Heterogeneous Graph Auto-Encoder for Credit Card Fraud Detection

Sudipta Majumder*
Moirangthem Tiken Singh†
Rabinder Kumar Prasad‡
Abhijit Boruah§
Gurumayum Robert Michael¶
N K Kaphungkui‖
N. Hemarjit Singh**

Abstract

The digital revolution has led to increased credit card usage and a corresponding rise in fraudulent activities. Traditional fraud detection methods often overlook the interconnected nature of financial data. This study presents a novel approach using Graph Neural Networks (GNNs) with attention mechanisms and heterogeneous graph structures to enhance credit card fraud detection. The method builds heterogeneous graphs to represent complex interactions among entities such as cardholders, merchants, and transactions. An autoencoder, trained on legitimate transactions, learns latent features to identify anomalies. The model's performance is evaluated using benchmark datasets and compared against existing techniques like GraphSAGE and FI-GRL. Experimental results show that the proposed model achieves superior performance, with an AUC-PR of 0.89 and an F1-score of 0.81. The integration of GNNs, attention mechanisms, and autoencoders effectively mitigates issues like class imbalance and captures intricate data relationships. This research uniquely applies attention-based GNNs on heterogeneous graphs for fraud detection, improving accuracy by addressing class imbalance and leveraging rich relational data. Evaluation is restricted to specific datasets, and real-world deployment may need adaptation for broader financial environments. The approach can be adopted by financial institutions to enhance fraud detection accuracy, reduce false positives, and strengthen customer trust and operational efficiency.

Key Words: Credit card fraud detection; Graph Neural Networks; Auto-encoders; Heterogeneous graphs; Class imbalance.

1 Introduction

Financial transactions, especially credit card usage, have experienced a surge due to the digital revolution. This has resulted in a vast amount of financial data, empowering companies to comprehend customer behavior and utilize data for decision-making. On the other hand, the convenience that comes with this has a downside - there is a noticeable rise in fraudulent activities. Traditional methods of fraud detection often struggle to keep pace with the evolving nature of these schemes. In order to tackle this challenge, the field of machine learning (ML) has surfaced as a potent tool that can effectively identify and prevent fraudulent transactions [2]. By leveraging ML algorithms, it becomes possible to analyze massive amounts of financial data, identify recurring patterns, and pinpoint potential fraud through anomaly detection. They enable financial institutions to automate the fraud detection process, facilitating real-time monitoring of transactions and activities. To detect fraud effectively, many professionals rely on techniques such as decision trees, random forests, and support vector machines [16, 19].

The conventional approaches to detecting fraud often face difficulties in capturing the intrinsic interrelationships that exist within financial data. Transactions typically involve multiple parties, including cardholders, merchants, banks, and various other entities. The representation of financial transactions as a graph enables us to take advantage of the connections among them, thereby enhancing the effectiveness of fraud detection measures. Despite their widespread use, it is important to acknowledge that traditional methods may face difficulties in accurately differentiating between relevant and irrelevant relationships within the graph, thus impacting their ability to effectively detect fraudulent activity.

Graph Neural Networks (GNNs) excel at processing graph data and utilizing attention mechanisms to focus on the most

*Department of Computer Science and Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: sudipta2020@dibru.ac.in.

†**Corresponding author**. Department of Computer Science and Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: tiken.m@dibru.ac.in.

‡Department of Computer Science and Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: rkp@dibru.ac.in.

§Department of Computer Science and Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: abhijit.boruah@dibru.ac.in.

¶Department of Electronics and Communication Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: robertmichael@dibru.ac.in.

‖Department of Electronics and Communication Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: pipizs.kaps@gmail.com.

**Department of Electronics and Communication Engineering, Dibrugarh University Institute of Engineering and Technology, Dibrugarh University, Assam 786004, India. Email: nhsingh@dibru.ac.in.

relevant entities and relationships within the network structure [47]. This makes them well-suited for tasks like fraud detection, where identifying the most critical factors contributing to a transaction's legitimacy is crucial. By applying attention, the GNN can prioritize information from neighboring nodes (e.g., cardholder's spending habits, merchant's location) that are most relevant to understanding the transaction's nature. This refined focus on critical relationships improves the model's ability to distinguish between normal transactions and those exhibiting suspicious patterns, potentially indicative of fraud.

In graph induction learning techniques, two types of graph representations of data are used: homogeneous graph [5] and heterogeneous graph [37]. Financial fraud data, especially involving credit cards, is inherently heterogeneous. It encompasses diverse entities like cardholders, merchants, and transactions, each with distinct attributes and relationships. Homogeneous graphs, which represent entities of the same type, may not fully capture this complexity. In contrast, heterogeneous graphs offer a more comprehensive representation, effectively capturing the multifaceted nature of financial transactions and the intricate relationships between entities within the financial ecosystem.

For instance, a heterogeneous graph might include nodes representing credit card numbers (`cc_num`), merchants information (`merchant_id`), and transaction numbers (`transaction_id`), with edges connecting them based on the specific relationship (e.g., a transaction between a cardholder and a merchant). This allows us to analyze the network structure and identify suspicious patterns that might be missed by simpler models. For instance, in Figure 1, the relationships between different data points are illustrated. These relationships are often overlooked by homogeneous graph learning algorithms and their variants.

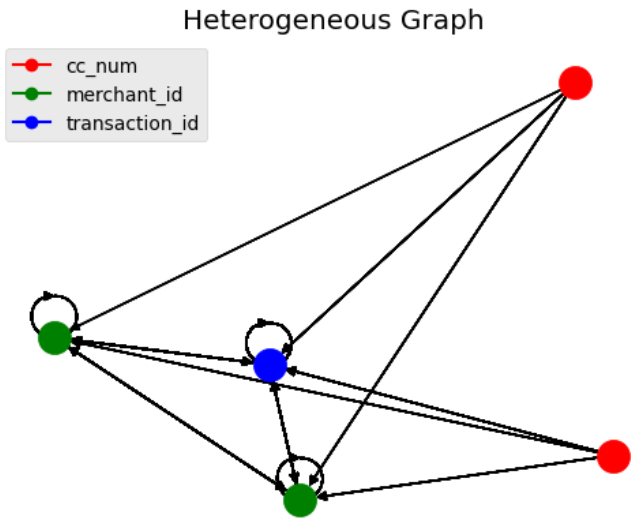


Figure 1: Relationships between different nodes.

The varying characteristics of nodes and edges in heterogeneous graph data make it difficult to apply GNNs directly, thereby necessitating a more sophisticated approach for information aggregation than what is typically used for homogeneous graphs. In addition, the effectiveness of supervised learning is often hindered by class imbalance in fraud data. This imbalance is characterized by a significantly smaller number of fraudulent transactions compared to genuine transactions. As a result, traditional supervised learning models struggle to learn effectively from such imbalanced data [8].

This work suggests a new approach that effectively handles heterogeneous graph data by leveraging advanced GNN techniques for aggregating information from diverse node and edge types. These techniques ensure that the varying attributes and relationships within the graph are adequately captured and utilized in the analysis process.

Furthermore, to tackle the issue of class imbalance, common techniques such as oversampling and undersampling [7] are used. Balancing class distribution can be achieved through oversampling, which generates more instances of the minority class (fraud transactions), or through undersampling, which reduces instances of the majority class (genuine transactions). Nonetheless, these approaches may be complicated and possess their own limitations.

To overcome these challenges, this approach integrates an autoencoder (AE) with a decoder that is trained on genuine transactions. By learning a latent representation, the AE can accurately reconstruct these transactions. The ability to detect fraudulent activities in complex heterogeneous graph data is enhanced by flagging deviations from the learned distribution during reconstruction, thereby addressing class imbalance.

Considering all scenarios discussed, this work aims to answer the following research questions (RQs):

- **RQ1: Effectiveness of GNNs with Attention for Fraud Detection:**How effectively can GNNs utilizing an attention mechanism detect and prevent credit card fraud when applied to a heterogeneous graph representation that captures the complex interrelationships within the financial ecosystem?
- **RQ2: Comparison of Autoencoder with Attention vs. Traditional Methods:** How does the proposed autoencoder-based fraud detection approach, which leverages GNNs with attention and is trained on a non-fraudulent transaction graph dataset, compare to traditional methods in terms of accuracy, efficiency, and scalability, especially considering significant class imbalance?

The methodology consists of several steps, one of which is the processing of a tabular dataset of financial transactions. This dataset is then transformed into a heterogeneous graph. As a result, the graph is subjected to analysis using autoencoders (AE) and graph neural networks (GNNs), which enables the identification of anomalies that can be linked to fraudulent activity. By focusing on the class imbalance problem, the proposed approach effectively tackles the challenge of fraud detection tasks. The results of this work have

significant implications for businesses and financial institutions, empowering them to gain valuable insights into customer behavior and enhance their ability to identify and prevent fraudulent transactions. Ultimately, this work contributes to the advancement of fraud detection systems and the overall security of financial transactions in the digital era.

This paper provides a comprehensive discussion of the relevant literature in Section 2. The problem statement is outlined in Section 3, aiming to address a specific problem. The methodology employed in this research is elucidated in Section 4. The results obtained from this methodology are analyzed and presented in Section 5. Finally, Section 5.5 concludes the paper by summarizing the key findings and implications.

2 Literature Review

In this section, we introduce a range of notable works that cover various topics such as probabilistic graphical models, machine learning algorithms (including deep learning models), and advanced graph neural networks and their various variants.

Papers such as [38] and [34] aim to address the problem of fraud detection in credit card transactions by modeling these transactions using a Hidden Markov Model (HMM), a probabilistic graphical model. The primary difference between them lies in their approach: in the first paper, a card-centric HMM is employed to detect abnormalities in transactions, while the latter paper opts for a merchant-centric HMM model. Both methods have the capability to identify fraud in real-time for merchants, operating in conjunction with modern transaction processing systems that handle card transactions.

Additionally, [27] models credit card transaction sequences using the HMM approach, considering three distinct perspectives:

(i) Determining whether fraud is present or absent in the sequence.

(ii) Crafting sequences by fixing either the cardholder or the payment terminal.

(iii) Constructing sequences based on the spent amounts or the elapsed time between consecutive transactions. The combination of these three binary perspectives results in eight distinct sets of sequences derived from the training dataset of transactions. Each of these sequences is then represented using a Hidden Markov Model (HMM). Subsequently, each HMM assigns a likelihood to a transaction based on its sequence of preceding transactions. These likelihood values serve as additional features for the Random Forest classifier to detect fraud. In brief, this model provides a concept of sequential information flow during credit card transactions as part of a feature for a machine learning model.

The paper [18] explores the issue of credit card fraud detection and conducts a comparative analysis of three machine learning algorithms: logistic regression, Naïve Bayes, and K-nearest neighbor. To address the class imbalance, the authors utilize different proportions of the dataset and employ a random undersampling technique. They evaluate the algorithms based on various metrics. According to the results, the logistic regression-based model outperforms the prediction models derived from Naïve Bayes and K-nearest neighbor. The paper also suggests that applying undersampling techniques to the data before model development can lead to improved results. In addition, several machine learning algorithms, such as support vector machine (SVM) [35], random forest (RF) [35, 22], AdaBoost, and Majority Voting [31], as well as artificial neural network (ANN) [33, 1], are being explored as models for controlling fraudulent transactions in credit cards.

To enhance the performance of the above-mentioned models, [17] defines a model in an ML-driven credit card fraud detection system that uses the genetic algorithm (GA) for feature selection. After identifying optimal features, this detection system utilizes a range of ML classifiers, including Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Artificial Neural Network (ANN), and Naive Bayes (NB).

While the aforementioned models perform well, a significant class imbalance exists in the credit card fraud dataset, with non-fraudulent transactions vastly outnumbering fraudulent ones. As a result, these models tend to prioritize high precision by predominantly predicting the majority class. To address this issue, several machine learning models (referenced as [28]) employ one or a combination of oversampling and undersampling techniques (as mentioned in [6]).

The study cited as [3] conducts a comparative investigation of various approaches to address class imbalance. The findings indicate that a combination of oversampling and undersampling methods performs well when applied to ensemble classification models, including AdaBoost, XGBoost, and Random Forest. Deep learning algorithms such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), combined with a multilayer perceptron, are employed in the studies referenced as [28] and [13]. In [13], the authors use the Hybrid Synthetic Minority Oversampling Technique and Edited Nearest Neighbor (SMOTE-ENN) to balance the distribution of positive (fraud) and negative (non-fraud) instances in the dataset. However, the effectiveness of the SMOTE-ENN technique is crucial, as poor performance in resampling can significantly degrade the model's overall performance.

While oversampling and undersampling techniques can address class imbalance, they come with drawbacks like increased computational cost, potential for overfitting, and information loss (as discussed in [7, 44]). Additionally, they can be sensitive to noise [46] and have limited effectiveness for highly imbalanced datasets [12]. Therefore, [12] propose an approach for Chronic Kidney Disease (CKD) prediction using imbalanced data. Their method leverages information gain-based feature selection and a cost-sensitive AdaBoost classifier. However, this approach focuses on spatial data and might not be suitable for graph data due to potential loss of structural information and inadequate feature representation during feature selection. So, such models will often struggle to capture the full picture of fraudulent activity. As noted in [9], many methods focus solely on spatial data points representing

financial transactions, neglecting the valuable insights from temporal relationships. This limitation hinders the ability of these models to identify evolving fraud patterns. Furthermore, many existing models rely solely on labeled data for training, restricting their ability to leverage the vast amount of unlabeled data available in real-world credit card transactions [36].

To address these issues, an increasing number of researchers are exploring graph-based techniques for fraud detection, as discussed in [9] and [23]. In this approach, datasets are transformed into graphs, providing a better understanding of the relationships among financial transactions. Graph Neural Network (GNN) algorithms, as detailed in [45], are applied to these graph datasets, allowing for efficient data aggregation from neighboring nodes and the extraction of node representations within the graph datasets. Among the popular GNN variants, GraphSAGE [14] and GAT [43] stand out, utilizing sampling methods and attention mechanisms to gather neighbor information. These techniques have shown promising results in the field of fraud detection. Furthermore, the paper [24] introduces an algorithm designed to tackle the class imbalance problem in graph-based fraud detection. It employs an algorithm known as Pick and Choose Graph Neural Network (PC-GNN) to perform imbalanced supervised learning on graphs. The PC-GNN algorithm selects neighbor candidates for each node within the sub-graph using a neighborhood sampler. Ultimately, it aggregates information from the chosen neighbors and different relations to derive the final representation of a target node. The paper reports that PC-GNN surpasses state-of-the-art baselines in both benchmark and real-world graph-based fraud detection tasks.

However, inconsistency issues arise in the aggregation process of GNN models when applied to fraud detection tasks [25]. The aggregation mechanism relies on the assumption that neighbors share similar features and labels. When this assumption breaks down, the aggregation of neighborhood information becomes ineffective in learning node embeddings.

To address these challenges, researchers in [25] and [39] have employed a multi-relational graph, known as a heterogeneous graph, for the classification of financial fraud. In [25], context inconsistency, feature inconsistency, and relation inconsistency in GNN are introduced. To tackle these inconsistencies, the authors propose a new GNN framework called GraphConsis. GraphConsis addresses these issues by combining context embeddings with node features to handle context inconsistency, designing a consistency score to filter inconsistent neighbors and generate corresponding sampling probabilities to address feature inconsistency, and learning relation attention weights associated with the sampled nodes to tackle relation inconsistency.

In [39], the authors propose semi-supervised methods that operate with heterogeneous graph datasets to address class imbalance issues in online credit loans. This paper utilizes a Graph-Oriented Snorkel approach to incorporate external expert knowledge, ultimately improving the performance of the learning algorithm when dealing with imbalanced datasets.

Another noteworthy work, [26], introduces a heterogeneous graph-based approach for detecting malicious accounts in financial transactions. The authors present an algorithm called GEM, which adapts to learn discriminative embeddings for various node types. GEM employs an aggregator to capture node patterns within each type and utilizes an attention mechanism to enhance algorithm efficiency.

In [32], the authors endeavor to design heterogeneous graph embeddings. Their approach incorporates heterogeneous mutual attention and heterogeneous message passing, incorporating key, value, and query vector operations (self-attention mechanism). This work features both a detector and an explainer, capable of predicting the validity of incoming transactions and providing insightful, understandable explanations generated from graphs to aid in subsequent business unit procedures.

The framework employed in [41] utilizes an algorithm for graph representation learning to create concise numerical vectors that capture the underlying network structure. The authors in this work assess the predictive capabilities of inductive graph representation learning with GraphSage and Fast Inductive Graph Representation Learning algorithms on credit card datasets characterized by significant data imbalance.

3 Problem Statement

A heterogeneous graph is a specialized graph data structure that comprises multiple types of nodes and edges, wherein each node or edge is uniquely associated with a distinct type. In essence, it represents a graph in which diverse node and edge types are interconnected. To provide a formal definition, the characteristics of a heterogeneous graph are delineated as follows:

Definition 3.1. *A heterogeneous graph, also known as a heterogeneous information network or heterogeneous network, is mathematically defined as* $G = (V, E, T, R, X)$*, where:*

- V *represents the set of nodes in the graph, and each node* $v^t \in V$ *is associated with a specific type* $t \in T$*, where* T *represents the set of node types.*
- E *represents the set of edges in the graph, and each edge* $e^r \in E$ *connects two nodes* (v^{t_1}, v^{t_2})*, where* t_1 *and* t_2 *are node types, and* $r \in R$*, where* R *represents the set of edge types or relationships.*
- $X = \{X_v, X_e\}$ *represents attributes of nodes and edges, respectively, where* X_v *represents the set of node attributes, and each node* $v^t \in V$ *can have a vector of attributes* x_{v^t}*, and* X_e *represents the set of edge attributes, where each edge* $e^r \in E$ *can have a vector of attributes* x_{e^r}*.*

By adhering to its definition, the financial fraud dataset can be depicted as a heterogeneous graph. These datasets encompass various entities, including customer or credit card numbers, merchants' names, and transaction numbers. These entities are represented as nodes in the graph, denoted by V. Specifically,

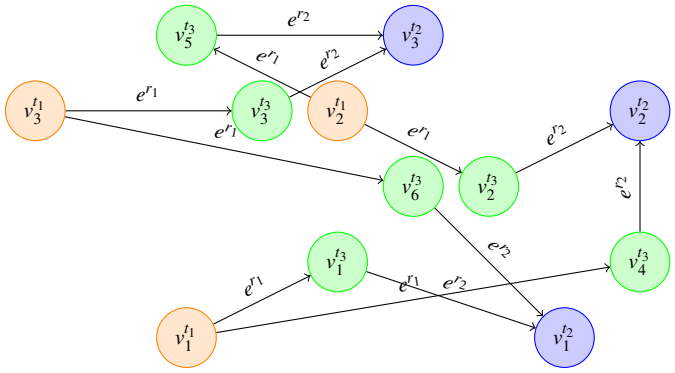


Figure 2: A heterogeneous graph illustrating different types of nodes and edges.

the nodes v^{t_1}, representing 'customer', v^{t_2}, representing 'merchants', and v^{t_3}, representing 'transaction', encapsulate the essence of this heterogeneous graph. Consequently, these nodes (v^{t_1}, v^{t_2}, and v^{t_3}) are distinguished by their respective types.

The heterogeneous graph depicted in Figure 2 illustrates a network where nodes are categorized into three distinct types: 'customers' (in orange), 'merchants' (in blue), and 'transactions' (in green). Each node type is uniquely identified by an index (v_i^t), where i indexes different instances of customers, merchants, and transactions within the same type t (e.g., $v_1^{t_1}$ for the first customer, $v_2^{t_1}$ for the second customer). The graph captures complex interactions: customers initiate transactions (e^{r_1}) that involve merchants (e^{r_2}). Notably, customers can engage in multiple transactions across different merchants, as represented by multiple transaction nodes ($v_1^{t_3}, v_2^{t_3}, \ldots, v_6^{t_3}$). This structured representation facilitates the analysis of interconnected relationships within heterogeneous networks, essential for understanding dynamics financial transactions.

Problem 1. *For the given graph $G = (V, E, T, R, X)$, the task is to determine whether it can be classified as fraudulent, considering that the transaction associated with the graph represents a fraudulent class.*

4 Methodology

The primary objective of this paper is to develop an encoder capable of learning graph embeddings for a given heterogeneous graph $G = (V, E, T, R, X)$, where V represents nodes, E denotes edges, T indicates node types, R specifies edge types, and X contains node attributes. This encoder is designed to effectively capture the complex information present in a heterogeneous graph, including both its structure and attributes. Subsequently, a decoder function f_{dec} reconstructs the graph. Figure 3 illustrates the model architecture used in this paper.

The model comprises l encoder units. The first encoder unit takes $(D(d^t), \phi, d^t)$ as input, where $D(d^t)$ represents the source nodes of $d^t \in V$, and ϕ denotes edges e^r from each source node

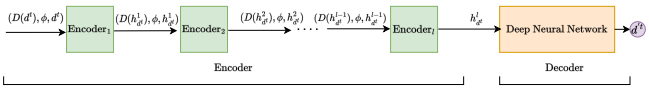


Figure 3: Encoder Units and Decoder Unit of the Model

to d^t. Each encoder unit processes these inputs to produce intermediate representations. The final output of Encoder$_l$ is fed into a decoder unit, implemented as a deep neural network, which utilizes the encoded information to generate d'^t.

The model calculates the reconstruction error by comparing the reconstructed graph to the original graph. This error measures the dissimilarity between the original input and the reconstructed output. A threshold for the reconstruction error is established to identify data points that deviate significantly from normal patterns. Any data point with a reconstruction error exceeding the threshold is classified as an anomaly, indicating a deviation from expected behavior, such as fraudulent activity in a financial transaction network.

While the methods discussed above claim to perform well with unbalanced heterogeneous graph datasets, techniques such as autoencoders and decoders, as presented in [40, 10, 11], offer alternative solutions. For instance, [11] successfully addressed imbalanced medical datasets using a modified Sparse Autoencoder (SAE) and Softmax regression for enhanced diagnosis. However, SAEs are less suitable for data with inherent relationships between elements, which is particularly relevant for fraud detection in transactional networks, where connections between nodes are crucial for identifying suspicious activity. Similarly, [10] employed a Stacked SAE (SSAE) for credit card default prediction on imbalanced data. Nevertheless, SSAEs, like SAEs, lack the ability to explicitly prioritize information from relevant neighboring nodes. This limitation necessitates a different approach for this work, which leverages a transactional network to represent data and identify fraudulent activities.

4.1 Dataset Preprocessing

The heterogeneous graph G is constructed from a financial transaction dataset, where nodes represent entities (e.g., customers, merchants, transactions) and edges represent interactions (e.g., payments, refunds). Node types T include "Customer," "Merchant," and "Transaction," while edge types R include "Pays," "Receives," and "Refunds." Node attributes X include features such as transaction amount, timestamp, customer demographics, and merchant category.

The dataset is preprocessed as follows:

- **Graph Construction**: Transactions are modeled as nodes connected to customer and merchant nodes via typed edges. For each transaction node d^t, the set of source nodes $D(d^t)$ includes the associated customer and merchant.
- **Feature Normalization**: Continuous attributes (e.g., transaction amount) are standardized to have zero mean

and unit variance. Categorical attributes (e.g., merchant category) are one-hot encoded.
- **Handling Missing Data**: Missing attributes are imputed using the mean (for continuous features) or mode (for categorical features) of the respective node type.
- **Graph Partitioning**: For large datasets, the graph is partitioned into subgraphs using community detection to facilitate scalable processing.

The preprocessed graph ensures that node and edge types are preserved, and attributes are in a suitable format for the encoder.

4.2 Encoder for Heterogeneous Graph

Based on the study by [15], a heterogeneous graph encoder for the autoencoder has been designed (Figure 4). For each destination node $d^t \in V$ and its source nodes $D(d^t) \in V$, the encoding process f^{enc} is applied as follows:

$$h_{d^t}^l = f_{\text{reparam}}\left(\text{Linear}_{d^t}\left(f^{\text{enc}}_{\forall v \in D(d^t)}(h_{v^t}^{l-1}, e^r, h_{d^t}^{l-1})\right) \oplus h_{d^t}^0, \text{mean}(h_{d^t}^{l-1}), \log(h_{d^t}^{l-1})\right) \tag{1}$$

Here, $l = 1, 2, \ldots, E_L$ represents the encoder layer, with a maximum of E_L layers, and initial values are set as $(h_{v^t}^0, e^r, h_{d^t}^0) = (v^t, e^r, d^t)$. The function $\text{Linear}_{d^t} : \mathbb{R}^{\frac{\dim}{k}} \to \mathbb{R}^{\dim}$ denotes a linear projection.

The encoding process f^{enc} is defined as:

$$f^{\text{enc}}_{\forall v^t \in D(d^t)} = \bigoplus_{\forall v^t \in D(d^t)} \left(f^{\text{Attent}}(v^t, e^r, d^t) \cdot f^{\text{Mssg}}(v^t, e^r, d^t)\right) \tag{2}$$

The attention mechanism is:

$$f^{\text{Attent}}(v^t, e^r, d^t) = \text{Softmax}\left(\|_{\forall k \in [1,H]} \text{Att}^k(v^t, e^r, d^t)\right) \tag{3}$$

Inspired by [42], the attention for each edge e^r is calculated using k-heads:

$$\text{Att}^k(v^t, e^r, d^t) = \text{LinearS}_t^k\left(h_{v^t}^{l-1}\right) \cdot W_r^{\text{Att}} \cdot \left(\text{LinearD}_t^k\left(h_{d^t}^{l-1}\right)\right)^T \tag{4}$$

Here, LinearS_t^k and LinearD_t^k map from $\mathbb{R}^{\dim *}$ to $\mathbb{R}^{\frac{\dim}{k}}$, and $W_r^{\text{Att}} \in \mathbb{R}^{\frac{\dim}{k} \times \frac{\dim}{k}}$ is a learnable edge matrix for edge type r.

The message passing function is:

$$f^{\text{Mssg}}(v^t, e^r, d^t) = \|_{\forall k \in [1,H]} \text{LinearM}_t^k\left(h_{v^t}^{l-1}\right) W_r^{\text{Mssg}} \tag{5}$$

The reparameterization function f_{reparam}, inspired by [21], models the latent variable probabilistically:

$$\begin{aligned} h_{d^t}^l &= f_{\text{reparam}}\left(\text{mean}(h_{d^t}^l), \log(h_{d^t}^l)\right) \\ &= \text{mean}(h_{d^t}^l) + \varepsilon \cdot \exp\left(\frac{1}{2} \cdot \log(h_{d^t}^l)\right) \end{aligned}$$

where $\varepsilon \sim \mathcal{N}(0,1)$.

4.3 Decoder for Heterogeneous Graph

The decoder reconstructs the graph's structure and attributes, accounting for its heterogeneous nature. For each node d^t, a node decoder f_{dec} reconstructs attributes d'^t:

$$d'^t = f_{\text{dec}}(h_{d^t}^l) \tag{6}$$

The decoder is a multi-layer perceptron (MLP) with type-specific parameters to handle different node types. The loss function is defined as:

$$L = \sum_{\forall N} \sum_{\forall t} \text{LOSS}(d^t, d'^t) \tag{7}$$

The loss function is specified as the mean squared error (MSE) for continuous attributes:

$$\text{LOSS}(d^t, d'^t) = \|d^t - d'^t\|_2^2 \tag{8}$$

For categorical attributes, cross-entropy loss is used. The total loss combines losses across all node types, weighted by their prevalence in the dataset to address class imbalance.

4.4 Algorithm

Algorithm 1 Fraud Detection on a Heterogeneous Graph

Require: Heterogeneous Graph G
Ensure: 'Fraud' or 'Not Fraud'

for $d^t \in G$ **do**
  $(h_{v^t}^0, e^r, h_{d^t}^0) \leftarrow (v^t, e^r, d^t)$ ▷ Initialization
  for $l \leftarrow 1$ **to** E_L **do** ▷ Message Passing Layers
    for $v^t \in D(d^t)$ **do** ▷ Neighborhood of d^t
      $h_{d^t}^l = \text{Linear}_{d^t}\left(f^{\text{enc}}(h_{v^t}^{l-1}, e^r, h_{d^t}^{l-1})\right) \oplus h_{d^t}^0$
    end for
    $h_{d^t}^l = f_{\text{reparam}}\left(\text{mean}(h_{d^t}^l), \log(h_{d^t}^l)\right)$ ▷ Reparameterization
  end for
  $d'^t = f_{\text{dec}}(h_{d^t}^l)$ ▷ Output Layer
end for
$L = \text{LOSS}(d^t, d'^t)$ ▷ Loss Calculation
if $L <$ Threshold() **then**
  return 'Non-Fraud'
else
  return 'Fraud'
end if

The algorithm depicted in Algorithm 1, outlines the method for detecting fraud in a heterogeneous graph structure. Here's a detailed breakdown of each step:

1. Input (Heterogeneous Graph G): This represents the financial transaction network, containing nodes (customers, merchants, transactions) and edges (interactions) with their respective types.

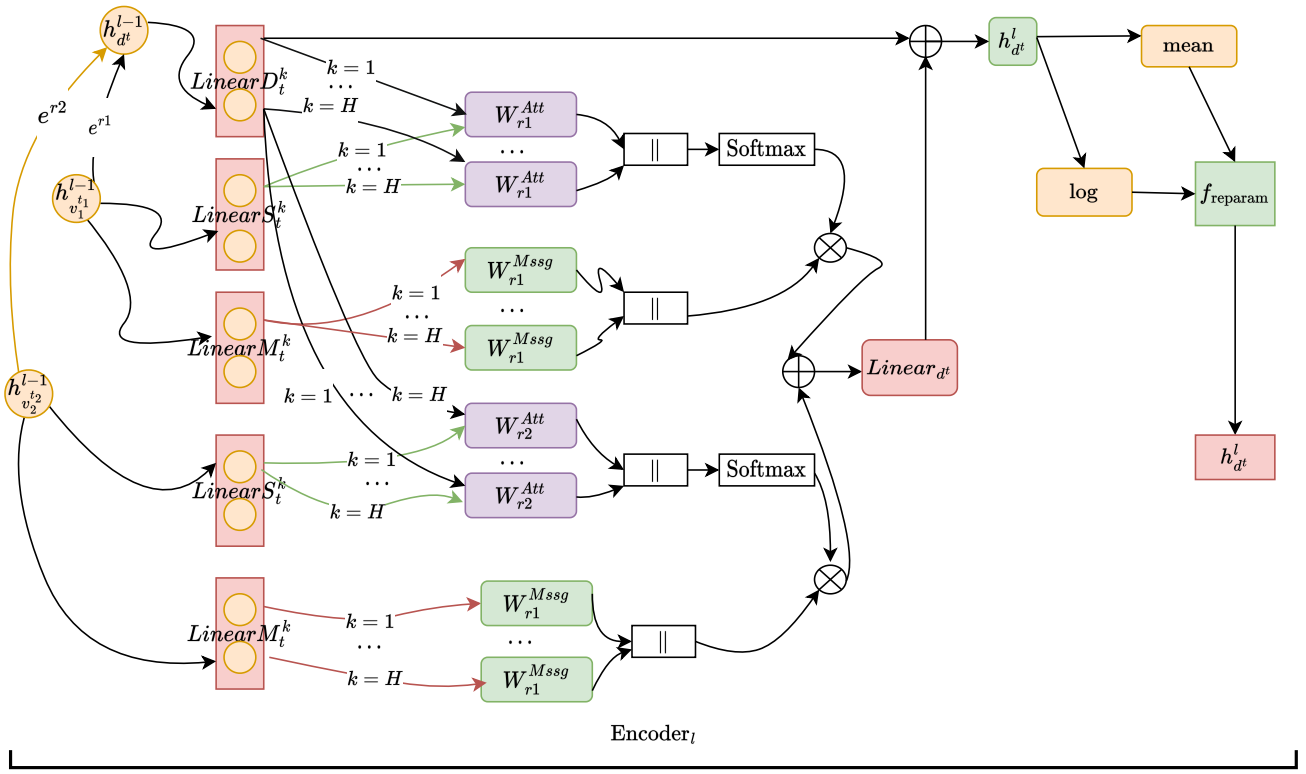


Figure 4: Encoder Unit for Heterogeneous Graphs. e^{r1} and e^{r2} denote edges from source nodes $v_1^{t_1}$ and $v_2^{t_2}$ to destination node d^t. At $l = 0$, it represents the initial encoder layer, producing $h_{d^t}^1$, and so on. k ranges from 1 to H, $\|$ signifies concatenation, $\oplus$ denotes addition, and $\otimes$ indicates dot product.

2. Output: **"Fraud" or "Not Fraud"**: The algorithm classifies the transaction associated with the input graph as either fraudulent or legitimate.
3. Algorithm Steps:
   - For each node d^t in the graph G, node d^t is initialized with $(h_{v^t}^0, e^r, h_{d^t}^0)$. It includes the features of the node itself $h_{d^t}^0$, the connecting edge type e^r, and the initial representation of the source node $h_{v^t}^0$.
   - Message Passing Layers (L Layers):
     - This loop iterates through a predefined number of layers (E_L) in the GNN architecture.
     - Within each layer l:
       * For each node v^t in the neighborhood of the current node d^t:
         · A message function f^{enc} (Equation 2) aggregates information from the source node's hidden representation $h_{v^t}^{l-1}$, the edge type e^r, and the previous hidden representation of the destination node $h_{d^t}^{l-1}$. The message undergoes a linear transformation with Linear_{d^t} as per equations (3-5).
         · By utilizing the attention mechanism, the messages undergo transformation and are subsequently combined with the initial hidden representation of the destination node $h_{d^t}^0$ through element-wise addition ($\oplus$).
       * The message passing happens iteratively for all neighbors of d^t.
       * The updated hidden representation $h_{d^t}^l$ is subjected to f_{reparam} (Equation 1) after message aggregation. Mean and logarithm are utilized in hidden representation to ensure greater stability during training.
4. Output Layer: The final hidden representation $h_{d^t}^l$ is passed through the decoder function f_{dec} (Equation 6) to produce the prediction vector d'^t.
5. Loss Calculation: The difference between the predicted output d'^t and the original node feature d^t is evaluated using a loss function LOSS. The LOSS function can use a metric such as mean squared error or any other appropriate loss function.
6. Fraud Classification: The threshold for anomaly detection is determined using a validation set of non-fraudulent transactions. The reconstruction errors are computed, and the threshold is set as:

$$\text{Threshold} = \mu + 2\sigma \tag{9}$$

where μ and σ are the mean and standard deviation of the reconstruction errors, respectively. This ensures that approximately 95% of non-fraudulent transactions are classified as "Non-Fraud." The threshold is tuned on the validation set to balance precision and recall.

Algorithm 1 explains the entire framework of the model, which is designed to identify if a specific data point is linked to fraudulent behavior, resulting in one of two possible outcomes: 'Fraud' or 'Not Fraud.' The algorithm calculates a loss value to measure the difference between the original transaction node and its decoded version. The computation of this loss relies on a loss function that has been predetermined. The next step in the process is for the algorithm to compare the resulting loss with a predetermined threshold, once all the calculations have been completed. In the case where the loss falls below the designated threshold, the data point is classified as 'Not Fraud'. The overall time complexity of the algorithm can be approximated as $\mathcal{O}(nE)$ by summing up these components, with n representing the number of nodes in the graph.

5 Experiment

This paper assesses the effectiveness of the proposed model through a series of experiments on credit card fraud datasets and a comparison with other existing machine learning and deep learning models.

5.1 Performance Metrics

In order to evaluate the performance of various models, this article employed evaluation metrics that include the precision rate (PR), the recall rate (RR), the ROC curve, and the F1 score. These metrics are defined as follows:

$$\mathrm{PR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$$

$$\mathrm{RR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$

In this context, true positive (TP) and false positive (FP) indicate the number of correctly and incorrectly predicted instances of fraud, respectively. Conversely, true negative (TN) and false negative (FN) correspond to the count of transactions accurately and inaccurately predicted as non-fraudulent.

Meanwhile, the ROC curve illustrates the classifier's ability to differentiate between fraud and non-fraud categories. This curve is created by plotting the true positive rate against the false positive rate at different threshold levels. The AUC, which ranges from 0 to 1, encapsulates the information from the ROC curve. A value of 0 signifies that all classifier predictions are erroneous, while a value of 1 indicates a perfect classifier.

The F1 score represents the harmonic mean of precision and recall. Precision is the ratio of true positive predictions to the total predicted positives and recall is the ratio of true positive predictions to the total actual positives. It provides a single value that harmonizes precision and recall, facilitating a balanced evaluation of classifier performance.

$$\mathrm{F1} = 2 * \frac{(\mathrm{PR} * \mathrm{RR})}{(\mathrm{PR}+\mathrm{RR})}$$

Given that the dataset is imbalanced, the F1 score is particularly valuable because it considers both precision and recall. This score provides a straightforward way to assess a classifier's overall effectiveness in accurately identifying positive instances while minimizing false positives and false negatives.

Another parameter used to gain insight into the model's performance is the Precision-Recall curve (AUC-PR) [30]. This metric offers valuable insights, particularly in situations where class distribution is imbalanced [29].

5.2 Datasets

The dataset ([20]) used in this article simulates credit card transactions and includes genuine and fraudulent activities that occurred between January 1, 2019, and December 31, 2020. The data encompass transactions carried out by 1000 customers using credit cards issued by a variety of banks, engaging in transactions with a pool of 800 different merchants.

Types of Dataset	Normal Data	Abnormal Data
Training Dataset	1842743	9651
Testing Dataset	553574	2145

Table 1: Distribution of Fraudulent Transactions on Training and Testing Dataset

Table 1 illustrates the distribution of fraudulent and non-fraudulent transactions in a dataset. It shows the number of occurrences of each type of transaction, with "1" representing fraudulent (Abnormal) transactions and "0" representing non-fraudulent (Normal) ones. This analysis gives an indication of the skewed and unbalanced ratio of fraudulent to non-fraudulent transactions.

5.3 Analysis of Algorithms

In the article ([4]), some of the best machine learning algorithms that handle fraud datasets are listed. Here is the list used in the article:

- Linear Regression
- Logistic Regression
- Decision Tree
- SVM (Support Vector Machine)
- ANN (Artificial Neural Network)
- Naïve Bayes
- DNN (Deep Neural Network)
- K-Means
- Random Forest

- Dimensionality Reduction Algorithms
- Gradient Boosting (XGB) Algorithms

These algorithms cover a wide range of machine learning (ML) aspects, including association analysis, clustering, classification, statistical learning, and link mining. They hold a crucial place among the essential topics explored in research and development within the field of machine learning. However, when evaluating these algorithms with datasets, their performance often falls short of expectations due to the inherent imbalance present in the data.

Performance of Machine learning (ML) Algorithms					
ML Algorithm	Testing Accuracy	Testing F1 Score	Test Precision	Test Recall	AUC
Decision Tree	0.99	0.29	0.22	0.43	0.82
XGB Classifier	0.99	0.33	0.27	0.43	**0.96**
ANN	0.99	0.33	0.23	0.32	0.92
Deep NN	0.99	0.33	**0.40**	0.26	0.81
AE	–	**0.67**	0.50	0.99	0.52
VAE	–	**0.67**	0.50	0.99	0.54
Sparse AE	–	**0.67**	0.50	0.99	0.54

Table 2: Performance Measurement of Few Selected Machine Learning Algorithms

Table 2 provides the performance metrics for a few machine learning algorithms. The table shows that the F1 score of all machine learning algorithms is too low, suggesting that these algorithms could not handle unbalanced datasets properly. Since the F1 score is low and the AUC curve is high for all ML algorithms, it indicates that these algorithms are adept at distinguishing between abnormal and normal data, as evidenced by the high AUC value. However, the F1 score is low due to the models facing challenges in achieving both high precision and high recall, attributed to the imbalanced nature of the data.

These scenarios arise when the negative class dominates the dataset, creating a highly imbalanced situation. In such cases, models tend to classify instances as the majority class, resulting in high true negatives and low false positives but at the cost of missing true positives and having low recall. To address the challenges posed by unbalanced data, various algorithms are explored. One of the algorithms under consideration is the autoencoder algorithm.

The exploration involves simple autoencoders (AE) using deep neural networks and their variations, such as variational autoencoders (VAE) and sparse autoencoders (Sparse AE). Table 2 also shows the performance of the autoencoders. Regardless of the specific type, the model's performance is evaluated using key metrics. The F1 score, which harmonizes precision and recall, yielded a value of 0.67. This suggests that the models have achieved a reasonable balance between making accurate positive predictions and effectively capturing actual positive instances. Overall, the performance is decent, showing a well-rounded approach.

However, the narrative changes when examining the Receiver Operating Characteristics (ROC) curve and its corresponding Area Under the Curve (AUC). With an AUC of 0.57, it implies that the models struggle to distinguish between fraud and normal classes. Their ability to classify effectively in this context appears limited and performs only slightly better than random guessing.

In a deeper dive, the precision achieved by the autoencoder models in the test set is 0.50. This means that roughly half of the abnormal predictions it generates are accurate, while the other half are incorrect. On the other hand, the recall rate is impressive at 0.99. This means that the models excel at identifying almost all the actual abnormal instances present in the dataset.

In summary, while autoencoder models demonstrate balanced performance in terms of the F1 score, with commendable recall and reasonable precision, the AUC score and precision rates indicate room for improvement. Enhancing the discriminatory capacity of models and refining their positive prediction accuracy could be areas of focus to further elevate their performance in classification tasks.

5.4 Analysis of the Proposed Model

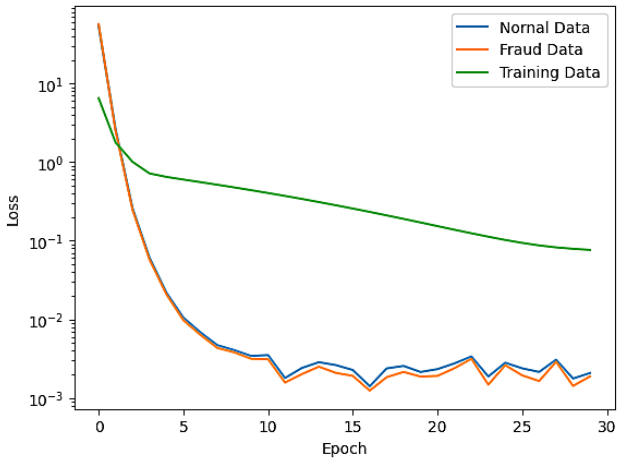
Parameter Name	**Value**
Size of Hidden Layers	64
Number of heads (H)	16
Number of Layers for the Encoder (l)	124
Number of Layers for the Decoder	64
Dropout Rate	0.4
Regularization Rate	0.01

Table 3: Values for different parameters used in the model.

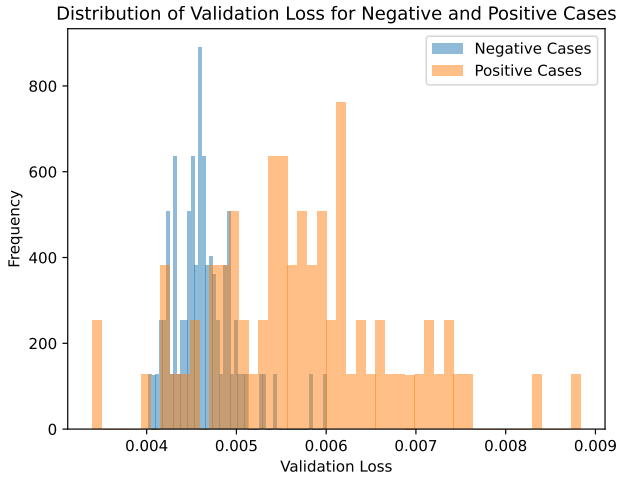
After tuning the parameters for different hyperparameters, the performance of the model is represented as shown in Figure 5. Finally, the proposed model uses the parameters defined in Table 4 to evaluate the model's performance.

In Figure 5a, the training loss is compared with the validation loss for positive (fraud) and negative datasets. This plot provides insight into how effectively the model handles overfitting and underfitting of the data. The model, using the parameters from Table 4, demonstrates immunity to both overfitting and underfitting, effectively managing these issues. Figure 5b illustrates the loss distribution (histogram) generated by the model from the dataset. This distribution shows the loss values for both positive and negative data in the dataset. The figure reveals that the loss for negative instances is concentrated between 0.004 and 0.005, while the loss for positive instances is distributed beyond 0.006.

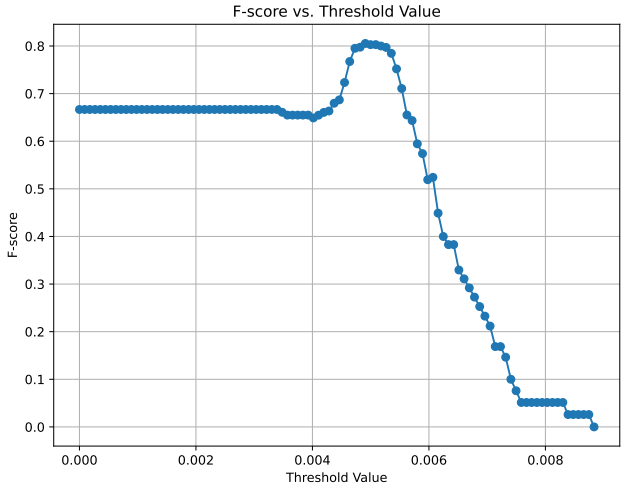
Figure 5c defines the model's F1 score versus the classification threshold value. From the figure, it can be seen



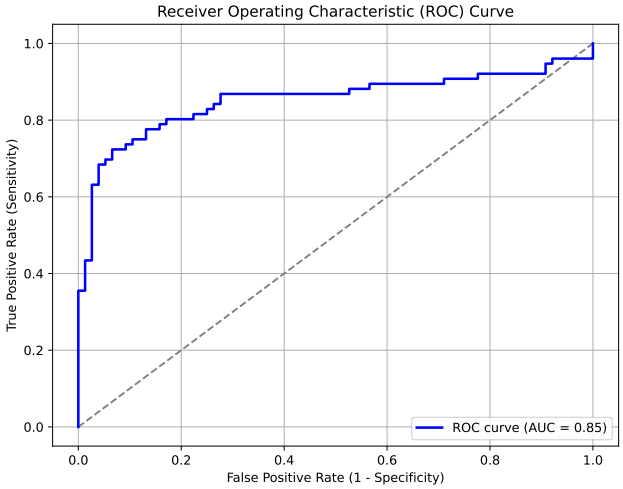
(a) Training loss and evaluation loss.



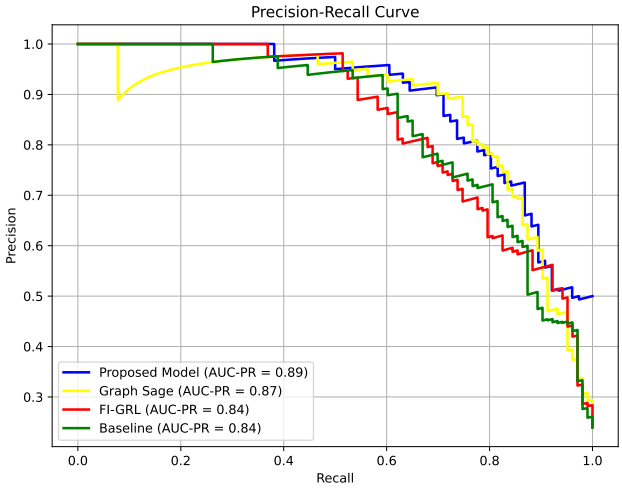
(b) Distribution of validation loss.



(c) F Score vs Threshold graph.



(d) ROC curve



(e) Precision-Recall curve of the model with an AUC-PR of 0.89.

Figure 5: Performance evaluations of the proposed model.

that the F1 score reaches its highest value of 0.81 at a loss value of 0.005. Additionally, the ROC curve was plotted based on the threshold, resulting in the ROC curve shown in Figure 5d, and an AUC of 0.85 was obtained for the model.

The Precision-Recall (PR) curve (Figure 5e) compares the performance of four algorithms: the Proposed Model, Graph Sage [41], FI-GRL [41], and Baseline [41]. The Proposed Model exhibits the highest performance with an AUC-PR of 0.89, indicating the best balance between precision and recall. Graph Sage follows closely with an AUC-PR of 0.87, showing strong but slightly inferior performance compared to the Proposed Model. Both FI-GRL and the Baseline models have an AUC-PR of 0.84, indicating moderate performance and similar effectiveness in maintaining precision and recall. Overall, the Proposed Model stands out as the most effective, followed by Graph Sage, with FI-GRL and Baseline performing similarly but less effectively.

Again, Table 4 summarizes the performance of various graph learning algorithms on metrics including AUC-PR, F1-Score, and ROC-AUC. The proposed model achieves the highest AUC-PR (0.89) and F1-Score (0.81) but has a lower ROC-AUC (0.85) compared to Graph Sage and XBoost, which achieve a ROC-AUC of 0.93.

Performance of Graph Learning Algotihms			
Graph Algorithm	**AUC-PR**	**F1 Score**	**ROC-AUC**
Proposed Model	**0.89**	**0.81**	0.85
Graph Sage and XBoost ([41])	0.86	0.80	**0.93**
FI-GRL([41])	0.84	0.70	0.92
Baseline([41])	0.84	0.74	0.91

Table 4: Performance Measurement of Graph Learning Algorithms. AUC-PR provides sufficient information to assess performance due to the imbalanced nature of the dataset used.

Finally, Figure 6 showcases the following algorithms: Proposed Model, Graph Sage and XBoost, FI-GRL, Baseline, Decision Tree, XGB Classifier, ANN (Artificial Neural Network), and Deep NN (Deep Neural Network). This radar chart highlights the exceptional performance of the Proposed Model, with high scores in F1 score, AUC-PR, and ROC-AUC, demonstrating a strong and balanced performance.

While Graph Sage and XBoost show great performance in class discrimination with high ROC-AUC, their AUC-PR is slightly lower, suggesting a trade-off when dealing with imbalanced datasets. Both FI-GRL and Baseline demonstrate strong classification performance with high ROC-AUC, but they may prioritize precision or recall at the expense of balance, resulting in a lower F1 score.

The Decision Tree and XGB Classifier face challenges in their competition, as the Decision Tree exhibits overall weakness, and the XGB Classifier lacks balance despite its

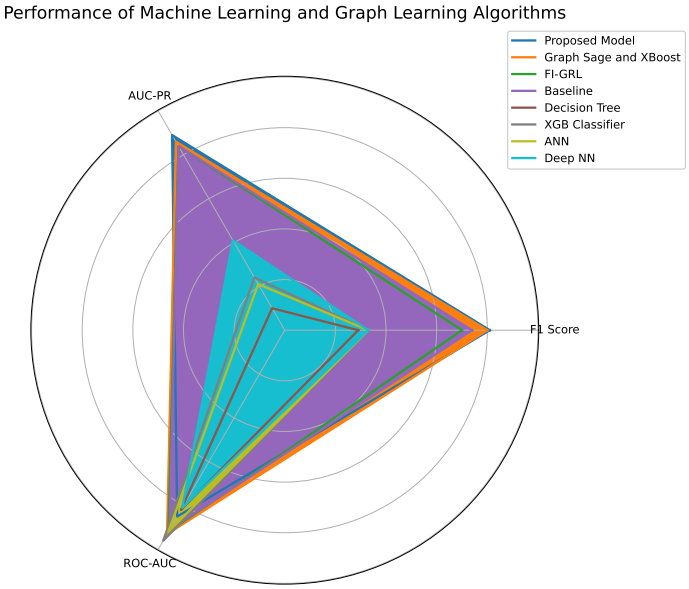


Figure 6: Performance Radar Chart. It compares several machine learning (ML) algorithms, including the Proposed Algorithm, using three key metrics: F1 Score, AUC-PR (Area Under the Precision-Recall Curve), and ROC-AUC (Area Under the Receiver Operating Characteristic Curve). This visualization highlights the strengths and weaknesses of each algorithm across these important performance metrics, providing a comprehensive view of their comparative effectiveness.

strong classification ability. Finally, ANN and Deep NN exhibit moderate performance across all metrics, lacking a clear specialization. With its balanced performance, the Proposed Model stands out as a strong candidate for general use, unlike other algorithms that focus on specific needs.

5.5 Conclusion

This paper presents a novel heterogeneous graph autoencoder with an attention mechanism to extract meaningful patterns from complex graph structures. The encoder generates node embeddings, which are used to form a probabilistic distribution via a variational autoencoder, capturing uncertainty and enabling diverse node sampling. This addresses the first research question effectively. A deep neural network then processes these embeddings to reconstruct the original node representations, enhancing their quality within the heterogeneous graph. Reconstruction errors from the decoder are analyzed to distinguish fraudulent from non-fraudulent transactions, with a simple search algorithm determining an optimal threshold, addressing the second research question. The proposed model is benchmarked against state-of-the-art methods, including GraphSAGE and FI-GRL, consistently outperforming these baselines and resolving the third research

question.

Despite its strengths, the model has limitations. Scalability is a challenge for very large graphs, as time complexity remains high despite optimizations like neighbor sampling. Performance depends heavily on hyperparameter tuning, including embedding dimensions, attention heads, and network depth. The model's generalizability to datasets with different structural properties or domains beyond finance is untested. It also relies on rich node and edge attributes, which may be unavailable in some real-world scenarios. Additionally, the static graph assumption limits its ability to capture evolving fraud patterns, and the model lacks interpretability, hindering explanation of predictions. The Gaussian-based thresholding approach may be suboptimal for datasets with non-Gaussian error distributions.

These limitations suggest several directions for future work. Advanced sampling techniques, automated hyperparameter optimization, and transfer learning could enhance scalability, robustness, and adaptability. Incorporating temporal graph modeling would enable detection of dynamic fraud patterns, while attention visualization and explainability methods could improve interpretability. Replacing static thresholding with adaptive or non-parametric approaches may improve anomaly detection. Finally, integrating the autoencoder with supervised or rule-based methods could boost performance and practicality in real-world fraud detection systems.

Statements and Declarations

Competing Interests

The authors declare that there are no competing interests associated with this research work.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Informed Consent

Informed consent was obtained from all individual participants included in the study.

Data Availability

The datasets generated and/or analyzed during the current study are available in upon reasonable request from the corresponding author.

AI Assistant

This article acknowledges the use of ProWritingAid, an AI-powered writing assistance tool, to enhance grammar, readability, and overall writing quality. The tool was employed to identify and correct grammatical errors, improve sentence structure, and refine word choice while ensuring clarity and coherence. It is affirmed that all substantive ideas, analysis, and conclusions presented in this document are original and solely attributable to the author(s). ProWritingAid was used exclusively for linguistic refinement and did not influence the core content or arguments.

References

[1] Oluwatobi Noah Akande, Sanjay Misra, Hakeem Babalola Akande, Jonathan Oluranti, and Robertas Damasevicius. A supervised approach to credit card fraud detection using an artificial neural network. In Hector Florez and Ma Florencia Pollo-Cattaneo, editors, *Applied Informatics*, pages 13–25, Cham, 2021. Springer International Publishing.

[2] Abdulalem Ali, Shukor Abd Razak, Siti Hajar Othman, Taiseer Abdalla Elfadil Eisa, Arafat Al-Dhaqm, Maged Nasser, Tusneem Elhassan, Hashim Elshafie, and Abdu Saif. Financial fraud detection based on machine learning: A systematic literature review. *Applied Sciences*, 12(19):9637, Sep 2022.

[3] Ranjeet Kumar Ranjan Amit Singh and Abhishek Tiwari. Credit card fraud detection under extreme imbalanced data: A comparative study of data-level algorithms. *Journal of Experimental & Theoretical Artificial Intelligence*, 34(4):571–598, 2022.

[4] Vinay Arora, Rohan Singh Leekha, Kyungroul Lee, and Aman Kataria. Facilitating user authorization from imbalanced data logs of credit cards using artificial intelligence. *Mobile Information Systems*, 2020:8885269, Oct 2020.

[5] Deyu Bo. *Homogeneous Graph Neural Networks*, pages 27–59. Springer International Publishing, Cham, 2023.

[6] Paula Branco, Luís Torgo, and Rita P. Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv.*, 49(2), aug 2016.

[7] Jason Brownlee. How to combine oversampling and undersampling for imbalanced classification. `https://shorturl.at/J4d2X`, 2021. Accessed: October, 2023.

[8] Patience Chew Yee Cheah, Yue Yang, and Boon Giin Lee. Enhancing financial fraud detection through addressing class imbalance using hybrid smote-gan techniques. *International Journal of Financial Studies*, 11(3), 2023.

[9] Dawei Cheng, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. Graph neural network for fraud detection via spatial-temporal attention. *IEEE Transactions on Knowledge and Data Engineering*, 34(8):3800–3813, 2022.

[10] Sarah A Ebiaredoh-Mienye, E Esenogho, and Theo G Swart. Artificial neural network technique for improving prediction of credit card default: A stacked sparse autoencoder approach. *International Journal of Electrical and Computer Engineering*, 11(5):4392, 2021.

[11] Sarah A Ebiaredoh-Mienye, Ebenezer Esenogho, and Theo G Swart. Integrating enhanced sparse autoencoder-based artificial neural network technique and softmax regression for medical diagnosis. *Electronics*, 9(11):1963, 2020.

[12] Sarah A Ebiaredoh-Mienye, Theo G Swart, Ebenezer Esenogho, and Ibomoiye Domor Mienye. A machine learning method with filter-based feature selection for improved prediction of chronic kidney disease. *Bioengineering*, 9(8):350, 2022.

[13] Ebenezer Esenogho, Ibomoiye Domor Mienye, Theo G. Swart, Kehinde Aruleba, and George Obaido. A neural network ensemble with feature engineering for improved credit card fraud detection. *IEEE Access*, 10:16400–16407, 2022.

[14] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018.

[15] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, WWW '20, page 2704–2710, New York, NY, USA, 2020. Association for Computing Machinery.

[16] SK Saddam Hussain, E Sai Charan Reddy, K Gangadhar Akshay, and T Akanksha. Fraud detection in credit card transactions using svm and random forest algorithms. In *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 1013–1017. IEEE, 2021.

[17] Emmanuel Ileberi, Yanxia Sun, and Zenghui Wang. A machine learning based credit card fraud detection using the ga algorithm for feature selection. *Journal of Big Data*, 9(1):24, 2022.

[18] Fayaz Itoo, Meenakshi, and Satwinder Singh. Comparison and analysis of logistic regression, naïve bayes and knn machine learning algorithms for credit card fraud detection. *International Journal of Information Technology*, 13(4):1503–1511, Aug 2021.

[19] Rongrong Jing, Hu Tian, Gang Zhou, Xingwei Zhang, Xiaolong Zheng, and Daniel Dajun Zeng. A gnn-based few-shot learning model on the credit card fraud detection. In *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*, pages 320–323, 2021.

[20] Kartik2112. Credit card transactions fraud detection dataset. Retrieved 2023-09-21 from `https://www.kaggle.com/datasets/kartik2112/fraud-detection`, 2021.

[21] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[22] Tzu-Hsuan Lin and Jehn-Ruey Jiang. Credit card fraud detection with autoencoder and probabilistic random forest. *Mathematics*, 9(21), 2021.

[23] Yuhua Ling, Ran Zhang, Mingcan Cen, Xunao Wang, and M. Jiang. Cost-sensitive heterogeneous integration for credit card fraud detection. In *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 750–757, 2021.

[24] Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. Pick and choose: A gnn-based imbalanced learning approach for fraud detection. In *Proceedings of the Web Conference 2021*, WWW '21, page 3168–3177, New York, NY, USA, 2021. Association for Computing Machinery.

[25] Zhiwei Liu, Yingtong Dou, Philip S. Yu, Yutong Deng, and Hao Peng. Alleviating the inconsistency problem of applying graph neural network to fraud detection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1569–1572, New York, NY, USA, 2020. Association for Computing Machinery.

[26] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 2077–2085, New York, NY, USA, 2018. Association for Computing Machinery.

[27] Yvan Lucas, Pierre-Edouard Portier, Léa Laporte, Liyun He-Guelton, Olivier Caelen, Michael Granitzer, and Sylvie Calabretto. Towards automated feature engineering for credit card fraud detection using multi-perspective hmms. *Future Generation Computer Systems*, 102:393–402, 2020.

[28] Ibomoiye Domor Mienye and Yanxia Sun. A deep learning ensemble with data resampling for credit card fraud detection. *IEEE Access*, 11:30628–30638, 2023.

[29] Neptune.ai. F1 score, accuracy, roc auc, pr auc: Evaluation metrics you need to know. `https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc`, 2023. Accessed: 2024-05-25.

[30] David M. W. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *ArXiv*, abs/2010.16061, 2011.

[31] Kuldeep Randhawa, Chu Kiong Loo, Manjeevan Seera, Chee Peng Lim, and Asoke K. Nandi. Credit card fraud detection using adaboost and majority voting. *IEEE Access*, 6:14277–14284, 2018.

[32] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. Xfraud: Explainable fraud transaction detection. *Proc. VLDB Endow.*, 15(3):427–436, nov 2021.

[33] Asha RB and Suresh Kumar KR. Credit card fraud detection using artificial neural network. *Global Transitions Proceedings*, 2(1):35–41, 2021. 1st International Conference on Advances in Information, Computing and Trends in Data Engineering (AICDE - 2020).

[34] William N. Robinson and Andrea Aria. Sequential fraud detection for prepaid cards using hidden markov model divergence. *Expert Systems with Applications*, 91:235–251, 2018.

[35] S K Saddam Hussain, E Sai Charan Reddy, K Gangadhar Akshay, and T Akanksha. Fraud detection in credit card transactions using svm and random forest algorithms. In *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 1013–1017, 2021.

[36] Xiang Sheng, Yifan Li, Zhiyuan Liu, and Maosong Sun. Semi-supervised credit card fraud detection via attribute-driven graph representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1234–1241, 2023.

[37] Chuan Shi, Xiao Wang, and Philip S. Yu. *Structure-Preserved Heterogeneous Graph Representation*, pages 29–69. Springer Singapore, Singapore, 2022.

[38] Abhinav Srivastava, Amlan Kundu, Shamik Sural, and Arun Majumdar. Credit card fraud detection using hidden markov model. *IEEE Transactions on Dependable and Secure Computing*, 5(1):37–48, 2008.

[39] Hao Tang, Cheng Wang, Jianguo Zheng, and Changjun Jiang. Enabling graph neural networks for semi-supervised risk prediction in online credit loan services. *ACM Trans. Intell. Syst. Technol.*, sep 2023. Just Accepted.

[40] Huang Tingfei, Cheng Guangquan, and Huang Kuihua. Using variational auto encoding in credit card fraud detection. *IEEE Access*, 8:149841–149853, 2020.

[41] Rafaël Van Belle, Charles Van Damme, Hendrik Tytgat, and Jochen De Weerdt. Inductive graph representation learning for fraud detection. *Expert Systems with Applications*, 193:116463, 2022.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[44] Fangyuan Yang, Kang Wang, Lisha Sun, Mengjiao Zhai, Jiejie Song, and Hong Wang. A hybrid sampling algorithm combining synthetic minority over-sampling technique and edited nearest neighbor for missed abortion diagnosis. *BMC Medical Informatics and Decision Making*, 22(1):344, 2022.

[45] Bingxu Zhang, Changjun Fan, Shixuan Liu, Kuihua Huang, Xiang Zhao, Jincai Huang, and Zhong Liu. The expressive power of graph neural networks: A survey, 2023.

[46] Zhong-Liang Zhang, Rui-Rui Peng, Yuan-Peng Ruan, Jian Wu, and Xing-Gang Luo. Esmote: an overproduce-and-choose synthetic examples generation strategy based on evolutionary computation. *Neural Computing and Applications*, 35(9):6891–6977, 2023.

[47] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

Authors

Sudipta Majumder earned his PhD from NERIST, Arunachal Pradesh, India. His research interest include IOT, cyber security and optimization techniques.

Moirangthem Tiken Singh earned his PhD from Dibrugarh University, India. His research interests include cybersecurity, semantic communication, and graph learning.

Rabinder Kumar Prasad earned his PhD from Dibrugarh University, India. His research interests span cybersecurity, data mining, and unsupervised learning.

Abhijit Boruah earned his PhD from Dibrugarh University, India. His research interests include computer vision, robot kinematics, and ontology design.

Gurumayum Robert Michael earned his PhD from Dibrugarh University, India. India. His research interests include Emotion recognition from speech, IoT and Circuit Design.

N K Kaphungkui received his PhD from Dibrugarh University, India. His research interests include speech processing and electronic circuit design.

N. Hemarjit Singh is pursuing his PhD at Dibrugarh University, India. His research interests include intelligent system design, wireless sensor networks, AI/ML for sensor systems, and industrial automation.