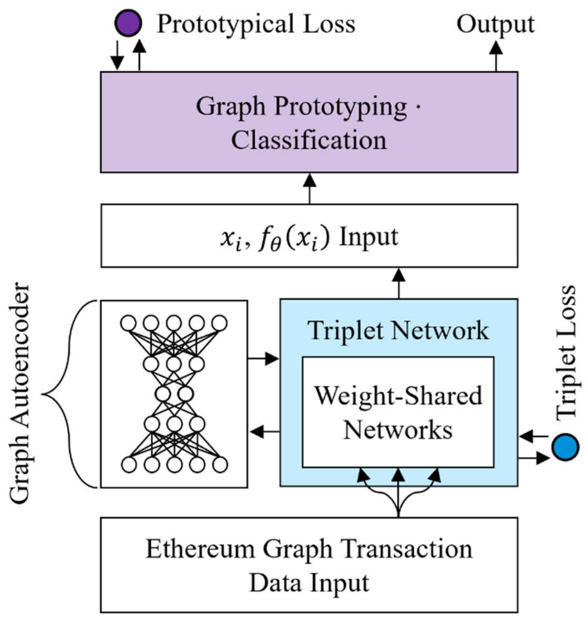

Prototypical Loss
Output
Graph Prototyping · Classification
$x_i, f_\theta(x_i)$ Input
Graph Autoencoder
Triplet Network
Weight-Shared Networks
Triplet Loss
Ethereum Graph Transaction Data Input

FIGURE 2. Overview of the disentangled prototypical graph autoencoder (DP-GCAE) architecture.

Benign data generally shows higher reconstruction accuracy than scam data, as detailed in [32]. This distinction aids in flagging transactions with significant reconstruction discrepancies as potential anomalies, leveraging the difference in reconstruction quality to identify deviations from standard transaction patterns. By employing autoencoder-based anomaly detection mechanism, our DP-GCAE model efficiently identifies transactions that deviate from the norm, thereby enhancing its capability to detect within the Ethereum blockchain. This approach represents a significant advancement in the field of graph-based anomaly detection, offering a more sophisticated and effective tool for identifying potential security threats in cryptocurrency transactions.

Algorithm 1 Triplet Selection for Graph Autoencoder Training

```
Input:
- Node labels (data.y)
- Graph data (data.x, data.edge_index)
Output:
- Triplets tensor (anchor, positive, and negative nodes)
Main Process:
1: Initialize Variables (Identify unique labels = data.y)
2: Generate Triplets:
3: Identify Anchor and Positive Nodes
     label_indices = (data.y == label).nonzero(as_tuple=False).view(-1)
4: Identify Negative Nodes
     other_indices = (data.y != label).nonzero(as_tuple=False).view(-1)
5: Two nodes for the anchor/positive and at least one node for the negative
   selection within each label
6: Form Positive Pairs and Select Negatives
   positive_pairs = list(combinations(label_indices.cpu(). numpy(), 2))
7: triplets = []
   for pair in positive_pairs:
   anchor, positive = pair
   negative = random.choice(other_indices).item()
   triplets.append([anchor, positive, negative])
8: Output Triplets:
   triplets_tensor = torch.tensor(triplets, dtype=torch.long)
```

2) LEARNING GRAPH REPRESENTATION VIA DISENTANGLED GRAPH AUTOENCODER

DP-GCAE introduces a disentanglement strategy that employs a triplet network to enhance the ability of the model to differentiate and isolate transactional patterns within the Ethereum network. This function refines the embedding space to distinctly separate nodes of similar transactional behavior from those that differ, a fundamental step in accurately identifying anomalies.

In Figure 3, the DP-GCAE architecture begins with the input data, which represents transactional activities within the network. These transactions are first processed through a searching algorithm, such as Breadth-First Search (BFS). Once the subgraphs are determined, the data is passed to the triplet network, composed of anchor (g_a), positive (g_P), and negative (g_n) nodes. z is used to reconstruct the original graph structure, enabling us to discern and classify anomalies based on their reconstruction errors.

Algorithm 1 is described as the process used for sampling triplets—composed of anchor, positive, and negative nodes—critical for training the GAE. This algorithm specifically selects anchors and positives from the same class to underscore similarities in transactional characteristics, while negatives are chosen from different classes to introduce the necessary contrasts and complexities. The design and execution of Algorithm 1 are thus central to the GAE in learning discriminative and meaningful node embeddings that are essential for accurate and effective anomaly detection within transaction networks.

The triplet loss function ($L_{triplet}$) is then applied to this network to optimize the process of disentangling the transaction patterns. The core idea behind the triplet loss function is to learn a representation in which transactions of the same class are brought closer together, while those from different classes are pushed further apart in the embedding space. The mathematical formulation of the triplet loss function, expressed in Equation (2), ensures that the distance between the anchor and the positive nodes (similar transactions) is minimized, while the distance between the anchor and the negative nodes (dissimilar transactions) is maximized, exceeding a defined margin α. The margin α is a critical hyperparameter that governs the separation between matched (anchor-positive) and unmatched (anchor-negative) pairs in the latent space. Equation (2), plays a pivotal role in fine-tuning the sensitivity of the model to the diverse spectrum of transactional behaviors, thereby significantly augmenting its efficacy in isolating, and identifying anomalies with heightened accuracy.

$$L_{triplet} = L(a, p, n) = \sum_{i}^{N} [||f(x_i^a) - f(x_i^p)||^2 - ||f(x_i^a) - f(x_i^n)||^2 + \alpha] \tag{2}$$

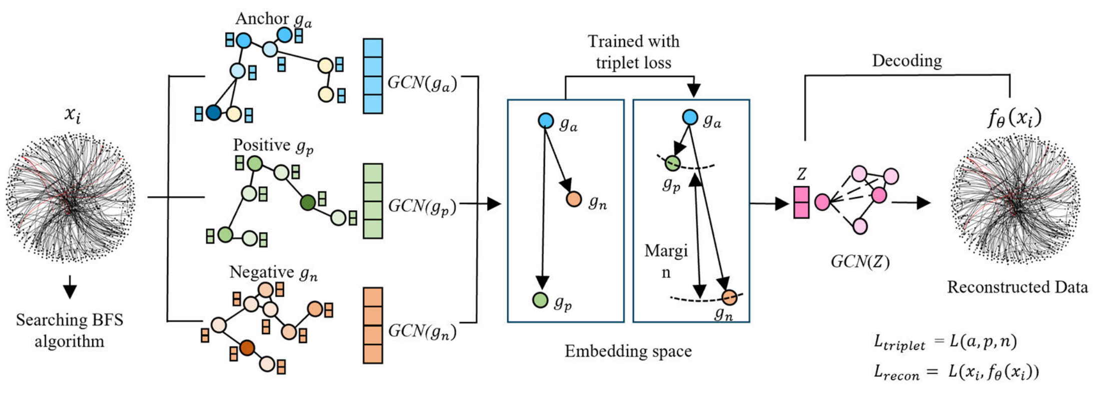


FIGURE 3. Disentangled graph convolutional autoencoder.

where $f\left(x_i^a\right)$, $f\left(x_i^p\right)$ and $f\left(x_i^n\right)$ represent the embeddings for the anchor, positive, and negative nodes respectively, as clarified in Equation (3):

$$\forall(f\left(x_i^a\right), f\left(x_i^p\right), f\left(x_i^n\right)) \in T \tag{3}$$

The anchor and the positive nodes are transactions that belong to the same class, reflecting similar transactional patterns, whereas the negative node represents a transaction that diverges from the anchor, illustrating a different behavior or class. The effectiveness of the triplet loss function hinges on its ability to simultaneously pull together similar transactions while pushing apart those that are dissimilar, beyond the margin α. This fosters a robust embedding space where transactions are organized in a manner that intuitively reflects their underlying relationships and patterns, thereby facilitating the detection of anomalies based on the learned representations.

Algorithm 2, detailed in the manuscript, plays a critical role in dynamically updating the loss weights during the training of the GAE. This algorithm ensures that the emphasis is appropriately balanced between the accuracy of reconstruction and the efficacy of disentanglement, as clarified in Equation (4):

$$\begin{aligned} L &= w_{recon} \cdot L_{recon} + w_{triplet} \cdot L_{triplet} \\ &= w_{recon} \cdot \sum_{x_i \in X_{train}} ||f_\theta\left(x_i\right) - x_i||^2 \\ &\quad + w_{triplet} \cdot \sum_{i}^{N} [||f\left(x_i^a\right) - f\left(x_i^p\right)||^2 - ||f\left(x_i^a\right) \\ &\quad - f\left(x_i^n\right)||^2 + \alpha] \end{aligned} \tag{4}$$

It calculates the change in loss between iterations, subtly shifting the balance towards the reconstruction or triplet loss based on which aspect of the model requires refinement. This real-time adjustment, informed by the loss history, allows for a nuanced tuning of the model, which actively responds to the evolving complexity of the data it processes. The algorithm's role is to adjust the contribution of reconstruction and triplet losses, refining the focus of the model during the learning process.

Within this framework, w_{recon} represents the updated weight for the reconstruction loss, whereas $w_{triplet}$ denotes the updated weight for the triplet loss, calculated as $w_{triplet} = 1 - w_{recon}$. Furthermore, L_{recon} symbolizes the reconstruction loss, and $L_{triplet}$ signifies the triplet loss. The term α denotes the margin, integral to the calculation of the triplet loss. By leveraging this loss function, DP-GCAE effectively captures and magnifies the subtle distinctions between typical transaction patterns, enabling the model to detect anomalies with enhanced F1 score.

C. PROTOTYPICAL GRAPH CONVOLUTIONAL NEURAL NETWORK

1) LEARNING ABNORMAL THRESHOLD VIA GRAPH CONVOLUTIONAL NETWORK

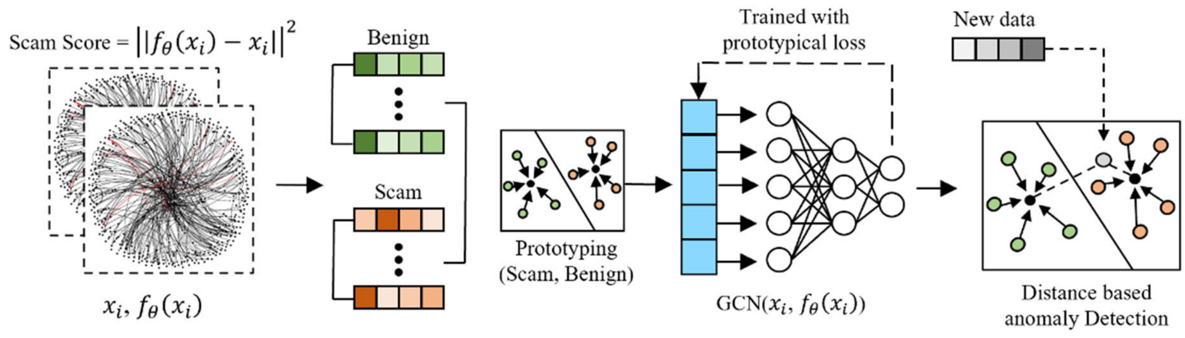
The GCN within our DP-GCAE model is a sophisticated component that significantly enhances the capability of the model to detect anomalies. The process begins with the calculation of a 'Scam Score' for each transaction, a metric derived as per Equation (5). This score quantitatively reflects the discrepancy between original transactions and their reconstructed counterparts, serving as a preliminary indicator of potential anomalies. Once obtained, these scam scores, alongside the transaction data, are inputted into the GCN.

$$ScamScore = ||f_\theta\left(x_i\right) - x_i||^2 \tag{5}$$

The GCN's architecture is adept at capturing the relational nuances present in the transaction graph by leveraging the inherent graph structure. It aggregates and synthesizes information from each node's local neighborhood, leveraging the structural properties of the graph.

Through such a process, the GCN progressively refines the representations of nodes across its layers, enhancing the transactional features iteratively. This methodical enhancement of features empowers the GCN to extract pivotal information from the complex transaction network, which is crucial for the precise classification of transactions. Equation (6) outlines how the learned weights in the GCN's layers contribute to this feature aggregation process:

$$h_i^{(l+1)} = \sigma(\sum\nolimits_{j \epsilon N(i)} \frac{1}{C_{ij}} W^{(l)} h_j^{(l)}) \tag{6}$$



Scam Score = $||f_\theta(x_i) - x_i||^2$
Benign
Scam
$x_i, f_\theta(x_i)$
Prototyping (Scam, Benign)
Trained with prototypical loss
GCN($x_i, f_\theta(x_i)$)
New data
Distance based anomaly Detection

FIGURE 4. Prototypical graph convolutional neural network.

where $h_i^{(l+1)}$ denotes the feature representation of node i at layer $l+1$, σ represents a non-linear activation function, $W^{(l)}$ is the weight matrix at layer l, $N(i)$ is the set of neighboring nodes of i, and C_{ij} is the normalization constant for edge (i, j). This equation is critical for integrating the information from a node's neighbors into its updated feature representation, enhancing the ability of the model to discern nuanced patterns indicative of fraudulent activities.

To address the optimization of the threshold for anomaly detection, we introduce Equation (7), which formalizes the optimization objective for determining the threshold value:

$$\theta^* = argmin_\theta \sum_{x_i \in X_{train}} (L_{anomaly}(\theta; x_i) + \lambda ||\theta||^2) \quad (7)$$

where $L_{anomaly}$ represents the loss function specific to anomaly detection, λ is the regularization parameter, and θ encapsulates the parameters of the GCN model, including weights and biases. This optimization seeks to balance the sensitivity of the model to anomalies with the need to minimize overfitting, ensuring robustness across diverse transaction patterns. Through such a process, the GCN progressively refines the representations of nodes across its layers, iteratively enhancing transactional features. This methodical enhancement of features empowers the GCN to extract pivotal information from the complex transaction network, crucial for the precise classification of transactions.

GCN's role extends to learning the threshold that delineates normal behavior from potential scams. By employing an optimized threshold value, the network ensures that the classification of transactions is not only accurate but also reliable, minimizing the incidence of false positives and negatives. Furthermore, this approach of setting a dynamic threshold allows the model to adjust its sensitivity based on the evolving landscape of the blockchain transactions, ensuring that it remains robust against both known and emerging fraudulent patterns.

2) PROTOTYPING OF CRYPTOCURRENCY TRANSACTIONS

In the DP-GCAE model, the GCN's prototyping approach is operationalized through the implementation of a prototype network, which serves to categorize nodes within the transaction network into distinct classes. This approach is particularly effective in capturing the essence of transactional behavior and is central to the ability of the model to discern between different categories of transactions. To clarify, prototyping involves the creation of a representative 'prototype' for each class of transaction within the network. These prototypes are essentially the centroid vectors in the embedding space, calculated as the mean of all node embeddings belonging to the same class. For each class c, a prototype P_c is derived as the mean vector of the embeddings of all nodes belonging to that class, as shown in Equation (8):

$$P_c = \frac{1}{|C_c|} \sum\nolimits_{x \in C_c} f(x) \quad (8)$$

where $|C_c|$ represents the number of nodes within class C, and $f(x)$ denotes the embedding function mapping nodes to the embedding space. Equation (8) represents a foundational step in distilling the core characteristics of each transaction class. This process not only facilitates a clear demarcation between different transactional behaviors but also enhances the interpretability of the model by providing tangible reference points within the embedding space.

Algorithm 2 Updating Loss Weights in Graph Autoencoder Training

```
Input:
- Reconstruction loss(recon_loss) history and Triplet loss(t_loss) history
- Current reconstruction loss and triplet loss
Output:
- Updated weights for reconstruction and triplet losses
(new_recon_weight, new_t_weight)
Main Process:
1: set initial weights: return 0.5, 0.5.
2: Calculate the difference between the current loss and the previous loss for
both recon and triplet:
3:    delta_recon_loss = current_recon_loss - recon_loss_history[-1]
4:    delta_t_loss = current_t_loss - t_loss_history[-1]
5: Adjust the weights based on the change in losses:
6:    if delta_recon_loss > delta_t_loss
7: Increase the weight of t_loss and decrease the weight of recon_loss.
8        adjustment = min (0.05, delta_recon_loss - delta_t_loss)
9        new_recon_weight = max (0.1, recon_loss_weight - adjustment)
10: else (triplet loss has increased more than reconstruction loss):
11:    Increase the weight of recon_loss and decrease the weight of t_loss.
12:        adjustment = min (0.05, delta_t_loss - delta_recon_loss)
13:        new_recon_weight = min (0.9, recon_loss_weight + adjustment)
14: Calculate the new weight for t_loss:
15:    new_t_weight = 1 - new_recon_weight
16: Return the updated weights: return new_recon_weight, new_t_weight
```

In Equation (9), where each prototype is adjusted to account for variance within the class:

$$P'_c = P_c + \alpha\left(\frac{1}{|C_c|}\sum\nolimits_{x\in C_c}(f(x) - P_c)^2\right) \tag{9}$$

where, α is a scaling factor that adjusts the influence of intra-class variance, providing a more representative prototype that captures not just the average, but also the spread of the node embeddings within each class. This nuanced consideration ensures that our prototypes are not merely static centers but dynamic entities that reflect the true diversity of transactional patterns within each class. The prototypical loss L_{proto} is then calculated to minimize the intra-class variations while maximizing the inter-class separation, utilizing a softmax over distances from class prototypes defined in Equation (10) [33]:

$$L_{proto} = -\sum\nolimits_{c\in classes} log\left(\frac{exp(-d(f(x), P'_c))}{\sum_{c'\in classes} exp\left(-d\left(f(x), P'_{c'}\right)\right)}\right) \tag{10}$$

The prototypical loss function acts as a sophisticated mechanism that finely tunes the sensitivity of the model to the nuances of transactional data. In training, loss function encourages the model to position the embeddings of nodes closer to their corresponding class prototype than to others, thereby achieving a clear distinction between transaction classes in the embedded space. Figure 4 depicts the Prototypical Graph Convolutional Neural Network, highlighting how it uses scam scores to distinguish between normal and anomalous transactions. It shows the process of training the model with prototypical loss and then using it to predict new transactions as either normal or anomalous based on learned prototypes. To operationalize this conceptual framework within the DP-GCAE model, Algorithm 3 delineates a systematic procedure for initializing, computing, and utilizing class prototypes to classify nodes in the Ethereum transaction graph. The process begins with followed by the initialization of class prototypes and the computation of prototypes for each transaction class.

The crucial steps of calculating distances between node embeddings and prototypes, converting these distances to similarity scores, and ultimately classifying nodes based on these scores. This algorithmic flow culminates in the prediction of labels (scam/benign) for each node, underscoring the practical implementation of our prototyping approach. This algorithmic process, along with the prototypical loss function, enables the DP-GCAE model to effectively prototype anomalies in cryptocurrency transactions, leading to a more nuanced detection capability as reflected in the enhanced F1 score.

Algorithm 3 Prototypical Network in Graph Neural Network

```
Input:
- Node embeddings(z), node labels(data.y)
Output:
- Predicted scam/benign labels in Ethereum Transaction Graph
Main Process:
1: Preprocess Node Embeddings:
      Normalize the node embeddings: z_norm = normalize(z)
2: Initialize Prototypes:
      Create a zero tensor for prototypes of each class
      : zeros (2, size of z_norm [1])
3: Calculate Prototypes for Each Class:
      For each class index i (0 for benign, 1 for scam):
         Extract embeddings for class I = z_norm [data.y == i]
         Compute the mean: prototypes[i] = mean(class_embeddings)
4: Compute Distances:
      Calculate the distance between node embeddings and prototypes:
      dists = distance (z_norm, prototypes)
5: Convert Distances to Similarity Scores:
   : similarity_scores = exp(-dists)
6: Classify Nodes:
   Assign each node to the class with the highest similarity:
   predicted_labels = argmax (similarity_scores, axis=1)
7: Output Predicted Labels:
   Return the predicted labels for each node in the graph.
```

IV. EXPERIMENTAL RESULTS

A. DATASET AND PREPROCESING

Our empirical analysis was conducted using data from the Ethereum blockchain, specifically targeting a substantial connected subgraph within the transaction network. The graph is homogeneous, with nodes representing individual transaction accounts, which further adds to the complexity and realism of our study. The rationale behind selecting this dataset, sourced from the comprehensive study by Chen et al. [34], stems from its inherent complexity, realism, and the challenging nature of the data, particularly in terms of class imbalance.

The Ethereum blockchain, with its vast and intricate network of transactions, provides a realistic and challenging environment for testing our DP-GCAE model, making it an ideal candidate for our study. To obtain the connected subgraph, we utilized a series of random walks originating from 1,165 seed nodes. This method was chosen for its effectiveness in capturing a comprehensive yet manageable segment of the network, ensuring a balance between depth and breadth of data. The resulting subgraph, comprising 2,973,382 nodes connected by 13,551,214 edges, presents a challenging yet insightful dataset for anomaly detection due to its scale, complexity, and the prevalence of class imbalances.

TABLE 3. Analysis of statistical characteristics by graph size of the Ethereum transaction network.

Graph size	10,000	20,000	30,000	40,000
Centrality avg	0.0006	0.0003	0.0002	0.0001
Centrality std	0.0020	0.0019	0.0012	0.0013
Maximum degree Centrality	0.1017	0.2084	0.1208	0.1863
Class imbalance	0.1319	0.0619	0.0404	0.0300
Community average rate (%)	0.05%	0.02%	0.01%	0.01%

Table 3, which provides valuable insights into the network's structural properties [35]. It includes data points such as average and standard deviation of centrality, maximum degree centrality, class imbalance, and the community

TABLE 4. Number of parameters for the autoencoder and the graph convolutional network (GCN) models.

Layer (type:depth-idx)	Param #
GAEWithTriplet	- -
├─GCNConv: 1-1	80
│ └─SumAggregation: 2-1	- -
│ └─Linear: 2-2	1,603,200
├─GCNConv: 1-2	40
│ └─SumAggregation: 2-3	- -
│ └─Linear: 2-4	3,200
├─BatchNorm1d: 1-3	160
│ ├─BatchNorm1d: 1-4	80
Total params: 1,606,760	
GCNWithPrototypes	- -
├─GCNConv: 1-1	64
│ └─SumAggregation: 2-1	- -
│ └─Linear: 2-2	1,282,560
├─BatchNorm1d: 1-2	128
├─GCNConv: 1-3	2
│ └─SumAggregation: 2-3	- -
│ └─Linear: 2-4	128
│ ├─BatchNorm1d: 1-4	4
Total params: 1,282,886	

average rate for varying graph sizes (10,000 to 40,000 nodes). Particularly notable is the community average rate, which serves as an indicator of how nodes are distributed across different communities. A lower average rate implies that nodes are dispersed across many diverse communities, reflecting the network's complexity and diversity.

B. IMPLEMENTATION DETAILS AND EVALUATION METRICS

Our experiments were conducted in an environment equipped with NVIDIA GeForce RTX 4060 Ti GPU and Intel Core i5-13400f CPU, running on Windows 11 operating system. For software, we utilized the Anaconda virtual environment with Python version 3.8.18. The deep learning framework PyTorch, version 2.1.0, was our primary tool for modeling. Given the notable class imbalance present in our dataset, our evaluation metrics were centered around precision, recall, and the F1 score.

The F1 score is crucial as it represents the harmonic mean of precision and recall, providing a balanced measure that is especially relevant in scenarios with skewed class distributions. The F1 score is calculated using the formula:

$$F1 = \frac{2 \times precision \times recall}{precision + recall}.$$

Table 4 in our study provides a detailed of the number of parameters for both the Autoencoder and the GCN models. This table highlights the complexity and scale of each model by quantifying the individual components that contribute to their overall architecture. The inclusion of these parameter counts serves to offer a comprehensive perspective on the structural aspects of each model, reflecting their respective computational requirements and intricacies.

C. PERFORMANCE COMPARISION OF DP-GCAE AND OTHER MODELS

Table 5,6 provides a comprehensive evaluation of precision, recall, and F1 scores for various graph sizes (10,000-50,000) nodes. We conduct a comparative analysis of our proposed DP-GCAE method against established graph traversing and embedding techniques. We look at Deep Walk [36], a pioneering graph embedding method that utilizes random walks to capture the sequential nature of paths in a graph, translating the structural information of the graph into a low dimensional space. Next, Node2Vec [37], an extension of Deep Walk. Node2Vec uses a biased random walk procedure to efficiently explore diverse neighborhoods, allowing it to learn richer representations. Another method is LINE [38] designed to preserve both first order and second order proximities in large networks, effectively capturing the structure of the network. In addition, LSTM [39], a type of recurrent neural network known as Long Short Term Memory networks, which are adept at learning from sequences, making them suitable for analyzing graph structures that evolve over time.

Our analysis also includes graph neural networks (GNN) [40], which generalize deep learning concepts to graph data, enabling the learning of node representations through their neighborhood. We compare with GCN [41], Graph Convolutional Networks, which use a convolutional architecture to exploit the natural graph structure, aggregating neighborhood features to learn a node's representation. Another is GAT [42], Graph Attention Networks, introducing an attention mechanism to graph neural networks, allowing for more nuanced weight assignment to neighboring nodes.

TTAGN [27] is a network framework tailored for detecting phishing addresses in the Ethereum blockchain, utilizing a combination of temporal edge representations and advanced node analysis. It employs LSTM to analyze transaction patterns, attention mechanisms for understanding interactions, and structural enhancements for comprehensive node representation. The GA-GNN model [43] applies Neural Architecture Search (NAS) specifically for Graph Neural Networks (GNNs), a domain less explored than traditional data structures like images or text. Through an evolutionary process, GA-GNN efficiently finds optimal configurations, enhancing performance in graph representation learning and node classification. Lastly, we scrutinize a variant of our model, the DP-GCAE without Autoencoder-based anomaly detection [44], to assess its impact on performance.

The table 5,6 demonstrates that our DP-GCAE model achieves higher F1 scores than the existing methods. This trend is evident across all graph dimensions, highlighting the efficacy of our disentangled prototyping approach in classifying transactions within the Ethereum blockchain network. In comparison, the DP-GCAE without Autoencoder-based anomaly detection shows a slight decline in performance, with precision, recall, and F1 scores. This decline underscores the importance of the Autoencoder component in our approach.

TABLE 5. Precision, recall, and F1 score comparison of different methods for graphs of sizes 30,000, 40,000, and 50,000.

Model	**Graph size = 30,000 (Runtime = 79.41s)**			**Graph size = 40,000 (Runtime = 171.72s)**			**Graph size = 50,000 (Runtime = 543.04)**		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Graph traversing/embedding									
Deep Walk [36]	0.1251	0.7108	0.2049	0.1453	0.5754	0.2227	0.1575	0.5945	0.2426
Node2Vec [37]	0.1094	0.6956	0.1832	0.1424	0.6689	0.2267	0.1554	0.6475	0.2426
LINE [38]	0.1409	0.5352	0.2163	0.1332	0.5597	0.2087	0.1726	0.5222	0.2538
LSTM [39]	0.5145	0.9657	0.6713	0.5073	**0.9971**	0.6724	0.4993	**0.9914**	0.6641
Comparatives (graph neural networks)									
GNN [40]	0.8447	0.5536	0.5658	0.6382	0.6267	0.6117	0.6458	0.6284	0.6079
GCN [41]	**0.9000**	0.2800	0.4271	**0.9375**	0.3036	0.4587	0.9022	0.3578	0.5123
GAT [42]	0.8483	0.5662	0.5818	0.6629	0.6485	0.6381	**0.9717**	0.6392	0.6338
TTAGN [27]	0.8550	0.7210	0.7830	0.8330	0.8070	0.8200	0.8590	0.7770	0.8160
GA-GNN [43]	0.8469	0.7743	0.8090	0.8486	0.7686	0.8066	0.8442	0.7743	0.8077
Ours									
DP-GCAE	0.7664	**0.9831**	**0.8614**	0.7303	0.9867	**0.8393**	0.7158	0.9684	**0.8305**
w/o Autoencoder-based Anomaly detection (DP-GCN)	0.6893	0.8042	0.7423	0.8140	0.7609	0.7865	0.8121	0.8089	0.7902

TABLE 6. Precision, recall, and F1 score comparison of different methods for graphs of sizes 10,000, and 20,000.

Model	**Graph size = 10,000 (Runtime 43.78s)**			**Graph size = 20,000 (Runtime = 57.37s)**		
	Precision	Recall	F1	Precision	Recall	F1
Graph traversing/embedding						
Deep Walk [36]	0.1022	0.9789	0.1851	0.1170	0.9789	0.2090
Node2 Vec [37]	0.0834	0.9855	0.1538	0.0877	0.9808	0.1610
LINE [38]	0.1077	0.9553	0.1936	0.1145	0.9893	0.2053
LSTM [39]	0.5014	**0.9886**	0.6654	0.5029	**0.9971**	0.6686
Comparatives (graph neural networks)						
GNN [40]	0.8371	0.4571	0.5913	0.8057	0.4483	0.5761
GCN [41]	**0.9440**	0.2876	0.4217	0.7248	0.3086	0.4329
GAT [42]	0.8584	0.5371	0.6608	0.8213	0.4857	0.6104
TTAGN [27]	0.9023	0.7893	0.8320	**0.8977**	0.6771	0.7720
GA-GNN [43]	0.8480	0.7971	0.8218	0.8429	0.7971	0.8194
Ours						
DP-GCAE	0.8533	0.9884	**0.9159**	0.7644	0.9961	**0.8650**
w/o Autoencoder-based Anomaly detection (DP-GCN)	0.7307	0.7829	0.7559	0.6651	0.8171	0.7333

The enhanced performance of the DP-GCAE across different graph sizes suggests its adaptability and scalability, key factors for models intended for real world applications.

D. PERFORMANCE COMPARISION BASED ON DISENTANGLEMENT AND PROTOTYPICAL APPROACHES

In Table 7, we elucidate the comparative analysis of anomaly detection models based on the presence and absence of disentanglement and prototypical mechanisms. We scrutinize the performance across graph sizes of 5,000, 10,000 and 20,000 nodes.

The models in comparison include (a) devoid of disentanglement (w/o Disentangle), (b) lacking prototypical network components (w/o Prototypical), missing both (w/o Disentangle/Prototypical), and our full DP-GCAE model. The inclusion of disentanglement and prototypical mechanisms distinctly enhances model performance, suggesting

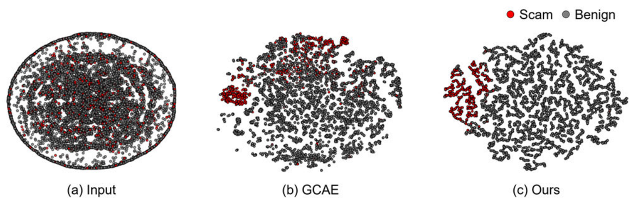


FIGURE 5. Feature space visualization using t-SNE technique for (a) input space, (b) GCAE embedded space, and (c) our DP-GCAE model.

their synergistic effect is pivotal for the nuanced task of anomaly detection. In our experiments, we found that the DP-GCAE model achieved a superior F1 score compared to the other models for all three graph sizes. The findings suggest a pronounced contribution of both disentanglement and prototypical processes in refining the accuracy of fraudulent transaction detection within Ethereum's complex transaction landscape.

TABLE 7. Performance comparison based on disentanglement and prototypical.

Model	Graph size = 5,000 (Runtime = 36.83s)			Graph size = 10,000 (Runtime 43.78s)			Graph size = 20,000 (Runtime = 57.37s)		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
w/o disentanglement	0.9075	0.9626	0.9342	0.7542	0.9825	0.8533	0.6904	0.9882	0.8129
w/o prototyping	**0.9707**	0.9045	0.9365	0.7424	0.9879	0.8478	0.7012	0.9916	0.8215
GCAE	0.8936	0.9251	0.9091	0.7298	0.8889	0.8015	0.6278	0.9826	0.7661
DP-GCAE	0.9266	**0.9796**	**0.9524**	**0.8533**	**0.9884**	**0.9159**	**0.7644**	**0.9961**	**0.8650**

Notably, the removal of disentanglement results in a marked decline in performance, underscoring the critical role of disentangled representations in capturing nuanced patterns of benign and scam transactions. Similarly, the absence of prototypical networking curtails the ability of the model to generalize from representative samples, leading to diminished performance. The compounded effect of omitting both is the most significant, further corroborating the individual contributions of these methods.

E. EFFECTS OF DISENTANGLED PROTOTYPING

To demonstrate the efficiency of our disentangled prototyping approach, DP-GCAE model, t-SNE visualization was performed on various models. Figure 5 provides a visual representation of the feature space derived from (a) the original input, (b) the space embedded via a traditional GCAE, and (c) the refined feature space represented by the DP-GCAE model. In the original input space, features tend to be distributed without any noticeable structure, obfuscating their intrinsic grouping.

The visualization indicates overlapping regions where traditional GCAE methods struggle to discern distinct groups, leading to potential misclassifications. The transition to the GCAE embedded space shows beginnings of clustering. However, the functionality remains complex, blurring the boundaries between different classes. The DP-GCAE models feature space, on the other hand, reveals a clear delineation of clusters, suggesting a significant reduction in intra-class variance and an enhancement in inter-class margin. In conclusion, this stark contrast in the visualizations not only highlights the DP-GCAE's superior feature separation but also affirms the practicality of our approach in extracting meaningful patterns from complex data.

F. ADAPTABILITY OF SCAM DETECTION MODEL

As scams continue to evolve and diversify, the number of available scam samples for training naturally decreases. This reduction poses significant challenges for maintaining model performance in detecting new and varying scam types. To address this, we conducted an experiment to evaluate how our models, DP-GCAE and the baseline GCAE, perform as the number of scam samples diminishes, focusing particularly on changes in the F1 score. This experiment is crucial in testing the adaptability of our models to evolving scam scenarios.

In Figure 6, we observe that both models experienced a decline in F1 scores as the number of scam samples progressively decreased. However, the performance drop was considerably less severe for DP-GCAE compared to GCAE. The substantial decline in GCAE's performance underscores its sensitivity to the reduced availability of scam samples, whereas DP-GCAE demonstrated greater robustness under similar conditions. This resilience of DP-GCAE suggests that the integration of the triplet networks and the prototypical network significantly bolsters the adaptability of the model.

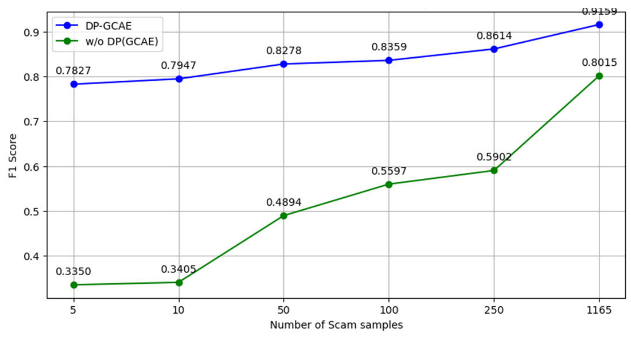


FIGURE 6. F1 score across different scam sample sizes.

G. CASE ANALYSIS OF GCAE AND DP-GCAE

We delve into a comparative case analysis to elucidate the performance distinctions between the Graph Convolutional Autoencoder (GCAE) and our model, DP-GCAE. Two key scenarios are explored: instances where GCAE misclassified but DP-GCAE succeeded, and cases where both models misclassified. Figure 7, Table 8 presents the cases 'Misclassified by GCAE but correct in Ours', which offers an insightful glance into transactions exhibiting a wide range of volumes and frequencies. Notably, despite the subtle disparities between normal and scam transactions, the DP-GCAE demonstrates a remarkable aptitude for accurate prediction. This precise discernment is especially crucial in a field like cryptocurrency, where the cost of misclassification can be high. The enhanced discernment capability of our model, which effectively differentiates between classes that appear indistinguishable to the conventional GCAE, stems from its ability to disentangle and prototype features in a manner that is conducive to high stakes decision making.

H. DISCUSSION

The transaction patterns shown in Figure 8, Table 9 are associated with a much smaller number of edges, suggesting a

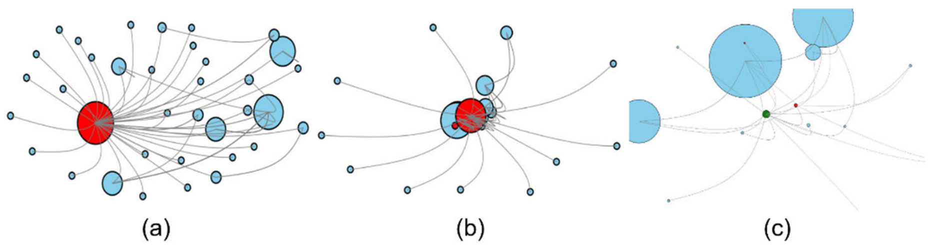


FIGURE 7. Transaction distribution of nodes that are misclassified by GCAE but are correct by Ours.

TABLE 8. Detailed transaction history information of nodes that were misclassified by GCAE but correct by ours.

	Node name	Report-ed as	Edges	Block	From	To	Value (ETH)
Case 1 (Misclassi-fied by GCAE but correct in Ours)	(a)	Phishing scam	Total: 39 in: 39 out: 0	5967270	Binance 2	Self	0.0741
				5964208	Gemini	Self	4.0395
	(b)		Total: 27 in: 15 out: 12	5713890	Self	0x239937C8..	1.0000
				5674417	0x3f5CE5FB..	Self	0.4900
				5674232	0x0681d8Db..	Self	0.5236
	(c)	Benign	Total: 44 in: 16 out: 28	4315810	Fake_Phishing167	Self	0.0100
				4263934	Self	Fake_Phishing58	0.9951
				4263931	Self	Golem Token	0.0000

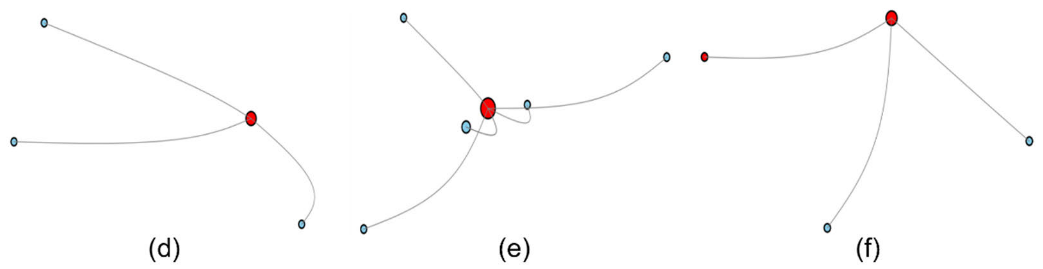


FIGURE 8. Transaction distribution of nodes that misclassified by both GCAE and Ours.

potential correlation between the sparsity of the network and the prediction accuracy of the model. This lack of connectivity can often camouflage transaction patterns, making it challenging for models to discern fraudulent activity. Consequently, the insights gathered from these patterns underscore the importance of incorporating a more holistic view of the transactional ecosystem, including temporal dynamics and transaction flow trends, which could provide further discriminatory power for our model. These observations have led us to consider the integration of additional contextual data, such as transfer destinations and temporal information, to improve identification of complex transaction behaviors. By enriching the model with these layers of context, we anticipate a marked improvement in the ability of the model to detect nuanced discrepancies between benign and scam.

The goal is not only to improve the accuracy of current models, but also to mitigate the risk of misclassification, which can have significant implications for the parties involved. We therefore advocate a balanced approach that accounts for the complexity of trading networks while remaining alert to the ethical implications of detection capabilities. As we continue to evolve the DP-GCN model, we continue to adhere to ethical AI principles to ensure that our efforts to protect the Ethereum trading network do not inadvertently compromise user privacy or fairness.

TABLE 9. Detailed transaction history information of nodes that were misclassified by both GCAE and ours.

	Node name	Reported as	Edges	Block	From	To	Value (ETH)
Case 2 (Both misclassified)	(d)	Phishing scam	Total: 3	6216765	Self	0xA019ae3D..	1.1715
			in: 2	6207897	0x5D64ae45..	Self	0.6724
			out: 1	6206705	0xa9f739d5..	Self	0.5000
	(e)		Total: 6	5508396	Self	0xCbd02526..	0.3116
			in: 3				
			out: 3	5505414	0x3f5CE5FB..	Self	0.3121
	(f)		Total: 3	4315810	Fake_Phishing916	Self	9.0924
			in: 3	4263934	Fake_Phishing1157	Self	20.4013
			out: 0	4263931	0x0845F48B..	Self	0.6300

V. CONCLUDING REMARKS

In this paper, we introduced the Disentangled Prototypical Graph Convolutional Autoencoder (DP-GCAE), a novel approach for detecting phishing scams within cryptocurrency transaction networks. Our model uniquely integrates representation learning via triplet networks and prototyping approach, housed within a graph convolutional autoencoder framework. The comprehensive experiments conducted on Ethereum transaction data demonstrate the significant superiority of DP-GCAE in accurately identifying anomalous transactions, outperforming existing methods by a notable margin.

Our case analyses revealed that, while the DP-GCAE model exhibits robust performance in most scenarios, challenges arise when dealing with transactions having fewer graph connections. This observation points to the potential necessity of incorporating additional contextual data into the model, such as transfer destination information, to better capture the nuances of transaction behaviors. Looking forward, we aim to refine and enhance the DP-GCAE model to address the identified limitations. This includes experimenting with various forms of additional data inputs and exploring advanced network architectures to further improve the accuracy and F1 score of the model.

To conclude, we hope that our contributions in this paper will serve as a steppingstone for future research in this area, ultimately leading to a safer and more secure digital transaction environment. Our goal is to foster an ecosystem where innovations in anomaly detection not only enhance security but also uphold the core values of transparency and integrity that are fundamental to the blockchain and cryptocurrency domains.

REFERENCES

[1] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, and J. Wang, "Untangling blockchain: A data processing view of blockchain systems," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 7, pp. 1366–1385, Jul. 2018.

[2] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond Bitcoin," *Appl. Innov.*, vol. 2, nos. 6–10, p. 71, 2016.

[3] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1495–1505, Apr. 2019.

[4] S. Aggarwal and N. Kumar, "Blockchain 2.0: Smart contracts," *Adv. Comput.*, vol. 121, pp. 301–322, Jan. 2021.

[5] B. Wang, H. Liu, C. Liu, Z. Yang, Q. Ren, H. Zheng, and H. Lei, "BLOCKEYE: Hunting for DeFi attacks on blockchain," in *Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng., Companion (ICSE-Companion)*, May 2021, pp. 17–20.

[6] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[7] D. Guo, J. Dong, and K. Wang, "Graph structure and statistical properties of Ethereum transaction relationships," *Inf. Sci.*, vol. 492, pp. 58–71, Aug. 2019.

[8] M. Weber, G. Domeniconi, J. Chen, D. Karl I. Weidele, C. Bellei, T. Robinson, and C. E. Leiserson, "Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics," 2019, *arXiv:1908.02591*.

[9] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, and Z. Zheng, "Who are the phishers? Phishing scam detection on Ethereum via network embedding," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 2, pp. 1156–1166, Feb. 2022.

[10] S. S. Kushwaha, S. Joshi, D. Singh, M. Kaur, and H.-N. Lee, "Systematic review of security vulnerabilities in Ethereum blockchain smart contract," *IEEE Access*, vol. 10, pp. 6605–6621, 2022.

[11] A. Almomani, M. Alauthman, M. T. Shatnawi, M. Alweshah, A. Alrosan, W. Alomoush, B. B. Gupta, B. B. Gupta, and B. B. Gupta, "Phishing website detection with semantic features based on machine learning classifiers: A comparative study," *Int. J. Semantic Web Inf. Syst.*, vol. 18, no. 1, pp. 1–24, Feb. 2022.

[12] W. Chen, X. Guo, Z. Chen, Z. Zheng, and Y. Lu, "Phishing scam detection on Ethereum: Towards financial security for blockchain ecosystem," in *Proc. IJCAI*, vol. 7, 2020, pp. 4456–4462.

[13] H. Hu, Q. Bai, and Y. Xu, "SCSGuard: Deep scam detection for Ethereum smart contracts," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2022, pp. 1–6.

[14] X. Du, J. Yu, Z. Chu, L. Jin, and J. Chen, "Graph autoencoder-based unsupervised outlier detection," *Inf. Sci.*, vol. 608, pp. 532–550, Aug. 2022.

[15] B. Huang, J. Liu, J. Wu, Q. Li, and D. Lin, "Ethereum phishing fraud detection based on heterogeneous transaction subnets," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2023, pp. 1–5.

[16] R. Li, Z. Liu, Y. Ma, D. Yang, and S. Sun, "Internet financial fraud detection based on graph learning," *IEEE Trans. Computat. Social Syst.*, vol. 10, no. 3, pp. 1394–1401, Jun. 2022.

[17] Q. Li, Y. He, C. Xu, F. Wu, J. Gao, and Z. Li, "Dual-augment graph neural network for fraud detection," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2022, pp. 4188–4192.

[18] Y. Xia, J. Liu, and J. Wu, "Phishing detection on Ethereum via attributed ego-graph embedding," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 5, pp. 2538–2542, May 2022.
[19] P. Li, H. Yu, X. Luo, and J. Wu, "LGM-GNN: A local and global aware memory-based graph neural network for fraud detection," *IEEE Trans. Big Data*, vol. 9, no. 4, pp. 1116–1127, Aug. 2023.
[20] Z. Qin, Y. Liu, Q. He, and X. Ao, "Explainable graph-based fraud detection via neural meta-graph search," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2022, pp. 4414–4418.
[21] L. Liu, W.-T. Tsai, M. Z. A. Bhuiyan, H. Peng, and M. Liu, "Blockchain-enabled fraud discovery through abnormal smart contract detection on Ethereum," *Future Gener. Comput. Syst.*, vol. 128, pp. 158–166, Mar. 2022.
[22] T. T. Nguyen, T. C. Phan, H. T. Pham, T. T. Nguyen, J. Jo, and Q. V. H. Nguyen, "Example-based explanations for streaming fraud detection on graphs," *Inf. Sci.*, vol. 621, pp. 319–340, Apr. 2023.
[23] X. Wang, Z. Liu, J. Liu, and J. Liu, "Fraud detection on multi-relation graphs via imbalanced and interactive learning," *Inf. Sci.*, vol. 642, Sep. 2023, Art. no. 119153.
[24] J. Long, F. Fang, C. Luo, Y. Wei, and T.-H. Weng, "MS_HGNN: A hybrid online fraud detection model to alleviate graph-based data imbalance," *Connection Sci.*, vol. 35, no. 1, Dec. 2023, Art. no. 2191893.
[25] J. Liu, C. Yin, H. Wang, X. Wu, D. Lan, L. Zhou, and C. Ge, "Graph embedding-based money laundering detection for ethereum," *Electronics*, vol. 12, no. 14, p. 3180, Jul. 2023.
[26] S. Li, R. Wang, H. Wu, S. Zhong, and F. Xu, "SIEGE: Self-supervised incremental deep graph learning for Ethereum phishing scam detection," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 8881–8890.
[27] S. Li, G. Gou, C. Liu, C. Hou, Z. Li, and G. Xiong, "TTAGN: Temporal transaction aggregation graph network for Ethereum phishing scams detection," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 661–669.
[28] L. Wang, M. Xu, and H. Cheng, "Phishing scams detection via temporal graph attention network in Ethereum," *Inf. Process. Manage.*, vol. 60, no. 4, Jul. 2023, Art. no. 103412.
[29] M. Ul Hassan, M. H. Rehmani, and J. Chen, "Anomaly detection in blockchain networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 289–318, 1st Quart., 2023.
[30] R. Tan, Q. Tan, P. Zhang, and Z. Li, "Graph neural network for Ethereum fraud detection," in *Proc. IEEE Int. Conf. Big Knowl. (ICBK)*, Dec. 2021, pp. 78–85.
[31] O. Atkinson, A. Bhardwaj, C. Englert, V. S. Ngairangbam, and M. Spannowsky, "Anomaly detection with convolutional graph neural networks," *J. High Energy Phys.*, vol. 2021, no. 8, pp. 1–19, Aug. 2021.
[32] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *Proc. Wireless Telecommun. Symp. (WTS)*, Apr. 2018, pp. 1–5.
[33] J. Wang and Y. Zhai, "Prototypical Siamese networks for few-shot learning," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 178–181.
[34] L. Chen, J. Peng, Y. Liu, J. Li, F. Xie, and Z. Zheng, "Phishing scams detection in Ethereum transaction network," *ACM Trans. Internet Technol.*, vol. 21, no. 1, pp. 1–16, Feb. 2021.
[35] T. Chen, Z. Li, Y. Zhu, J. Chen, X. Luo, J. C.-S. Lui, X. Lin, and X. Zhang, "Understanding Ethereum via graph analysis," *ACM Trans. Internet Technol.*, vol. 20, no. 2, pp. 1–32, May 2020.
[36] X. Huo, M. Li, and Z.-H. Zhou, "Control flow graph embedding based on multi-instance decomposition for bug localization," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 4223–4230.
[37] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD KDD*, 2016, pp. 855–864.
[38] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. ACM WWW*, 2015, pp. 1067–1077.
[39] W. Luo, W. Liu, and S. Gao, "Remembering history with convolutional LSTM for anomaly detection," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2017, pp. 439–444.
[40] R. Tan, Q. Tan, Q. Zhang, P. Zhang, Y. Xie, and Z. Li, "Ethereum fraud behavior detection based on graph neural networks," *Computing*, vol. 105, no. 10, pp. 2143–2170, Oct. 2023.
[41] T. Huang, D. Lin, and J. Wu, "Ethereum account classification based on graph convolutional network," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 5, pp. 2528–2532, May 2022.
[42] S. Ma, J.-W. Liu, X. Zuo, and W.-M. Li, "Heterogeneous graph gated attention network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–6.
[43] M. Shi, Y. Tang, X. Zhu, Y. Huang, D. Wilson, Y. Zhuang, and J. Liu, "Genetic-GNN: Evolutionary architecture search for graph neural networks," *Knowl.-Based Syst.*, vol. 247, Jul. 2022, Art. no. 108752.
[44] S.-J. Buu and H.-J. Kim, "Disentangled prototypical graph convolutional network for phishing scam detection in cryptocurrency transactions," *Electronics*, vol. 12, no. 21, p. 4390, Oct. 2023.

JUNHA KANG received the B.S. degree in computer science from Gyeongsang National University, South Korea, where she is currently pursuing the M.S. degree. Her primary research interests include the application of deep learning to improve security measures in the digital domain, particularly in combating phishing scams in cryptocurrency networks. Her work not only highlights the critical need for innovative approaches in the detection of phishing scams but also paves the way for applying deep learning techniques to a variety of graph-based anomaly detection tasks in the security sector.

SEOK-JUN BUU (Member, IEEE) received the Ph.D. degree in computer science from Yonsei University, South Korea. Since 2023, he has been an Assistant Professor with the Department of Computer Science, Gyeongsang National University. His primary research interests include the application of deep learning to address a wide array of industrial challenges through empirical methods. He focuses on enhancing deep learning applications by incorporating domain knowledge and real-world constraints, as well as optimizing architecture and hyperparameters through traditional AI techniques, such as genetic algorithms.