scientific reports

Check for updates

OPEN

HMOA-GNN: adaptive adversarial GraphSAGE with hierarchical hybrid sampling and metric-optimized graph construction for credit card fraud detection

Lina Ni[1,2], Xuqiang Li[1], Yuewei Zhou[1], Hang Qi[1], Xiaohui Man[1] & Jinquan Zhang[1]✉

Accurate credit card fraud detection is vital for protecting financial systems and reducing economic losses. Graph neural networks (GNNs) have shown strong potential by capturing complex patterns in transaction networks. However, existing GNN-based approaches exhibit limitations in handling class imbalance, adapting to non-graph transaction data, and capturing the relative importance of features. Therefore, we propose HMOA-GNN, a novel framework for credit card fraud detection designed to handle tabular and highly imbalanced transaction data. First, the density-driven hierarchical hybrid sampling (DEHS) module balances the dataset by generating synthetic fraudulent transactions in dense regions and removing noise. Next, the metric-optimized latent space similarity graph construction (MOLS-GC) module applies metric learning to build graphs that satisfy the homophily assumption. Finally, the Adversarially trained, feature-adaptive GraphSAGE-based model (AdaAdvSAGE) enhances feature aggregation through adversarial learning and adaptive feature selection. Experiments on multiple real-world datasets demonstrate the superior performance of our framework in credit card fraud detection.

Keywords Credit card fraud detection, Hybrid resampling, Graph neural networks, Feature engineering

Credit card fraud, which involves unauthorized use of credit or debit cards including non-consensual transactions and technologically enabled card cloning[1], causes direct financial losses for consumers and adverse consequences such as credit impairment, legal disputes, and privacy breaches, significantly disrupting daily life and financial stability. For financial institutions, the impact is equally severe, encompassing economic losses as well as reputational damage, reduced customer trust, and heightened compliance risks. In response to the growing frequency of fraud incidents, banks and payment platforms are often required to invest substantial resources into fraud investigations, customer compensation, and system upgrades, thereby driving up operational costs[2]. Therefore, credit card fraud has evolved from an individual-level threat into a systemic risk, making the development of efficient and accurate detection methods a pressing issue for financial security.

Traditional rule-based credit card fraud detection (CCFD) methods are often vulnerable to evasion, as fraudsters can imitate legitimate transaction behaviors to bypass detection systems[3,4]. To overcome this limitation, machine learning and deep learning approaches have been employed to uncover latent anomalous patterns within transaction processes, formulating fraud detection either as a binary classification problem or as an anomaly detection task[5,6]. However, most existing methods[7] still analyze transactions in isolation based on static features, which limits their ability to capture sophisticated and evolving fraud strategies. In response, recent research has shifted toward modeling dependencies among transactions. Graph-based approaches[8], particularly graph neural networks (GNNs), can capture both local and global interaction patterns, thereby enhancing the comprehensiveness and generalization of fraud detection systems.

Despite significant advancements in CCFD, several persistent challenges continue to limit the effectiveness and generalizability of existing approaches. One major issue lies in the extreme class imbalance inherent in

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, Shandong, China. [2]Key Laboratory of the Ministry of Education for Embedded System and Service Computing, 201804 Shanghai, China. ✉email: zhangjinquan@sdust.edu.cn

transaction datasets, where fraudulent transactions represent only a minute fraction of the total data. This imbalance hinders the model's ability to distinguish fraudulent behavior accurately[9–11]. Sampling methods can mitigate class imbalance by generating synthetic minority-class transactions or removing a portion of majority-class transactions. However, many existing methods adopt a single oversampling strategy and tend to overlook the underlying density and distributional characteristics of the data[12–14]. This may lead to the generation of noisy or redundant samples near class boundaries. In parallel, undersampling the majority-class may remove informative samples, undermining the model's ability to capture the overall data distribution[15–17]. Therefore, it is necessary to design a hybrid sampling strategy that considers the distribution of transaction data, thereby addressing the challenge of extreme class imbalance in transaction datasets.

In addition, although GNNs have shown remarkable success in modeling graph-structured data[18–20], they are not directly applicable to conventional tabular (i.e., non-graph-structured) transaction data, which often lacks explicit graph structures[21,22]. Constructing an informative and task-aligned graph topology from non-graph-structured transactions is complicated by noise, heterogeneity, and potential violations of the homophily assumption[23]. The homophily assumption states that connected nodes in a graph are likely to share the same label. Moreover, most existing GNN models, especially those that employ GraphSAGE-style aggregation methods, adopt static feature selection and weighting mechanisms[24,25], which fail to capture the varying importance of features across different transaction samples. This limitation reduces the model's expressiveness in identifying complex and evolving fraud patterns. Therefore, constructing a transaction graph topology that adheres to the homophily assumption, uncovering latent behavioral patterns in non-graph-structured transactional data, and accounting for the varying importance of features across different transaction samples are essential for accurately identifying sophisticated fraudulent activities in real-world scenarios.

Therefore, in this paper, we propose a multi-strategy enhanced adaptive adversarial GNN framework, named HMOA-GNN, for CCFD. To mitigate potential transaction data quality degradation resulting from the reliance on a single sampling strategy, we propose a multi-stage hierarchical hybrid sampling strategy, augmented by a density-based sample generation mechanism. Here, the majority class refers to legitimate transactions, whereas the minority class refers to fraudulent transactions. By generating new samples in high-density regions of the minority class, this method enhances discriminative features of fraudulent samples while minimizing noise, leading to more accurate decision boundaries. Considering the limitations of traditional GNNs when applied to non-graph-structured transaction data, we draw inspiration from the homophily assumption[23]. Based on this idea, we design a metric-optimized latent space similarity graph construction method to build a transaction graph that conforms to the assumption. This graph mapping method effectively uncovers latent behavioral patterns and complex relationships in non-graph-structured transaction data, thereby extending the applicability of GNNs in fraud detection and establishing a well-founded graph-structural foundation for subsequent feature representation learning.

In addition, to overcome the inadequacy of feature aggregation mechanisms in existing GNN methods, particularly their inability to capture the relative importance of features across different transaction samples, we present an adversarially trained, feature-adaptive GraphSAGE-based model. This model employs an adversarial learning strategy to adaptively estimate the contribution of each transaction feature to the classification outcome and dynamically adjusts their weights, thereby enhancing feature utilization during the aggregation process. By integrating the structural information from a graph constructed under the homophily assumption, the model effectively captures complex dependencies among transaction features while introducing inter-layer residual connections to mitigate over-smoothing, thereby enhancing its capability to fraud detection.

The contributions of this work are as follows:

1. We propose *HMOA-GNN*, a novel framework for CCFD designed to address tabular and highly imbalanced transaction data. Specifically, it includes a Density-driven *H*ierarchical Hybrid Sampling (DEHS) module, a Metric-Optimized Latent Space Similarity Graph Construction (MOLS-GC) module, and an Adversarially trained, Feature-Adaptive GraphSAGE-based model (AdaAdvSAGE).
2. We propose the DEHS module, which employs a hybrid sampling strategy to hierarchically identify central and boundary samples, generate synthetic minority samples guided by local density distributions, and eliminate noisy or redundant data. This approach aims to construct a more balanced and informative training dataset, thereby alleviating the effects of extreme class imbalance.
3. In order to construct a transaction graph topology that aligns with the homophily assumption, we present the MOLS-GC method, which employs metric learning to refine transaction embeddings and constructs similarity graphs in the latent space, thus promoting the clustering of transactions belonging to the same class and enhancing the model's structural representation.
4. Aiming to overcome the limitations of GraphSAGE in feature-level discrimination and overall robustness, we propose AdaAdvSAGE, which integrates adversarial training with an adaptive feature selection module and introduces inter-layer residual connections to alleviate over-smoothing in deep layers. This architecture produces more discriminative and robust node representations, thereby enhancing performance in fraud detection tasks.

The remainder of this paper is organized as follows. Section 2 introduces the research background and related work. Section 3 presents the preliminary knowledge. Section 4 describes the proposed method. Section 5 introduces the experimental design and results. Section 6 summarizes the paper.

Related works

We summarize the related work in two main areas: (1) strategies for addressing class imbalance in datasets; (2) the application of GNNs in CCFD.

Sampling methods for class imbalance

Class imbalance in transaction datasets remains a prevalent and challenging issue in CCFD tasks. Numerous sampling strategies have been proposed to effectively mitigate the issue of class imbalance, with undersampling methods aiming to reduce the majority class while preserving data distribution. Kumar et al.[15] employed an entropy and neighborhood-based approach to eliminate low-entropy majority samples from overlapping regions, reducing redundancy. Sun et al.[16] introduced a kernel-based method to remove majority samples from high-density minority areas, mitigating information loss. Zhu et al.[26] employed clustering to identify and discard noisy samples based on hyperspheres around cluster centers.

Oversampling techniques increase minority samples by generating synthetic samples. Chawla et al.[12] developed SMOTE, which interpolates between minority samples to balance the data. Ni et al.[27] refined this with a spiral oversampling strategy to reduce overlap. Li et al.[28] proposed a subspace-based method to maintain original distribution and prevent decision boundary shifts. Maldonado et al.[13] introduced FW-SMOTE, using weighted Minkowski distance to address high-dimensional data.

Hybrid sampling methods combines undersampling and oversampling to leverage their strengths. Lin et al.[29] studied the sequence of applying each and its impact on imbalance. Guo et al.[30] proposed a hybrid of Tomek links, BIRCH clustering, and B-SMOTE to improve intra-class and inter-class balance. Alamri and Ykhlef[31] presented a dynamic hybrid method integrating Bagging, which adapts sampling ratios and targets hard-to-classify samples to enhance robustness.

However, existing hybrid sampling methods often suffer from limited adaptability, relying on fixed strategies that may not generalize well across datasets with varying class overlap or noise. They also risk discarding useful information or generating suboptimal samples due to insufficient consideration of data structure.

Graph neural network for credit card fraud detection

GNNs have achieved significant progress in financial fraud detection[32]. Inspired by Convolutional Neural Networks (CNNs), GNNs extend convolution operations to non-Euclidean graph-structured data by recursively aggregating features from neighbor nodes to update target node representations, enabling modeling of entities such as accounts and transactions[33]. Among them, GraphSAGE[25] introduces an inductive message-passing mechanism in the spatial domain, enabling efficient representation learning for previously unseen nodes by sampling and aggregating their local neighbors. However, it treats the features of all neighbor nodes uniformly during aggregation, which constrains its capacity to capture complex dependencies among transaction features arising from diverse behavioral patterns. Li et al.[34] proposed MG-HRL, a multi-view graph-based hierarchical representation learning method that models transaction networks as heterogeneous information networks with six meta-paths to mine correlations among users and employs heterogeneous hypergraph representation learning to capture high-order representations of transaction subgraphs, achieving superior performance in detecting organized money laundering groups. Yang et al.[35] proposed a multiview fusion neural network (FMvPCI) that integrates multiview graph convolutional encoding with fuzzy clustering to unify protein embeddings and cluster memberships, thereby enhancing the accuracy of protein complex identification. Su et al.[36] proposed an interpretable and generalizable transformer-based graph representation learning framework that integrates multi-omics data with both homogeneous and heterogeneous biological network topologies to achieve accurate cancer gene prediction across pan-cancer and cancer-specific scenarios. Yang et al.[37] proposed a variational Bayesian learning–based link-driven attributed graph clustering method (LCAAG), which infers node cluster labels from link-level modeling and achieves superior accuracy and scalability. Liu et al.[38] proposed PSAGNN, a novel model that employs phased optimization, biased perturbation, and weighted penalties to exploit interbank preferences and scale-free network properties, effectively countering feature and structural poisoning attacks for superior interbank credit rating prediction.

Despite their effectiveness in many tasks, GNNs face challenges in CCFD due to the incompatibility between tabular transaction data and the input format required by GNNs, as such data lacks inherent graph structure. Qiao et al.[39] provided a systematic review of major methods for graph construction and learning, ranging from general machine learning approaches to specific applications. However, their discussion on emerging deep learning-based GNN techniques remains insufficient. Carneiro and Zhao[40] analyzed four graph construction methods based on K-nearest neighbors (KNN) and ϵ-neighborhood (ϵNN) criteria. However, these graph construction approach relies on Euclidean distance in the original high-dimensional space, where such a metric becomes ineffective at distinguishing sample proximity. Consequently, the resulting graph topology often fails to capture the intrinsic relationships within the data, thereby hindering the construction of a transaction graph consistent with the homophily assumption.

Therefore, existing GNN-based approaches for CCFD still face limitations in capturing the complex dependencies among transaction features induced by diverse behavioral patterns. Moreover, these methods often overlook the challenges posed by high-dimensional, non-graph-structured transaction data, where traditional distance metrics become ineffective and the underlying intricate relational patterns remain insufficiently explored and modeled.

Preliminary

This section introduces definitions related to transaction data, graph structures, message passing and aggregation, followed by an introduction to task aims.

Tabular transaction data

We utilize a dataset consisting of a single transaction record of cardholder. Each record represents a distinct transaction event, rather than a monetary transfer between two peer entities.

Formally, each transaction is represented as a tuple:

$$\langle feature_1, feature_2, \ldots, feature_n \rangle \tag{1}$$

where each $feature_i$ corresponds to an attribute such as:

- *Amount*: the monetary value of the purchase,
- *Timestamp*: the time at which the transaction occurred,
- *(Optional)*: additional fields such as merchant category code, transaction location, device type, or card presence indicators.

Tabular transaction dataset is structured as a collection of such records, with no inherent graph or network structure. This data representation allows for the extraction of both static features (e.g., transaction amount, merchant category) and behavioral patterns (e.g., spending frequency, time-of-day preferences). It serves as the foundation for downstream fraud detection tasks.

Graph-structured transaction data

We construct a graph-structured transaction dataset $G = (V, E)$, where each node $v_i \in V$ is associated with a feature vector v_i, and the initial node embedding is defined as $\mathrm{h}_i^0 = \mathrm{v}_i$. The embedding of node v_i at the k-th layer is denoted by h_i^k. To enforce the homophily assumption, an undirected edge is established between two nodes if their corresponding transactions exhibit high similarity. It is important to note that each transaction in the original dataset is represented as a node, which is referred to as a transaction node, in the graph-structured dataset.

Message passing and aggregation

We adopt the message-passing framework from GNNs to learn representations over transaction graphs.

Let $v_i \in V$ denote the i-th node in the graph. At each layer k, the node v_i embedding h_i^k is updated through a two-step process: message aggregation from neighbor nodes and embedding update. The general form of the message-passing update is

$$\mathrm{h}_i^k = \sigma\left(W^k \cdot \mathrm{Agg}^k\left(\{\mathrm{h}_u^{k-1} : u \in \mathscr{N}_{k_{\mathrm{NN}}}(v_i)\} \cup \{\mathrm{h}_i^{k-1}\}\right)\right) \tag{2}$$

where $\mathscr{N}_{k_{\mathrm{NN}}}(v_i)$ denotes the set of k_{NN} nearest neighbors of node v_i, $\mathrm{Agg}^k(\cdot)$ is a differentiable, permutation-invariant function (e.g., mean, LSTM, or pooling), W^k is a trainable weight matrix at layer k, σ is a non-linear activation function, such as ReLU.

Aims

CCFD aims to identify whether a given transaction is fraudulent. This task is inherently a binary classification problem, where the goal is to learn a classifier

$$f : \mathbb{R}^n \to \{0, 1\} \tag{3}$$

where $\mathrm{x} \in \mathbb{R}^n$ represents the feature vector of a single transaction, $y \in \{0, 1\}$ denotes the corresponding label, with $y = 1$ indicating a fraudulent transaction and $y = 0$ indicating a legitimate transaction, and $f(\mathrm{x})$ denotes the model's prediction of the likelihood that the transaction is fraudulent.

Methods

System model

The architecture of our proposed HMOA-GNN framework, which is illustrated in Fig. 1, comprises four steps:

Step (a): Transaction dataset balancing. To address the inherent class imbalance in transaction datasets, we propose a Density-driven Hierarchical Hybrid Sampling strategy comprising two processing channels: an undersampling channel and an oversampling channel. The undersampling channel initially reduces the proportion of majority-class samples, and the resulting subset is employed to estimate the underlying density distributions of transaction samples. Guided by these density estimates, the oversampling channel synthesizes minority-class samples and applies a noise filtering process to ensure representativeness and diversity. This dual-channel hybrid sampling procedure yields a balanced transaction dataset, effectively mitigating bias introduced by skewed class distributions.

Step (b): Transaction graph construction. We develop a Metric-optimized Latent Space Similarity Graph Construction method based on a metric learning model implemented with an autoencoder architecture. The training objective integrates reconstruction loss and triplet loss to embed the balanced, tabular transaction data into a compact low-dimensional latent space. Subsequently, a KNN-based graph construction strategy is applied to the resulting pairwise distance matrix to generate edge connections between transaction records. This process transforms tabular transaction data into graph-structured representations that conform to the homophily assumption, thereby enabling downstream graph-based learning, while simultaneously mitigating the issue of distance metric degradation in high-dimensional spaces.

Step (c): Transaction node representation learning. We propose AdaAdvSAGE, an adversarially trained and feature-adaptive GraphSAGE-based model designed for transaction node representation learning. In this model, the adaptive feature selection module assigns weights to features based on their discriminative relevance across diverse transaction nodes, resulting in weighted representations that preserve critical behavioral signals and support more effective modeling of fraudulent transaction patterns. To enhance the expressive power of graph representations, we design an adversarial training strategy that perturbs the weighted features in the

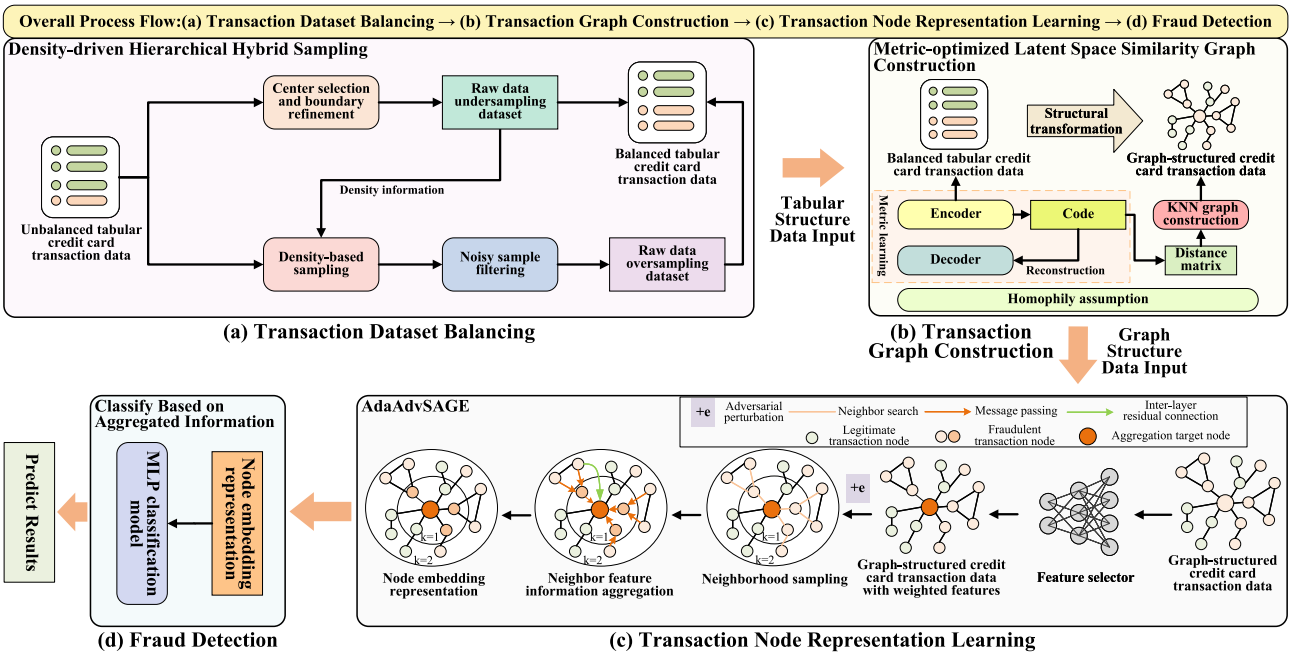


Fig. 1. Framework of HMOA-GNN.

direction of the loss gradient with respect to the weighted features to generate adversarial samples. These are then used alongside the original data during training, promoting the robustness of the model. To alleviate over-smoothing in deeper GNN layers, inter-layer residual connections are incorporated, facilitating the preservation of informative signals across layers. As a result, the model yields high-quality node embeddings that effectively capture aggregated multi-hop neighborhood information.

Step (d): Fraud detection. The learned node embedding representations are fed into a multi-layer perceptron (MLP) classifier, which is trained to infer the probability that a given transaction is fraudulent. This final classification stage operationalizes the preceding representation learning into actionable fraud detection decisions, facilitating timely and accurate identification of illicit activities.

Density-driven hierarchical hybrid sampling

As we all know, credit card transaction data is highly imbalanced. Conventional sampling methods often fail to differentiate informative from noisy samples and neglect the density structure of minority classes, leading to information loss or the generation of overlapping samples in sparse regions. These issues degrade data representativeness and increase the risk of overfitting.

To address these issues, we propose a Density-driven Hierarchical Hybrid Sampling (DEHS) method, whose overall method is shown in Fig. 2. As illustrated, the method comprises three components: center selection and boundary refinement, density-driven selective sampling, and noisy sample filtering, which are organized into two distinct sampling channels. In the undersampling channel, we implement a center-selection and boundary-refinement strategy to reduce majority-class transaction samples through outlier filtering, clustering-based undersampling, and majority-class pruning, thereby preserving representative structural patterns. Subsequently, the resulting subset is used to estimate the density distribution of minority-class transaction samples. Concurrently, in the oversampling channel, these density estimates guide the selective synthesis of additional minority-class samples in high-density regions, with the synthesized fraudulent samples regulated by a noise filtering mechanism to minimize overlap while enhancing representativeness and diversity. This multi-stage, dual-channel hybrid sampling method ultimately yields a well-balanced transaction dataset, effectively mitigating the bias and performance degradation caused by severe class imbalance.

Center selection and boundary refinement strategy

To mitigate the adverse effects of noise and redundancy in the transaction dataset, which can obscure class boundaries and degrade model performance, we adopt a multi-stage sample processing strategy. This strategy comprises outlier filtering, clustering-based undersampling, and majority-class pruning, aiming to preserve representative structures while improving the quality and balance of the training data.

Outlier filtering: First, we introduce a majority-class purification module based on isolation forest. Isolation forest detects outliers effectively, reducing noise and improving training data quality. The specific process is as follows:

The credit card transaction dataset D is divided by transaction type into legitimate transactions set D_0 and fraudulent transactions set D_1. Let the i-th legitimate transaction in D_0 be x_i^{leg}, $x_i^{\text{leg}} \in D_0$. For each transaction, the anomaly score s_i^{asc} is computed individually using the formula as follows:

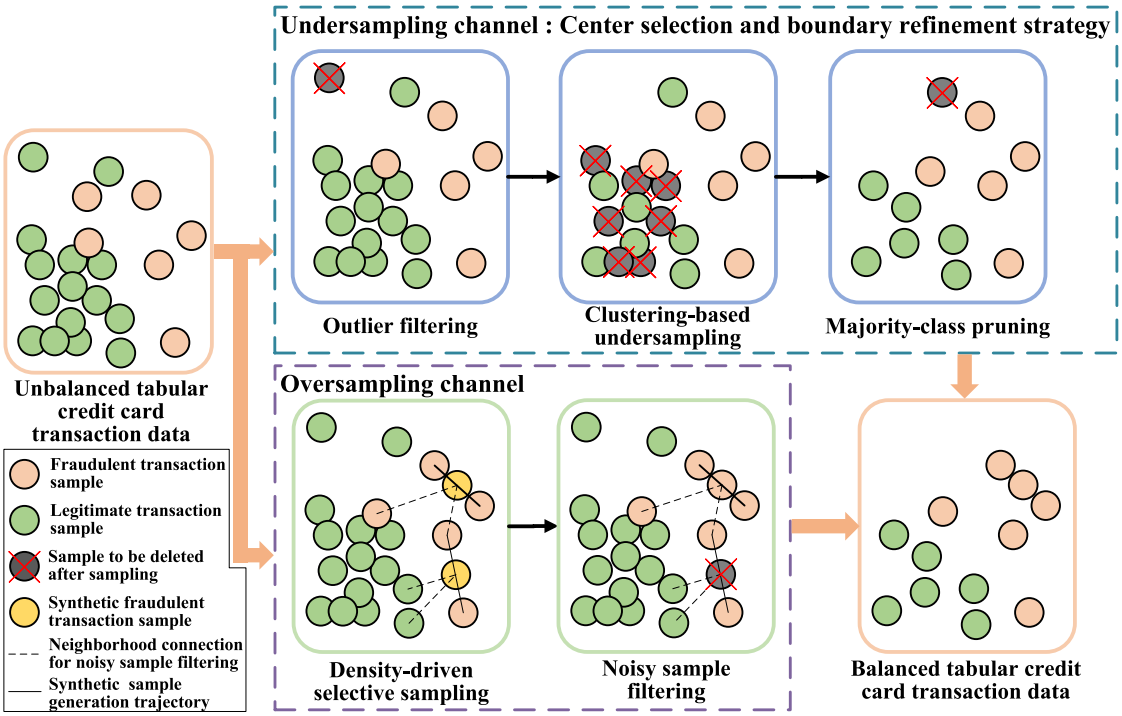


Fig. 2. Process of DEHS method.

$$s_i^{\mathrm{asc}} = s\left(x_i^{\mathrm{leg}}, n_0\right) = 2^{-\frac{E\left(\kappa\left(x_i^{\mathrm{leg}}\right)\right)}{c(n_0)}} \tag{4}$$

where n_0 is the number of legitimate transactions in D_0. $s\left(x_i^{\mathrm{leg}}, n_0\right)$ is the anomaly score of x_i^{leg} in D_0, ranging from $(0, 1]$. The function $\kappa\left(x_i^{\mathrm{leg}}\right)$ calculates the path lengths of x_i^{leg} across all isolation trees. $E\left[\kappa\left(x_i^{\mathrm{leg}}\right)\right]$ corresponds to the average of $\kappa\left(x_i^{\mathrm{leg}}\right)$. The term $c(n_0)$ is a normalization constant, defined as the expected path length of a point in a fully balanced binary tree of size n_0, calculated as follows:

$$\begin{aligned} c\left(n_0\right) &= 2H\left(n_0 - 1\right) - \left(\frac{2\left(n_0 - 1\right)}{n_0}\right) \\ H(n_0 - 1) &= 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n_0 - 1} \end{aligned} \tag{5}$$

where $H(n_0 - 1)$ denotes the $(n_0 - 1)$-th harmonic number, i.e., the sum of the reciprocals of the first $n_0 - 1$ positive integers.

In the context of the Isolation Forest, it represents the expected path length of an unsuccessful search in a binary search tree. For large n_0, $H(n_0 - 1)$ can be approximated by

$$H(n_0 - 1) \approx \ln(n_0 - 1) + \gamma \tag{6}$$

where γ is the Euler–Mascheroni constant, which equals 0.5772 (approximately).

Then, based on the anomaly score s_i^{asc} computed using Eq. (4), the anomaly-score threshold α is set to the k-th highest score among legitimate transactions, thereby identifying those k transactions with scores exceeding α as outliers.

Subsequently, based on the predefined anomaly-score threshold α, transactions in D_0 whose anomaly scores exceed α are identified as outliers and removed. This process yields the refined legitimate transaction dataset D_0^{OF}, which can be expressed as

$$D_0^{\mathrm{OF}} = \left\{x_i^{\mathrm{leg}} \in D_0 | s\left(x_i^{\mathrm{leg}}, n_0\right) \leq \alpha\right\} \tag{7}$$

Clustering-based undersampling: To facilitate the selection of representative samples, we take as input the dataset $D_0^{\mathrm{OF}} = \{x_1, x_2, \ldots, x_{n_0^{\mathrm{OF}}}\}$, obtained from the preceding outlier filtering stage. Each $x_i \in \mathbb{R}^d$ represents the d-dimensional feature vector of the i-th majority-class transaction. This dataset then serves as the basis for the subsequent clustering step.

We subsequently employ a K-means clustering–driven grouping stage on D_0^{OF} to identify and cluster similar majority-class transactions. The algorithm partitions the input samples into k clusters $\{C_1, C_2, \ldots, C_k\}$ by iteratively minimizing the within-cluster Euclidean distance $\mathscr{J}$:

$$\mathscr{J} = \sum_{i=1}^{n_0^{\mathrm{OF}}} \sum_{j=1}^{k} r_{ij} \left\| x_i - \boldsymbol{\mu}_j \right\|^2 \tag{8}$$

where $r_{ij} \in \{0, 1\}$ is the cluster-assignment indicator, such that $r_{ij} = 1$ if and only if $x_i \in C_j$, and $\boldsymbol{\mu}_j \in \mathbb{R}^d$ denotes the centroid of cluster C_j.

After convergence, the representative sample x_j^* in each cluster is defined as:

$$x_j^* = \arg \min_{x_i \in C_j} \left\| x_i - \boldsymbol{\mu}_j \right\|, \quad j = 1, 2, \ldots, k. \tag{9}$$

Then, the set of all selected representatives forms the cluster-based representative set

$$D_{\mathrm{clu}} = \{x_1^*, x_2^*, \ldots, x_k^*\}. \tag{10}$$

This clustering-based undersampling strategy removes most redundant samples while maintaining structural diversity and representativeness, thereby contributing to a more balanced dataset for subsequent processing.

Majority-class pruning: Next, to further eliminate redundant transaction samples, we propose a one-sided selection (OSS) strategy to streamline majority-class samples. It is capable of retaining majority-class samples that are close to the minority class, thereby preserving the decision boundary while reducing noise introduced by redundant majority samples. The OSS strategy is executed according to the following procedure:

(1) Train initial KNN classifier: A KNN classifier is trained using all transactions and predicts sample labels based on Euclidean distance. For each sample x_i, KNN classifier finds the k closest samples in the training set and determines its class by majority vote.

(2) Select informative majority samples: For every majority-class sample $x_i^{\mathrm{leg}} \in D_{\mathrm{clu}}$, check whether it is misclassified as a minority sample by the KNN classifier. If a majority sample is misclassified as a minority, it is likely close to minority samples and considered critical for classification. These misclassified samples are labeled as boundary majority transaction samples set S_{bd}, which is defined as

$$S_{\mathrm{bd}} = \{x_i^{\mathrm{leg}} \in D_{\mathrm{clu}} \mid \mathrm{KNN}(x_i^{\mathrm{leg}}) = \mathrm{fraudulent}\} \tag{11}$$

where $\mathrm{KNN}(x_i^{\mathrm{leg}})$ denotes the class label assigned to the legitimate transaction sample x_i^{leg} by a KNN classifier, which assigns the label corresponding to the most frequent class among the k closest samples in the feature space. In this context, $\mathrm{fraudulent}$ denotes the minority class label in the dataset.

(3) Remove redundant majority samples: Majority-class samples not misclassified by KNN classifier (e.g., $x \in D_{\mathrm{clu}} \backslash S_{\mathrm{bd}}$) are typically located far from the decision boundary and are considered redundant. Such redundant samples can be removed from the dataset to perform undersampling. Thus, the resulting majority transaction samples set D_{maj} is

$$\begin{aligned} D_{\mathrm{maj}} &= D_{\mathrm{clu}} \setminus \left\{ x_i^{\mathrm{leg}} \in D_{\mathrm{clu}} \,\middle|\, \mathrm{KNN}(x_i^{\mathrm{leg}}) \neq \mathrm{fraudulent} \right\} \\ &= S_{\mathrm{bd}} \end{aligned} \tag{12}$$

Density-driven selective sampling

In highly imbalanced credit card transaction datasets, the minority-class transactions in the set D_1 are both scarce and unevenly distributed. Existing oversampling methods[12–14] often fail to effectively exploit samples located near the decision boundary between the majority and minority classes, and their performance is further degraded by the high degree of overlap between synthetic minority samples and majority-class samples.

To overcome these limitations, we propose a density-driven selective sampling method. This method builds upon the sample density computed within D_1, which quantifies the concentration of neighboring samples having the same class label in the feature space. A higher density value indicates that a sample is surrounded by many same-class neighbors, whereas a lower density reflects a sparser local distribution. Leveraging this density information, our method selects minority-class samples from high-density regions within D_1 with a higher probability as base points for generating new synthetic fraudulent transaction samples.

Formally, for a sample $x_i \in D_1$, its sample density ρ_i is defined as:

$$\rho_i = \sum\nolimits_{j=1}^{k} \frac{1}{\mathrm{dist}(x_i, x_j)} \tag{13}$$

where k denotes the number of neighbor points considered in the sample density calculation. The function $\mathrm{dist}(x_i, x_j)$ denotes the Euclidean distance between two transaction samples x_i and x_j, and is defined as

$$\mathrm{dist}(x_i, x_j) = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{14}$$

where d indicates the dimensionality of the feature vectors, and $x_{i,k}$ and $x_{j,k}$ denote the k-th dimensional coordinates of x_i and x_j, respectively. In the specific context of this calculation, x_i refers to the target fraudulent transaction under analysis, and x_j refers to its j-th nearest fraudulent neighbor in the dataset D, with $x_i \neq x_j$.

Additionally, we define the density weight as a normalized measure that determines the probability of selecting a given minority-class sample for the generation of new synthetic samples. The density weight of the i-th sample $\tilde{w}_i$ is defined as follows:

$$\tilde{w}_i = \frac{\rho_i}{\sum_{j=1}^{N_1} \rho_j} \tag{15}$$

where N_1 denotes the number of minority-class samples in D_{maj} and ρ_i is the density value of the i-th sample. A higher density value yields a larger density weight, thereby increasing the likelihood that the corresponding sample will be selected during the new sample generation process.

To effectively steer the oversampling process toward denser regions within the minority class, a density-based weighting mechanism is incorporated into the sample selection strategy. Specifically, each sample x_i from the minority class is selected according to a probability distribution $\mathscr{P}(x_i)$, which is proportional to its normalized density weight $\tilde{w}_i$. The corresponding synthetic fraudulent sample $x_{\mathrm{new}}^{\mathrm{frd}}$ is subsequently generated as follows:

$$\mathscr{P}(x_i) = \frac{\tilde{w}_i}{\sum_{n=1}^{N_1} \tilde{w}_n} \tag{16}$$

$$x_{\mathrm{new}}^{\mathrm{frd}} = x_i + \mathrm{rand}(0,1) \cdot (x_j - x_i), \quad x_i \sim \mathscr{P}(x_i) \tag{17}$$

where $x_{\mathrm{new}}^{\mathrm{frd}}$ denotes a newly synthetic fraudulent transaction sample, $x_j - x_i$ geometrically refers to the line segment between x_j and x_i in the original dataset. New samples are randomly distributed along this segment, with higher-density minority-class samples having a greater chance of being selected as x_i.

The density-based selective sampling method generates new transaction samples by preferentially selecting minority-class transaction samples located in high-density regions. This sampling strategy reduces the redundancy typically introduced by conventional oversampling approaches.

Noisy sample filtering

After generating new synthetic fraudulent transaction samples, we apply a noise filtering mechanism to ensure that these synthetic samples contribute effectively to the subsequent learning of fraudulent patterns without introducing additional noise. In this context, noise refers to synthetic samples that deviate significantly from the distribution of genuine fraudulent data and that may degrade the performance of the classifier if incorporated into training. The neighborhood $\mathscr{N}_k\left(x_{\mathrm{new}}^{\mathrm{frd}}\right)$, comprising the k nearest transactions of the synthetic sample $x_{\mathrm{new}}^{\mathrm{frd}}$ based on the Euclidean distance in Eq. (14), is defined as:

$$\mathscr{N}_k\left(x_{\mathrm{new}}^{\mathrm{frd}}\right) = \{x_1, x_2, \ldots, x_k\} \tag{18}$$

where $x_{\mathrm{new}}^{\mathrm{frd}}$ denotes a synthetic fraudulent sample, $\mathscr{N}_k\left(x_{\mathrm{new}}^{\mathrm{frd}}\right)$ is the set of its k nearest transactions, and x_k is the k-th nearest transaction sample.

To quantify the proportion of fraudulent transactions within the neighborhood $\mathscr{N}_k(x_{\mathrm{new}}^{\mathrm{frd}})$, we introduce the indicator function $\mathrm{I}_{\mathrm{frd}}(\cdot)$, which returns 1 if a transaction is fraudulent (class $= 1$) and 0 otherwise. The formal definition of this indicator function is as follows:

$$\mathrm{I}_{\mathrm{frd}}(x_i) = \begin{cases} 1, & \text{if } x_i \text{ is a fraudulent transaction,} \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Based on this definition, the number of fraudulent transactions among the k nearest neighbors of $x_{\mathrm{new}}^{\mathrm{frd}}$ is given by

$$n_{\mathrm{frd}} = \sum_{i=1}^{k} \mathrm{I}_{\mathrm{frd}}(x_i) \tag{20}$$

where x_i denotes the i-th nearest transaction sample.

If a synthetic sample $x_{\mathrm{new}}^{\mathrm{frd}}$ is surrounded by many minority-class points, it is considered valid and retained; otherwise, it is treated as noise and discarded. To formalize this criterion, we specify a neighbor-count threshold θ_{nbrs} that quantifies the minimum number of minority neighbors required for retention. The filtering rule is given by

$$D_{\min} = \begin{cases} D_{\min} \cup \left\{ x_{\text{new}}^{\text{frd}} \right\}, & \text{if } n_{\text{frd}} \geq \theta_{\text{nbrs}}, \\ D_{\min}, & \text{otherwise.} \end{cases} \tag{21}$$

where $D_{\min}$ initially comprises the original minority-class transaction dataset D_1.

The oversampling and filtering procedure is iteratively applied until $D_{\min}$ is balanced with respect to the majority class, containing an adequate number of synthetic fraudulent transaction samples for subsequent model training.

Therefore, by employing our proposed density-driven hierarchical hybrid sampling method, we obtain the balanced dataset $\mathscr{D}_{\text{bal}}$, which is formally defined as follows:

$$D_{\text{bal}} = D_{\text{maj}} \cup D_{\min} \tag{22}$$

where the reduced majority-class dataset D_{maj} and the filtered synthetic minority-class dataset $D_{\min}$ are merged to obtain a more balanced and representative dataset D_{bal}, which is employed for subsequent model training and evaluation.

The pseudo-code of the DEHS is presented in Algorithm 1. As illustrated in Algorithm 1, in the undersampling channel, the anomaly score of each transaction sample in set D_0 is first computed. Subsequently, the center selection and boundary refinement strategy comprising outlier filtering, clustering-based undersampling, and majority-class pruning is applied to eliminate redundant majority-class samples, yielding a representative set of legitimate transactions D_{maj}. In the oversampling channel, the sample density of each transaction in set D_1 is computed, and corresponding density weights are calculated. Based on these weights, a sufficient number of synthetic minority-class samples are generated via a density-driven selective sampling strategy. After removing noisy samples, the refined synthetic set is combined with D_1 to form the fraudulent transaction sample set $D_{\min}$. Finally, D_{maj} and $D_{\min}$ are merged to construct a balanced credit card transaction dataset D_{bal}.

.

Input: Imbalanced credit card transaction dataset $D = D_0 \cup D_1$, where D_0 denotes legitimate transactions and D_1 denotes fraudulent transactions.

Output: Balanced credit card transaction dataset D_{bal}.

Undersampling channel:
for each legitimate transaction sample $x_i \in D_0$ **do**
    Calculate anomaly score s_i^{asc} using equation (4);
end for
Perform outlier filtering on D_0 using equation (7) to obtain D_0^{OF};
Perform clustering-based undersampling on D_0^{OF} using equation (8) to obtain D_{clu};
Perform majority-class pruning on D_{clu} using equation (11) to obtain D_{maj};
Oversampling channel:
for each fraud transaction sample $x_i \in D_1$ **do**
    Calculate sample density ρ_i using equation (13);
    Calculate density weight w_i using equation (15);
end for
while insufficient number of fraudulent transaction samples obtained **do**
    Generate new sample $x_{\text{new}}^{\text{frd}}$ using equation (17);
    Collect the sampled fraudulent transaction set $D_{\min}$;
end while
Obtain the balanced credit card transaction dataset D_{bal} using equation (22);
return D_{bal}.

Algorithm 1. DEHS.

Metric-optimized latent space similarity graph construction

GNNs have emerged as a powerful paradigm for modeling complex relational structures, exhibiting remarkable capability across a variety of relational learning tasks. Their strength lies in leveraging the graph topology to capture high-order dependencies and propagate information across connected nodes. However, in CCFD, the available transaction records are typically organized as independent tabular entries without an explicit graph structure. The lack of inherent inter-transaction connectivity presents a substantial challenge to the direct application of GNNs, rendering the uncovering of latent behavioral patterns and intricate relational dependencies considerably more difficult. Moreover, the high dimensionality of transaction features leads to the degradation of distance metrics in high-dimensional spaces, hindering effective representation learning and making it difficult to construct graph structures that conform to the homophily assumption.

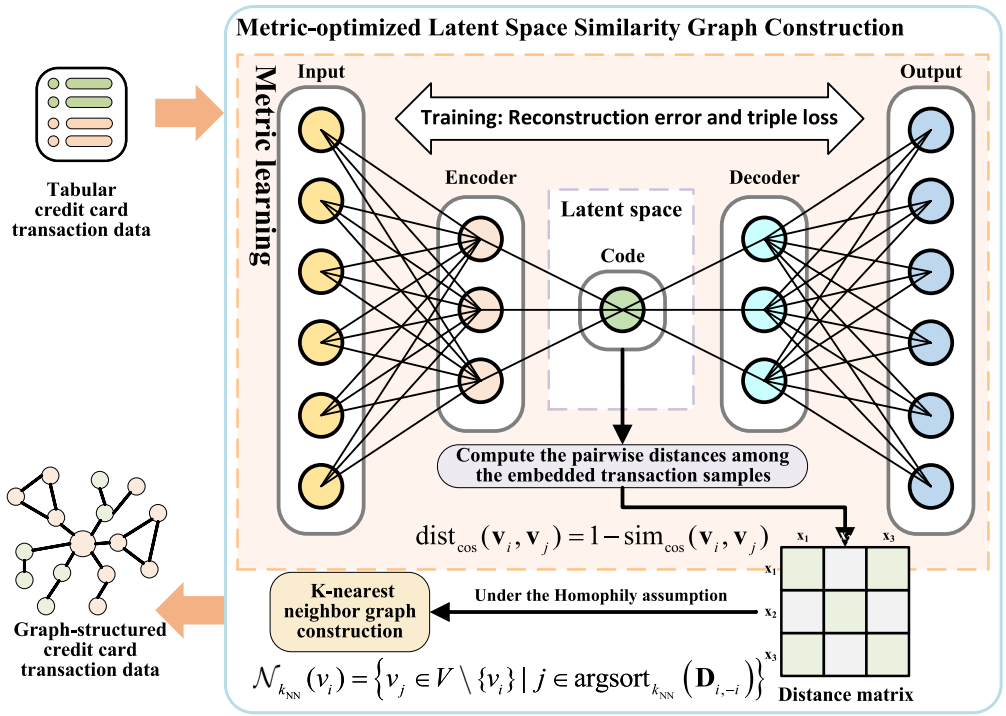


Fig. 3. Process of MOLS-GC method.

To address the above challenges, we propose a Metric-Optimized Latent Space Similarity Graph Construction (MOLS-GC) method. As illustrated in Fig. 3, we first train an autoencoder and adopt its encoder as a parameterized latent mapping function f_ϕ, which projects the original transaction features into a latent space where intra-class distances are minimized and inter-class separations are maximized, thereby yielding semantically coherent neighborhood structures in which fraudulent and legitimate transactions are distinctly segregated. In this optimized space, we compute the pairwise distances among the embedded transaction samples to obtain a distance matrix, upon which a KNN-based graph construction strategy is applied to connect samples exhibiting high mutual similarity. This process yields a transaction graph structure that promotes structural homogeneity and inherently conforms to the homophily assumption.

Latent space optimization via metric learning

To effectively prepare the data for graph construction, it is necessary to address metric degradation and the curse of dimensionality by performing embedding optimization, thereby shaping a latent space in which similarity relationships are both semantically meaningful and structurally coherent. To this end, we adopt a metric learning approach, which enables the model to explicitly learn a task-specific similarity structure by embedding transaction samples from the same class in close proximity while pushing apart those from different classes. This enhances class separability and produces embedding neighborhoods that are well suited for subsequent graph construction, particularly under the homophily assumption commonly leveraged in GNNs.

We define a parameterized latent mapping function $f_\phi : \mathbb{R}^d \to \mathbb{R}^m$, $m \ll d$, which maps an input transaction feature vector $x_i \in \mathbb{R}^d$ to a latent representation $\mathrm{v}_i = f_\phi(x_i) \in \mathbb{R}^m$. In this study, $f_\phi(\cdot)$ is implemented as the encoder component of an autoencoder architecture, with the corresponding decoder $g_\psi(\cdot)$ responsible for reconstructing the input in the original feature space. The reconstructed version of x_i, denoted as $\hat{x}_i$, is obtained as follows:

$$\hat{x}_i = g_\psi(f_\phi(x_i)) \tag{23}$$

To promote both information preservation and discriminative capability in the learned representations, we adopt a composite training objective that integrates reconstruction loss and triplet loss. This joint optimization strategy enables the model to learn embeddings that are not only semantically meaningful but also exhibit enhanced class separability, leading to the construction of more informative neighborhood structures that support effective graph learning tailored to downstream tasks. We now detail the formulation and role of each component in our method.

- *Reconstruction Loss:* The reconstruction loss $\mathscr{L}_{\mathrm{rec}}$ measures the fidelity of the reconstruction relative to the original input, ensuring that the latent representations preserve the intrinsic structural properties of the transaction data. Formally, it is defined as the mean squared error (MSE)

$$\mathscr{L}_{\text{rec}} = \frac{1}{n}\sum_{i=1}^{n} \|x_i - \hat{x}_i\|_2^2 \tag{24}$$

where $x_i \in D_{\text{bal}}$ denotes the *i*-th input sample from the balanced dataset, $\hat{x}_i$ is its reconstruction, and *n* is the total number of samples in the dataset. Minimizing $\mathscr{L}_{\text{rec}}$ constrains f_ϕ to preserve essential information while compressing the data into the latent space.

- *Triplet Loss:* The triplet loss $\mathscr{L}_{\text{tri}}$ imposes a metric structure on the embedding space by minimizing the distance between samples of the same class while maximizing the distance from those of different classes. For each batch, one sample is selected as the *anchor* $\mathrm{v_a}$, samples from the same class are treated as *positives* $\mathrm{v_p}$, and samples from different classes are treated as *negatives* $\mathrm{v_n}$. The loss is given by:

$$\mathscr{L}_{\text{tri}} = \max\left(0, \text{dist}_{\cos}(\mathrm{v_a}, \mathrm{v_p}) - \text{dist}_{\cos}(\mathrm{v_a}, \mathrm{v_n}) + \delta\right) \tag{25}$$

where $\text{dist}_{\cos}(\cdot,\cdot)$ denotes the cosine distance metric as defined in Eq. (30), and $\delta > 0$ is the margin. This formulation enforces that the distance between $\mathrm{v_a}$ and $\mathrm{v_p}$ is at least δ smaller than the distance between $\mathrm{v_a}$ and $\mathrm{v_n}$, thereby reducing intra-class variance and enhancing inter-class separation.

The final objective function combines the reconstruction and triplet losses in a weighted sum:

$$\mathscr{L} = \lambda_{\text{rec}}\mathscr{L}_{\text{rec}} + \lambda_{\text{tri}}\mathscr{L}_{\text{tri}} \tag{26}$$

where $\lambda_{\text{rec}}, \lambda_{\text{tri}} > 0$ are hyperparameters controlling the contribution of each loss term. This joint optimization ensures that f_ϕ learns a structurally coherent and discriminative latent representation space, which facilitates the subsequent construction of transaction graphs that satisfy the homophily assumption.

By applying the trained encoder f_ϕ to all samples in D_{bal}, we obtain the complete set of optimized low-dimensional transaction embeddings:

$$\mathrm{V} = \left\{ \mathrm{v}_i \;\middle|\; \mathrm{v}_i = f_\phi(x_i),\ x_i \in D_{\text{bal}} \right\} \subset \mathbb{R}^m \tag{27}$$

where v_i denotes the *m*-dimensional vector representation of the *i*-th transaction in the metric-optimized latent space.

We then define the node set $V = \{v_1, v_2, \ldots, v_n\}$ such that there exists a bijective correspondence between V and the node embedding set $\mathrm{V} = \{\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_n\} \subset \mathbb{R}^m$. Each embedding $\mathrm{v}_i \in \mathrm{V}$ serves as the attribute representation of the corresponding node $v_i \in V$. The formal definition is given as follows:

$$V = \left\{ v_i \;\middle|\; v_i \leftrightarrow \mathrm{v}_i,\ \mathrm{v}_i \in \mathrm{V} \subset \mathbb{R}^m \right\} \tag{28}$$

KNN-based graph construction

In GNNs, the homophily assumption refers to the principle that similar nodes are more likely to be connected. This assumption is pivotal for the effective application of GNNs in CCFD. To construct a graph structure that complies with this assumption, it is essential to ensure that the edges in the graph accurately reflect semantic similarity among nodes in the embedding space. The KNN method offers a natural and efficient solution, as it locally selects the most representative neighbors based on similarity in the latent space, thereby reinforcing the homophilic structure of the graph.

Upon completion of the autoencoder training phase, the encoder generates low-dimensional latent representations $\{\mathrm{v}_i\}_{i=1}^n$ for all *n* samples, where $\mathrm{v}_i \in \mathbb{R}^m$ denotes the *m*-dimensional embedding of the *i*-th sample. These embeddings are subsequently utilized for similarity-based neighborhood analysis in the latent space.

To enforce the homophily property in the constructed graph, we adopt cosine similarity as the primary similarity measure, defined as

$$\text{sim}_{\cos}(\mathrm{v}_i, \mathrm{v}_j) = \frac{\mathrm{v}_i \cdot \mathrm{v}_j}{\|\mathrm{v}_i\|_2 \, \|\mathrm{v}_j\|_2} \tag{29}$$

where $\mathrm{v}_i \cdot \mathrm{v}_j$ denotes the inner product between v_i and v_j, and $\|\cdot\|_2$ represents the Euclidean norm. For computational convenience, cosine similarity is transformed into a distance metric, which is formally defined as

$$\text{dist}_{\cos}(\mathrm{v}_i, \mathrm{v}_j) = 1 - \text{sim}_{\cos}(\mathrm{v}_i, \mathrm{v}_j) \tag{30}$$

yielding a symmetric distance matrix $\mathrm{D} = [\mathrm{D}_{ij}] \in \mathbb{R}^{n\times n}$, where each entry is given by

$$\mathrm{D}_{ij} = \text{dist}_{\cos}(\mathrm{v}_i, \mathrm{v}_j), \quad \forall\, i, j \in \{1, \ldots, n\}. \tag{31}$$

Based on the distance matrix D, the *k* nearest neighbors of node v_i, denoted by $\mathscr{N}_{k_{\text{NN}}}(v_i)$, are defined as the set of k_{NN} distinct nodes (excluding v_i itself) that have the smallest distances to v_i. Formally, this is given by:

$$\mathcal{N}_{k_{\mathrm{NN}}}(v_i) = \left\{ v_j \in V \setminus \{v_i\} \;\middle|\; j \in \mathrm{argsort}_{k_{\mathrm{NN}}}\left(\mathrm{D}_{i,-i}\right) \right\} \tag{32}$$

where $\mathrm{D}_{i,-i}$ denotes the i-th row of the distance matrix D with the i-th entry (the self-distance) excluded, and $\mathrm{argsort}_{k_{\mathrm{NN}}}(\cdot)$ returns the indices of the k_{NN} smallest values.

The undirected edge set E is then defined according to a symmetric connectivity rule: an edge (v_i, v_j) is established if and only if $v_i \in \mathcal{N}_{k_{\mathrm{NN}}}(v_j)$ or $v_j \in \mathcal{N}_{k_{\mathrm{NN}}}(v_i)$. This ensures bidirectional neighborhood consistency and enhances graph connectivity.

Finally, the graph is represented as

$$\begin{aligned} &G = (V, E), \\ &V = \{v_1, \ldots, v_n\}, \\ &E = \left\{ (v_i, v_j) \;\middle|\; v_i \in \mathcal{N}_{k_{\mathrm{NN}}}(v_j) \text{ or } v_j \in \mathcal{N}_{k_{\mathrm{NN}}}(v_i) \right\} \end{aligned} \tag{33}$$

where V denotes the set of node, and each node v_i is associated with an encoder-derived embedding $\mathrm{v}_i \in \mathbb{R}^m$.

We define the feature matrix $\mathrm{V} \in \mathbb{R}^{n \times m}$ such that the i-th row corresponds to the embedding of node v_i.

The pseudo-code of the MOLS-GC is presented in Algorithm 2. As illustrated in Algorithm 2, the proposed method first defines an autoencoder architecture comprising two components: an encoder and a decoder. The balanced transaction dataset D_{bal} is employed to train the autoencoder, with the objective function specified in Eq. (26). During training, the network parameters are iteratively optimized until convergence. Upon completion, the trained encoder is adopted as the latent mapping function to project the transaction dataset into a low-dimensional latent space, yielding the embedded dataset V. Subsequently, the pairwise distances among the embedded transaction samples in the low-dimensional space are computed to derive a distance matrix. On the basis of this matrix, a KNN-based graph construction procedure is employed, wherein each embedded transaction sample in V is treated as a graph node. Edges are then established between nodes exhibiting high mutual similarity, resulting in a graph $G = (V, E)$ whose structural properties inherently conform to the homophily assumption.

Input: Balanced transaction dataset $D_{\mathrm{bal}} = \{x_i\}_{i=1}^n$, $x_i \in \mathbb{R}^d$; number of samples n; original dimension d; latent dimension m; autoencoder architecture $\mathscr{A} = (f_\phi : \mathbb{R}^d \to \mathbb{R}^m,\ g_\psi : \mathbb{R}^m \to \mathbb{R}^d)$; maximum number of iterations T; convergence threshold ε; objective function $\mathscr{L}$ (equation (26)).

Output: Balanced graph-structured credit card transaction dataset $G = (V, E)$.

Latent space optimization via metric learning:
Define encoder f_ϕ and decoder g_ψ composing $\mathscr{A}$;
Initialize parameters (ϕ, ψ) and optimizer;
for $t = 1$ to T or until $|\mathscr{L}^{(t)} - \mathscr{L}^{(t-1)}| < \varepsilon$ **do**
    Update (ϕ, ψ) by minimizing equation (26) on D_{bal} via backpropagation;
end for
Obtain trained parameter$(\phi^\star, \psi^\star)$;
for each $x_i \in D_{\mathrm{bal}}$ **do**
    Compute the latent representation $\mathbf{v}_i \leftarrow f_{\phi^\star}(x_i)$;
end for
Collect the embedding set $\mathbf{V} \leftarrow \{\mathbf{v}_i\}_{i=1}^n$;
Obtain the node set V using equation (28);
KNN graph construction:
Initialize edge set $E \leftarrow \emptyset$;
Calculate distance matrix **D** using equation (30);
for each $v_i \in V$ **do**
    Obtain $\mathcal{N}_{k_{\mathrm{NN}}}(v_i)$ using equation (32);
end for
for each $v_i \in V$ **do**
    for each $v_j \in \mathcal{N}_{k_{\mathrm{NN}}}(v_i)$ **do**
        if $v_i \in \mathcal{N}_{k_{\mathrm{NN}}}(v_j)$ **or** $v_j \in \mathcal{N}_{k_{\mathrm{NN}}}(v_i)$ **then**
            $E \leftarrow E \cup \{(v_i, v_j)\}$;
        end if
    end for
end for
Construct $G = (V, E)$ using equation (33);
return $G = (V, E)$.

Algorithm 2. MOLS-GC.

AdaAdvSAGE

In spectral-based GCNs[41], the entire adjacency matrix and Laplacian matrix need to be stored during training, which leads to substantial memory overhead. Considering the massive volume of credit card transaction data in real-world applications, spectral-based GCN algorithms become impractical. In contrast, the spatial-based GraphSAGE[25] defines convolution operations directly on the neighborhood space of nodes, thereby enabling mini-batch training and alleviating memory limitations caused by large-scale training data. However, its static feature aggregation mechanism fails to adequately capture the relative importance of features across different transaction instances, reducing the robustness of the model and simultaneously constraining detection performance.

To address this issue, inspired by the GraphSAGE[25], we propose AdaAdvSAGE, a GraphSAGE-based model that incorporates an adaptive feature selection module and an adversarial training mechanism, while introducing inter-layer residual connections to alleviate over-smoothing and gradient vanishing, thereby enabling robust node representation learning. As illustrated in Fig. 4, the graph-structured transaction data is first processed by an adaptive feature selection module, wherein a feature selector, implemented as a feedforward neural network, adaptively adjusts the feature selection weights. This adaptive feature selection mechanism enables the model to selectively amplify discriminative features while attenuating noisy or redundant information, thereby enhancing its capacity to detect subtle and context-specific patterns indicative of fraudulent behavior.

Subsequently, after neighbor sampling, during the message-passing and aggregation phases, each node receives information from its neighbor nodes and integrates these neighbor features with its own representation through an aggregation function to update the node embedding. Meanwhile, we incorporate an adversarial training mechanism that constructs adversarial perturbations in the latent space by following the gradient of the classification loss with respect to the node features. These perturbations are used to generate adversarial samples, guiding the model to learn more robust representations in directions where it is most vulnerable.

Furthermore, to mitigate the problem of over-smoothing in deep graph aggregation layers, we implement inter-layer residual connections between consecutive aggregation layers, wherein the output of the current layer is combined in a weighted manner with the embedding from the preceding layer, ensuring effective information propagation and preservation of discriminative features throughout the network.

Overall, the proposed AdaAdvSAGE constitutes a unified GNN model tailored specifically for CCFD on transaction graphs that conform to the homophily assumption. It ensures both discriminative power and robustness while maintaining stable information flow across network depths. In the following sections, we provide a detailed analysis of each component in sequence.

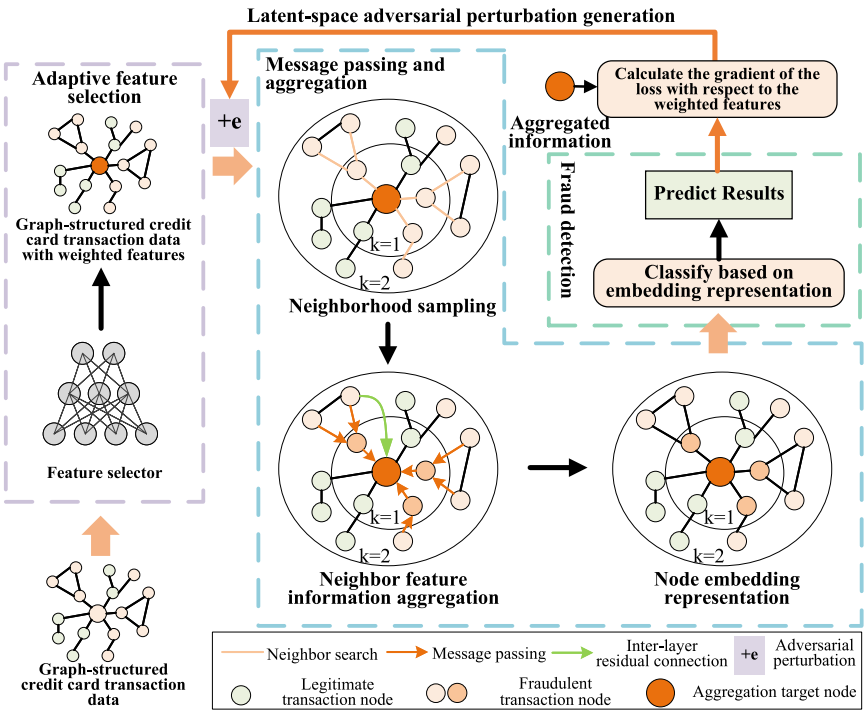


Figure 4. Process of AdaAdvSAGE.

Adaptive feature selection

In real-world CCFD, fraudulent transactions often resemble legitimate ones, particularly near decision boundaries, challenging graph-based models. Conventional GNNs fail to fully capture the relative importance of features in different transaction samples, diluting the discriminative signal and amplifying noise.

To address this limitation, we propose an adaptive feature selection module, which dynamically evaluates and assigns importance to each feature dimension based on the node's local context. This allows the model to adaptively emphasize discriminative features while effectively filtering out task-irrelevant noise.

We consider a graph $G = (V, E)$, where V and E denote the node and edge sets, respectively. For a given node $v_i \in V$, its feature vector is denoted by $\mathrm{v}_i \in \mathbb{R}^m$, where m is the dimensionality of the latent space. To enable adaptive feature selection, we introduce a learnable selector that computes feature-wise importance weights $w_i \in [0, 1]^m$ based on v_i, using a two-layer feedforward neural network. Specifically, the computation of w_i is defined by:

$$w_i = \sigma(W_2\, \delta(W_1\, \mathrm{v}_i + b_1) + b_2) \tag{34}$$

where $W_1 \in \mathbb{R}^{h\times m}$ and $W_2 \in \mathbb{R}^{m\times h}$ are the learnable weight matrices of the two linear transformations, $b_1 \in \mathbb{R}^h$ and $b_2 \in \mathbb{R}^m$ are the bias terms, $\delta(\cdot)$ denotes a non-linear activation function (LeakyReLU in our implementation), and $\sigma(\cdot)$ is an element-wise sigmoid function constraining the output to [0, 1].

The final feature representation $\tilde{\mathrm{v}}_i$ is computed by performing an element-wise product between the original feature vector v_i and the corresponding learned weights w_i, as follows:

$$\tilde{\mathrm{v}}_i = \mathrm{v}_i \odot w_i \tag{35}$$

where $\odot$ means multiplying two vectors element by element (each position multiplied separately). In this way, $\tilde{\mathrm{v}}_i$ is obtained as the weighted version of the feature vector of node v_i.

By adaptively highlighting informative, task-relevant dimensions of v_i while suppressing noisy or redundant ones before neighborhood aggregation, the proposed adaptive feature selection module enhances the model's ability to capture subtle fraudulent patterns in local graph structures and improves the effectiveness of subsequent graph convolutions.

Latent-space adversarial perturbation generation

Considering the presence of subtle and deceptive variations in transaction features, we adopt an adversarial training strategy to improve the robustness of proposed AdaAdvSAGE model. In CCFD, fraudulent transactions often closely resemble legitimate ones, making it difficult for the model to distinguish between them, especially near decision boundaries. Although GNNs are effective at capturing relational structures, they are vulnerable to small perturbations in latent node features. By generating adversarial samples through latent-space perturbations guided by the gradient of the classification loss, the model is encouraged to learn more discriminative and robust representations. The following describes how such adversarial perturbations are constructed and incorporated into the training process.

For a target node $v_i \in V$ with weighted feature vector $\tilde{\mathrm{v}}_i \in \mathbb{R}^m$ (as obtained in Eq. (35)), we define the training loss $\mathscr{L}_{\mathrm{BCE}}$ using the binary cross-entropy (BCE) function, which is suitable for binary classification tasks. Given the ground truth label $y \in 0, 1$ and the predicted probability $\hat{y} \in [0, 1]$, the BCE loss is formulated as

$$\mathscr{L}_{\mathrm{BCE}} = -\left[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right] \tag{36}$$

To construct adversarial perturbations, we compute the gradient of $\mathscr{L}_{\mathrm{BCE}}$ with respect to the weighted feature vector, denoted as $\nabla_{\tilde{\mathrm{v}}_i}\mathscr{L}_{\mathrm{BCE}} \in \mathbb{R}^m$. These gradients guide the injection of adversarial perturbations into node features, thereby enhancing the model's resilience to malicious or noisy variations in the latent space.

The adversarial perturbation is generated following the fast gradient sign method (FGSM):

$$\boldsymbol{\eta}_i = \varepsilon \cdot \mathrm{sign}(\nabla_{\tilde{\mathrm{v}}_i}\mathscr{L}_{\mathrm{BCE}}) \tag{37}$$

where $\varepsilon > 0$ is a hyperparameter controlling the perturbation magnitude, and $\mathrm{sign}(\cdot)$ is applied element-wise.

The perturbed feature vector is then given by

$$\tilde{\mathrm{v}}_i^{\mathrm{adv}} = \tilde{\mathrm{v}}_i + \boldsymbol{\eta}_i \tag{38}$$

During training, both $\tilde{\mathrm{v}}_i$ and $\tilde{\mathrm{v}}_i^{\mathrm{adv}}$ are fed into the subsequent message-passing and aggregation layers. This adversarial training strategy compels the model to learn under the most loss-sensitive conditions, thereby improving its resilience to adversarial perturbations and enhancing its capacity to detect anomalous or fraudulent patterns.

Inter-layer residual connections

Considering that GraphSAGE tends to suffer from over-smoothing in deep aggregation layers, we draw inspiration from ResNet[42] and introduce inter-layer residual connections between successive aggregation layers to mitigate the risk of losing discriminative information during multi-layer message propagation.

Let $v_i \in V$ denote the i-th node in the graph, whose initial representation is defined as $\mathrm{h}_i^0 = \mathrm{z}_i$, where $\mathrm{z}_i \in \{\tilde{\mathrm{v}}_i,\ \tilde{\mathrm{v}}_i^{\mathrm{adv}}\}$ denotes either the weighted input feature or its adversarial counterpart, as specified in Eqs. (35) and (38), respectively.

At each layer $k = 1, \ldots, K$, AdaAdvSAGE updates node representations by aggregating messages from each node and its neighbors using the standard GraphSAGE operator (Eq. 2), identical to GraphSAGE. Let $\hat{\mathrm{h}}_i^k \in \mathbb{R}^{d_k}$ denote the pre-residual output of the k-th message aggregation layer, where d_k is the dimensionality of the node representations at layer k. This output is computed from the previous-layer representation and the local neighborhood of node v_i. It can be represented as:

$$\hat{\mathrm{h}}_i^k = \sigma\left(W^k \cdot \mathrm{Agg}^k\left(\{\mathrm{h}_u^{k-1} : u \in \mathscr{N}_{k_{\mathrm{NN}}}(v_i)\} \cup \{\mathrm{h}_i^{k-1}\}\right)\right) \tag{39}$$

where $\mathscr{N}_{k_{\mathrm{NN}}}(v_i)$ denotes the set of neighbors of node v_i, $\mathrm{Agg}^k(\cdot)$ denotes the mean aggregation function, W^k is a trainable weight matrix at layer k, and σ represents the ReLU activation function.

The post-residual representation is then obtained via a convex combination of the aggregated output and the residual connection:

$$\mathrm{h}_i^k = (1-\gamma)\,\hat{\mathrm{h}}_i^k + \gamma\,\mathrm{h}_i^{k-1} \tag{40}$$

where $\gamma \in [0, 1]$ controls the strength of the residual connection. The parameter γ is treated as a tunable hyperparameter selected via validation.

When the input and output dimensions differ, i.e., $d_k \neq d_{k-1}$, a linear projection is applied to align the dimensions before residual fusion:

$$\mathrm{h}_i^k = (1-\gamma)\,\hat{\mathrm{h}}_i^k + \gamma\,\mathrm{P}^{(k)}\mathrm{h}_i^{k-1}, \quad \mathrm{P}^{(k)} \in \mathbb{R}^{d_k \times d_{k-1}} \tag{41}$$

This residual connection preserves discriminative information from lower layers and stabilizes gradient flow, thereby alleviating over-smoothing while maintaining effective cross-layer information propagation.

Adversarial training and inference

The proposed AdaAdvSAGE, a GraphSAGE-based GNN, integrates adaptive feature selection, adversarial perturbations, neighborhood aggregation, and inter-layer residual connections into a unified architecture. This design yields robust node representations and captures relational patterns in transaction graphs. To further illustrate how these components interact within the model, we provide a step-by-step description of its workflow. We next detail the forward pass, training objective, and inference.

For each node $v_i \in V$, we first obtain its weighted transaction feature vector $\tilde{\mathrm{v}}_i$ using Eq. (35). The corresponding adversarial version $\tilde{\mathrm{v}}_i^{\mathrm{adv}}$ is generated according to Eq. (38). Both representations are passed through a K-layer GNN composed of message aggregation and inter-layer residual connections, as described in Eqs. (40) and (41).

We process the data through the described GNN and obtain the final representation of each node, which subsequently serves as the basis for the classification task. Let h_i^K denote the final hidden representation of node v_i at the K-th layer. For classification, we apply two fully connected layers: the first one transforms the representation into a hidden space with a non-linear activation, while the second applies a sigmoid activation to produce the final predicted probability. Formally, the predicted probability for node v_i is computed as follows:

$$\hat{y}_i = \sigma\left(\mathrm{W}_2^\top \phi\left(\mathrm{W}_1 \mathrm{h}_i^K + b_1\right) + b_2\right) \tag{42}$$

where $\hat{y}_i$ denotes the predicted probability that node v_i belongs to the fraudulent transaction class, $\mathrm{W}_1 \in \mathbb{R}^{d' \times d_K}$ and $b_1 \in \mathbb{R}^{d'}$ are the weights and bias of the first layer, $\phi(\cdot)$ denotes a non-linear activation function such as ReLU, $\mathrm{W}_2 \in \mathbb{R}^{d'}$ and $b_2 \in \mathbb{R}$ are the parameters of the second (output) layer, and $\sigma(\cdot)$ is the sigmoid function that maps the output to [0, 1], the variable d_K denotes the dimension of the node representation h_i^K at the K-th GNN layer, while d' refers to the dimensionality of the intermediate hidden space used for classification. Note that d_K is typically determined by the GNN architecture, whereas d' is a tunable hyperparameter.

To enhance robustness, we adopt an adversarial training strategy, in which small perturbations are applied to each sample along the gradient direction of the loss function with respect to the input, thereby generating adversarial samples. The model is then optimized by minimizing the joint loss over clean and adversarial samples. This strategy improves the stability of the model against input perturbations while preserving accuracy on clean data, encouraging the reliance on more task-relevant and robust features. Specifically, we train the model on both clean and adversarial samples using the average BCE loss:

$$\mathscr{L}_{\mathrm{total}} = \frac{1}{2|V|} \sum_{v_i \in V} \left[\mathscr{L}_{\mathrm{BCE}}(\hat{y}_i, y_i) + \mathscr{L}_{\mathrm{BCE}}(\hat{y}_i^{\mathrm{adv}}, y_i)\right] \tag{43}$$

where $\hat{y}_i$ and $\hat{y}_i^{\mathrm{adv}}$ denote the predicted probability based on clean and adversarial inputs respectively, and $y_i \in \{0, 1\}$ is the true label.

After training, during inference, we discard the adversarial branch and compute the node representation on the clean input. Specifically, we first obtain h_i^K, the final hidden representation of node v_i at the K-th layer. We then use the same classifier as in Eq. (42) to compute the predicted probability for fraud detection:

$$Pred_i = \sigma\left(\mathrm{W}_2^{\top}\,\phi\left(\mathrm{W}_1\,\mathrm{h}_i^{K} + b_1\right) + b_2\right) \tag{44}$$

The output $Pred_i \in [0,1]$ quantifies the probability that transaction v_i is fraudulent. To obtain the final classification result, a decision threshold τ is applied: if $Pred_i \geq \tau$, the transaction is classified as fraudulent; otherwise, it is classified as legitimate. In practice, τ can be set to 0.5. This completes the fraud detection pipeline from adaptive feature selection to robust prediction.

The training and inference procedure for the AdaAdvSAGE model is summarized in Algorithm 3. The model comprises an adaptive feature-selection module, a GraphSAGE backbone with mean-pooling aggregation and inter-layer residual connections, an adversarial training mechanism, and a two-layer classifier. During each forward pass, the feature-selection module assigns feature-wise weights to each node to produce weighted node representations, which are then fed into a K-layer GraphSAGE to capture multi-hop neighborhood information. To improve robustness during training, adversarial samples are generated by perturbing the weighted features in the direction of the loss gradient; both clean and adversarial inputs are used jointly to optimize the network. Inter-layer residual connections between successive aggregation layers mitigate over-smoothing and help preserve discriminative signals across depths. Finally, a two-layer neural classifier operates on the resulting embeddings to output fraud-probability predictions.

Input: Balanced graph structure credit card transaction dataset $G = (V, E)$ with node features $\{\mathbf{v}_i \in \mathbb{R}^m\}_{i=1}^{|V|}$; label set $\{y_i \in \{0,1\}\}_{i=1}^{|V|}$.
Output: Prediction results of fraud probability of credit card transactions *Pred*.

1: **Adaptive feature selection:**
2: **for** each node $v_i \in V$ **do**
3: Compute feature selection weights w_i using equation (34);
4: Compute weighted feature vector $\tilde{\mathbf{v}}_i$ using equation (35);
5: **end for**
6: Collect weighted feature sets $\mathbf{V} \leftarrow \{\tilde{\mathbf{v}}_i\}_{i=1}^{|V|}$;
7: **Adversarial perturbation generation:**
8: **for** each node $v_i \in V$ **do**
9: Initialize $h_i^0 \leftarrow \tilde{\mathbf{v}}_i$;
10: Compute prediction $\hat{y}_i$ via forward propagation of the GNN using equation (42) (see Algorithm 3, line 16-24);
11: Compute gradient $\nabla_{\tilde{\mathbf{v}}_i} \mathscr{L}_{\mathrm{BCE}}$;
12: Generate perturbation η_i using equation (37);
13: Generate adversarial input $\tilde{\mathbf{v}}_i^{\mathrm{adv}}$ using equation (38);
14: **end for**
15: Collect weighted feature sets $\mathbf{V}^{\mathrm{adv}} \leftarrow \{\tilde{\mathbf{v}}_i^{\mathrm{adv}}\}_{i=1}^{|V|}$;
16: **GNN Forward propagation with inter-layer residual connections:**
17: **for** each input variant $\mathbf{z}_i \in \{\tilde{\mathbf{v}}_i, \tilde{\mathbf{v}}_i^{\mathrm{adv}}\}$ **do**
18: Initialize $h_i^0 \leftarrow \mathbf{z}_i$
19: **for** $k = 1$ to K **do**
20: Compute the pre-residual aggregated representation $\hat{h}_i^k$ using equation (39);
21: Compute the post-residual representation $\mathbf{h}_i^k$ using equation (40);
22: **end for**
23: Compute predicted probability $\hat{y}_i$ using equation (42);
24: **end for**
25: **Loss calculation and optimization:**
26: Compute total loss $\mathscr{L}_{\mathrm{total}}$ using equation (43);
27: Update all parameters via backpropagation on $\mathscr{L}_{\mathrm{total}}$;
28: **Credit card fraud detection:**
29: **for** each node $v_i \in V$ **do**
30: Perform adaptive feature selection on $\mathbf{v}_i$ using equation (35) to obtain $\tilde{\mathbf{v}}_i$;
31: Perform forward propagation using clean input only to get final embedding h_i^K;
32: Compute predicted probability $Pred_i$ using equation (44);
33: **end for**
34: **return** *Pred*.

Algorithm 3. AdaAdvSAGE.

Experiment

In this section, we evaluate the performance of the proposed HMOA-GNN framework using three widely recognized benchmark datasets in the domain of CCFD: the European Cardholders Transaction dataset[43], the IEEE-CIS Fraud Detection dataset[44], and the Simulated Credit Card Transactions dataset[45].

In this section, we first describe the experimental setup, including the hardware and software environments, as well as the procedures adopted in our experiments. Subsequently, ablation studies are conducted on the three core modules of the proposed HMOA-GNN framework (DEHS, MOLS-GC, and AdaAdvSAGE) to evaluate their respective contributions to the overall performance. Then, to examine the effectiveness of the proposed DEHS method in mitigating class imbalance, we compare it against four representative data balancing techniques: random undersampling, Synthetic Minority Oversampling Technique (SMOTE)[12], Histogram SMOTE (H-SMOTE)[46], and K-means-based undersampling[47]. The performance of the HMOA-GNN framework, when combined with each of these sampling strategies, is then evaluated on the fraud detection task.

Afterward, to further verify the effectiveness of the proposed HMOA-GNN framework in CCFD tasks, we conducted a comparative analysis with several representative models in this domain. The experimental results demonstrate that our method achieves superior overall performance compared with the existing approaches for CCFD.

Datasets

1. *European Cardholders Transaction dataset*: This Kaggle dataset[43] comprises 284,807 credit card transactions from european cardholders over two days in September 2013, of which only 492 (0.17%) are fraudulent. The severe class imbalance biases models toward legitimate transactions, reducing fraud detection accuracy. The dataset contains no missing values, obviating the need for imputation or feature filtering.
2. *IEEE-CIS Fraud Detection dataset:* Provided by Vesta[44], this dataset includes real e-commerce transactions from September–December 2017. It consists of a transaction table (590,540 entries, 41 features) and an identity table (394 features), yielding 435 features after merging.
3. *Simulated Credit Card Transactions dataset:* This dataset[45] comprises credit card transactions simulated between January 1, 2019, and December 31, 2020, including both legitimate and fraudulent activities. It covers 1,000 customers making transactions with 800 merchants. The dataset was generated using the Sparkov Data Generation tool created by Brandon Harris, and all simulation files were merged and converted into a standardized format.

Performance evaluation metrics

In the field of fraud detection, several evaluation metrics are commonly adopted to address the challenges posed by imbalanced datasets. In this study, we primarily employ Recall, F1-score, and AUC as performance metrics.

The AUC (Area Under the ROC Curve) is computed by plotting the trade-off between the True Positive Rate (TPR) and False Positive Rate (FPR) under varying classification thresholds. TPR and FPR are defined as follows:

$$TPR = \frac{TP}{TP + FN} \tag{45}$$

$$FPR = \frac{FP}{FP + TN} \tag{46}$$

where TP denotes true positives, FN false negatives, FP false positives, and TN true negatives.Accuracy measures the proportion of correctly classified samples in the total dataset

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{47}$$

Precision quantifies the percentage of predicted fraudulent transactions that are indeed fraudulent

$$precision = \frac{TP}{TP + FP} \tag{48}$$

Recall is defined as the proportion of actual fraudulent transactions that are correctly identified by the model. In general, the cost of failing to detect fraudulent transactions is higher than the cost of incorrectly classifying legitimate transactions as fraudulent. Therefore, in fraud detection scenarios, a higher recall is often preferred over precision. The mathematical definition of recall is given as follows:

$$recall = sensitivity = \frac{TP}{P} \tag{49}$$

The F1-score is the harmonic mean of Precision and Recall

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{50}$$

where F1-score balances both the Precision and Recall, making it a robust evaluation metric that helps avoid extreme bias toward one metric over the other.

Experimental setup

Experimental environment

All experiments were conducted on a high-performance computing workstation to ensure reproducibility and reliability of the results. The software environment consisted of Python 3.9.13 as the primary programming language, with deep learning models implemented using PyTorch 2.1.1 and accelerated by CUDA 12.1. The hardware configuration included an Intel Core i9-14900HX processor, a single NVIDIA GeForce RTX 4060 graphics processing unit with 8 GB of memory, and a total of 64 GB of system RAM, providing sufficient computational resources to support efficient training and evaluation of the proposed models.

Preprocessing

We propose the DEHS method to generate synthetic fraudulent transaction samples. The legitimate and fraudulent transactions in the dataset are separated, after which a center selection and boundary refinement strategy is applied to eliminate noise and redundant data from the legitimate transactions. The number of synthetic samples to be generated, along with the density of each minority-class sample, is then computed to obtain the corresponding density weight for each sample. Guided by the density weights, minority-class sample points are strategically selected for synthetic sample generation. Each synthetic sample is created along the line segment connecting the selected sample and its nearest neighbor. Subsequently, the density of each generated sample is evaluated to determine whether it constitutes noise. Noisy samples are discarded, whereas valid samples are retained. This selection–generation cycle is repeated until the desired number of synthetic samples is produced, yielding a balanced dataset.

Data graphization

We employ the MOLS-GC method to convert the non-graph-structured credit card transaction data into graph data. Using the portion of the balanced dataset processed by the DEHS method, we train an autoencoder to obtain a latent mapper. The credit card transaction data employed for training and detection are then passed through the latent mapper to obtain their latent representations in the new latent space. Each transaction's similarity to others in the latent space is computed, and edges are established to its nearest neighbors via the KNN algorithm, forming graph-structured data suitable for GNN input.

Training AdaAdvSAGE model

The graph-structured credit card transaction data are first balanced by the DEHS method. They are then transformed into graph format through the MOLS-GC process, which produces latent-space representations of the transaction nodes. After these steps, the data are fed into the proposed AdaAdvSAGE model.

During training, each transaction node undergoes adaptive feature selection. In this step, feature-wise weights are learned to highlight discriminative dimensions while suppressing noisy or redundant information. Next, adversarial perturbations are generated in the latent space by following the gradient of the classification loss with respect to node features. Both clean and adversarial samples are used together to optimize the model. This strategy improves the robustness of the learned representations. Node embeddings are further updated through GraphSAGE-based neighborhood aggregation with inter-layer residual connections. These connections mitigate over-smoothing and help preserve discriminative signals across layers. Through this process, we obtain the well-trained AdaAdvSAGE model for CCFD.

Method	Hyperparameters and values
DEHS	Isolation Forest contamination: 0.005
	Isolation Forest random state: 42
	Number of neighbors $k = 5$
	Noise filtering neighbor threshold $\theta_{\mathrm{nbrs}} = 3$
MOLS-GC	Encoder hidden layers: [64, 32, 16, 8]
	Decoder hidden layers: [8, 16, 32, 64]
	Activation function: LeakyReLU
	Dropout rate: 0.2
	Triplet loss margin: 1.0
	Relative weights of loss terms: $\lambda_{\mathrm{rec}} = \lambda_{\mathrm{tri}} = 0.5$
	Same-label samples per triplet: 3
	Different-label samples per triplet: 3
	$k = 32$ (for k-NN graph construction)
AdaAdvSAGE	Adversarial perturbation strength $\varepsilon = 0.01$
	Residual connection weight $\gamma = 0.5$
	Feature selector hidden dimension $d_{\mathrm{w1}} = 64$
	Number of SAGE layers: 2
	Sage layer dropout: 0.5

Table 1. Hyperparameter settings of the HMOA-GNN framework.

Fraud detection
After training, the graph-structured credit card transaction data are fed into the trained AdaAdvSAGE model. Each transaction node is first processed by the adaptive feature selection module, and its representation is subsequently propagated through the aggregation layers with residual connections, where the node embeddings are progressively refined. Finally, a two-layer classifier, implemented as a standard multi-layer perceptron, maps the embeddings to fraud probability scores. Transactions are then classified as fraudulent or legitimate according to a decision threshold. This inference procedure ensures clarity, reproducibility, and fairness in comparison.

Hyperparameter setting
During the experimental process, we tuned the hyperparameters and evaluated the resulting models until the optimal set of hyperparameters was identified. The best values of the hyperparameters are presented in Table 1.

Practical considerations and potential deployment

In terms of practical deployment within financial transaction platforms, the HMOA-GNN framework can be encapsulated as a scalable microservice, interfacing with existing business systems through RESTful APIs or gRPC. Prior to entering the detection model, transaction data undergoes preprocessing and graph construction within a distributed data pipeline (e.g., message queues and stream processing frameworks), ensuring stable integration with large-scale and high-velocity transaction streams. Leveraging the inductive learning capability and scalability of the AdaAdvSAGE model, the processed transaction data can be directed to the deployed model for either real-time or batch detection. In real-time mode, the model generates representations by jointly considering transaction features and the neighborhood information within the transaction graph, thereby enabling rapid risk assessment. In batch mode, the framework supports periodic large-scale analyses, which complement real-time detection by providing comprehensive risk monitoring. Furthermore, the system design allows integration with existing rule-based engines or risk-control platforms through modular connectors, enabling hybrid decision support. The architecture also supports horizontal scalability, ensuring robustness against peak transaction volumes and adaptability to the heterogeneous technical infrastructures of different financial institutions.

Ablation experiment

In our ablation studies, we define the baseline as a simplified variant of the HMOA-GNN framework in which all proposed optimization modules, including DEHS, MOLS-GC, and AdaAdvSAGE, are excluded.

Specifically, the baseline does not employ any mechanism to address the inherent class imbalance in the credit card transaction datasets. For graph construction, it adopts a random neighbor connection strategy, where each transaction record is linked to a set of randomly selected counterparts and treated as neighbors. On this graph, the baseline retains only the fundamental GraphSAGE-style message passing and aggregation operations, without incorporating any of the proposed enhancements. This design yields a minimal yet consistent setting, thereby ensuring that observed performance gains can be attributed solely to the integration of DEHS, MOLS-GC, or AdaAdvSAGE modules.

By contrasting the baseline with the module-augmented variants, we are able to quantify the distinct contributions of each component to the overall performance improvement.

Ablation experiment of DEHS module
To validate the effectiveness of the DEHS module, we sampled subsets from each of the three datasets using a fixed random seed (set to 42) to ensure reproducibility. We first examined the performance gains achieved by incorporating the DEHS module into baseline methods, thereby assessing its standalone contribution. Furthermore, we evaluated the overall performance of the HMOA-GNN framework both with and without the DEHS module, in order to quantify its impact on the framework as a whole.

Dataset	Algorithm	Evaluation Metrics				
		Accuracy	Precision	Recall	F1-score	AUC
European	Baseline	0.9745	0.0000	0.0000	0.0000	0.5000
	Baseline w/ DEHS	0.9745(–)	0.0000(–)	0.0000(–)	0.0000(–)	0.5000(–)
	HMOA-GNN	0.9965	0.9400	0.9216	0.9307	0.9975
	HMOA-GNN w/o DEHS	0.9920(↓)	1.0000(↑)	0.6863(↓)	0.8140(↓)	0.9323(↓)
IEEE-CIS	Baseline	0.9815	0.8718	0.5152	0.6476	0.9099
	Baseline w/ DEHS	0.9470(↓)	0.3684(↓)	0.8485(↑)	0.5138(↓)	0.9582(↑)
	HMOA-GNN	0.9805	0.6627	0.8333	0.7383	0.9445
	HMOA-GNN w/o DEHS	0.9825(↑)	0.8298(↑)	0.5909(↓)	0.6903(↓)	0.8995(↓)
Simulated	Baseline	0.9685	0.0000	0.0000	0.0000	0.9386
	Baseline w/ DEHS	0.9545(↓)	0.3474(↑)	0.5323(↑)	0.4204(↑)	0.9095(↓)
	HMOA-GNN	0.9815	0.7049	0.6935	0.6992	0.9177
	HMOA-GNN w/o DEHS	0.9765(↓)	0.8261(↑)	0.3065(↓)	0.4471(↓)	0.8557(↓)

Table 2. Ablation results of DEHS module.

Table 2 presents the ablation results on three benchmark datasets. Overall, the DEHS module demonstrates significant advantages in handling scenarios with extreme class imbalance, particularly on the IEEE-CIS and Simulated datasets. For the Baseline model, the minority class is almost entirely unrecognized without DEHS (e.g., both Recall and F1-score are 0 on the European and Simulated datasets). After incorporating DEHS, Recall improves to 0.8485 on IEEE-CIS and 0.5323 on Simulated, while F1-score increases from 0 to 0.5138 and 0.4204, respectively. These results indicate that DEHS effectively alleviates the detrimental impact of severe imbalance on detection performance, even though it may slightly reduce accuracy and precision under the Baseline model. Furthermore, within the HMOA-GNN framework, removing DEHS leads to a substantial decline in Recall and F1-score (e.g., Recall drops from 0.8333 to 0.6061 on IEEE-CIS, and from 0.6935 to 0.3065 on Simulated), further confirming the importance of DEHS in constructing a more balanced and informative training set.

In contrast, on the European dataset, the results of Baseline and Baseline w/ DEHS remain identical, with AUC consistently fixed at 0.5 and all minority-class metrics equal to 0. This phenomenon can be attributed to the extremely sparse and indistinct distribution of minority classes in this dataset. Under such circumstances, merely generating additional minority samples through DEHS is insufficient for building an effective decision boundary when relying solely on a simple baseline model. Nevertheless, when combined with the HMOA-GNN framework, the contribution of DEHS becomes evident: Recall increases from 0.6863 to 0.9216, and F1-score rises from 0.8140 to 0.9307, despite a slight decrease in precision (from 1.0000 to 0.9400). This demonstrates that the quality enhancement of samples provided by DEHS can be fully exploited under a framework equipped with structural modeling and adaptive feature learning capabilities.

In summary, the DEHS module proves to be a pivotal component in alleviating class imbalance and substantially enhancing the model's capability to detect minority classes, with its efficacy consistently validated across multiple benchmark datasets. The absence of improvement on the European dataset under the baseline model further underscores that data balancing in isolation is inadequate for addressing the challenges of minority class detection. Instead, its full potential can only be realized when integrated with structural modeling strategies that enable more discriminative representation learning.

Ablation experiment of MOLS-GC module

To validate the effectiveness of the MOLS-GC module, we sampled subsets from three datasets using a fixed random seed (set to 42) to ensure reproducibility. We then examined the performance improvements of baseline methods after integrating the MOLS-GC module and further assessed the standalone effectiveness of the module itself. In addition, we compared the overall performance of the HMOA-GNN framework with and without the incorporation of the MOLS-GC module, thereby evaluating its impact on the framework as a whole.

As summarized in Table 3, on the European dataset, the improvement brought by MOLS-GC is particularly significant. While the Baseline model exhibits trivial detection capability (AUC = 0.5000) due to its inability to capture fraudulent behavior under severe class imbalance, the inclusion of MOLS-GC dramatically elevates the AUC to 0.9136, even though the remaining metrics remain unchanged. This remarkable gain indicates that MOLS-GC enables the model to learn meaningful latent relational structures, even in the absence of strong label-driven supervision. The substantial increase in AUC suggests that the similarity graph constructed in the optimized latent space better captures potential homophilic and semantic relationships among transactions, thereby aligning with the homophily assumption fundamental to graph learning. Furthermore, when MOLS-GC is incorporated into the complete HMOA-GNN framework, the AUC further improves to 0.9975, whereas its removal causes a drastic drop to 0.5000. Such a pronounced contrast underscores the indispensable role of MOLS-GC in maintaining structural coherence and ensuring representational integrity throughout the framework.

In summary, the integration of MOLS-GC also leads to improvements in recall and F1-score under the Baseline w/ MOLS-GC configuration, demonstrating that latent-space graph construction enables the model to capture a broader range of fraudulent transactions. Although slight decreases are observed in precision and

Dataset	Algorithm	Evaluation Metrics				
		Accuracy	Precision	Recall	F1-score	AUC
European	Baseline	0.9745	0.0000	0.0000	0.0000	0.5000
	Baseline w/ MOLS-GC	0.9745(–)	0.0000(–)	0.0000(–)	0.0000(–)	0.9136(↑)
	HMOA-GNN	0.9965	0.9400	0.9216	0.9307	0.9975
	HMOA-GNN w/o MOLS-GC	0.0255(↓)	0.0255(↓)	1.0000(↑)	0.0497(↓)	0.5000(↓)
IEEE-CIS	Baseline	0.9815	0.8718	0.5152	0.6476	0.9099
	Baseline w/ MOLS-GC	0.9820(↑)	0.8409(↓)	0.5606(↑)	0.6727(↑)	0.8543(↓)
	HMOA-GNN	0.9805	0.6627	0.8333	0.7383	0.9445
	HMOA-GNN w/o MOLS-GC	0.9745(↓)	0.5641(↓)	1.0000(↑)	0.7213(↓)	0.9971(↑)
Simulated	Baseline	0.9685	0.0000	0.0000	0.0000	0.9386
	Baseline w/ MOLS-GC	0.9690(↑)	0.0000(–)	0.0000(–)	0.0000(–)	0.8831(↓)
	HMOA-GNN	0.9815	0.7049	0.6935	0.6992	0.9177
	HMOA-GNN w/o MOLS-GC	0.9795(↓)	0.7143(↑)	0.5645(↓)	0.6306(↓)	0.9224(↑)

Table 3. Ablation results of MOLS-GC module.

AUC, this trade-off indicates the formation of a more balanced decision boundary that emphasizes detection sensitivity, which is particularly advantageous in practical financial risk detection scenarios. Within the complete framework, HMOA-GNN equipped with MOLS-GC achieves an optimal balance between recall (0.8333) and AUC (0.9445), confirming that MOLS-GC enhances model generalization and adaptability across heterogeneous transaction distributions.

In the Simulated dataset, MOLS-GC slightly improves accuracy while maintaining comparable AUC levels. This consistency indicates that MOLS-GC remains stable even when latent structures are less distinctive, demonstrating both robustness and low sensitivity to distributional variations. When combined with the full HMOA-GNN architecture, MOLS-GC further enhances the F1-score and overall discriminability.

Overall, across all datasets, the integration of MOLS-GC enhances the structural homogeneity of the graph, leading to more coherent latent representations and better-organized topologies. These improvements collectively strengthen the model's capacity to cluster homogeneous samples and discriminate anomalous transactions, thereby underscoring the pivotal role of MOLS-GC in achieving structurally optimized and semantically consistent graph representations for effective fraud detection.

Building upon these observations, we further investigate how MOLS-GC drives such structural improvements. In the context of tabular credit card transaction data, where no inherent graph structure exists, we propose MOLS-GC to construct more meaningful adjacency relations that enhance homophily. To quantify the degree of homophily in the generated graphs, we employ the Label Agreement Rate as the primary evaluation metric, which measures the consistency of node labels within k-hop neighborhoods from 1-hop to 4-hop. In these homophily-enhanced graph structures, higher-order neighborhoods more effectively preserve label consistency, thereby providing richer and more reliable structural signals that can be leveraged by AdaAdvSAGE for robust representation learning [48].

As illustrated in Fig. 5, our experimental findings substantiate this observation: as k increases, the graphs constructed by MOLS-GC achieve higher Label Agreement Rates compared with those generated by random edge connections. This demonstrates that our method induces stronger homophily patterns in higher-order structures, offering a more solid foundation for downstream graph neural network learning.

Ablation experiment of AdaAdvSAGE module

To verify the effectiveness of the AdaAdvSAGE module, we sampled subsets from three benchmark datasets. A fixed random seed (set to 42) was applied during the sampling process to ensure reproducibility. We first examined the performance improvements of baseline methods after incorporating the AdaAdvSAGE module, and further compared the overall performance of the HMOA-GNN framework with and without this module, thereby evaluating both its standalone effectiveness and its contribution to the framework across the three datasets.

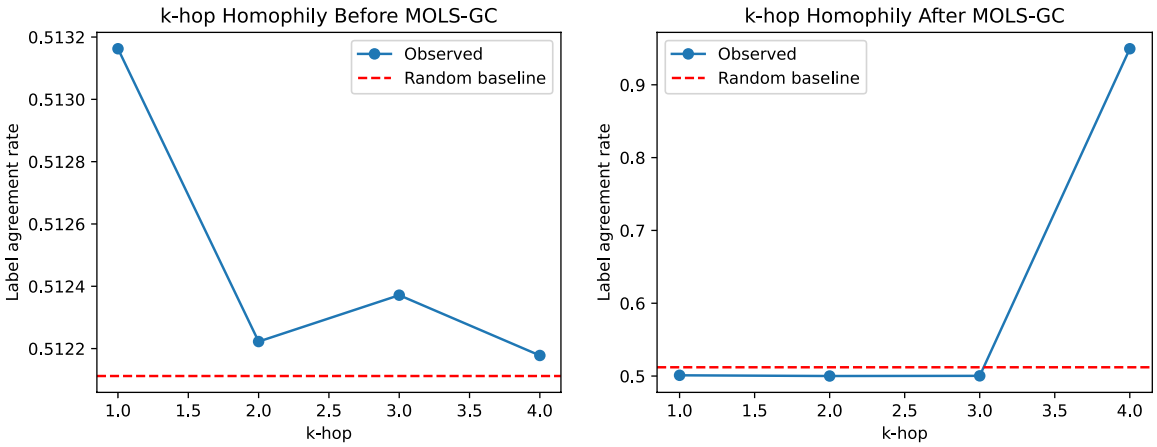
As summarized in Table 4, on the European dataset, both baseline models fail to identify any fraudulent transactions because of the extreme class imbalance, resulting in zero values for both Recall and F1-score. When AdaAdvSAGE is integrated into the complete HMOA-GNN framework, the model attains near-perfect detection capability, achieving an F1-score of 0.9307 and an AUC of 0.9975. In contrast, excluding AdaAdvSAGE yields a marginally higher Precision of 1.0000 but leads to lower Recall and AUC scores of 0.8824 and 0.9507, respectively. This outcome indicates that AdaAdvSAGE effectively maintains a balanced trade-off between precision and recall by enhancing feature robustness and alleviating overfitting tendencies toward the majority class.

For the IEEE-CIS dataset, the integration of the proposed AdaAdvSAGE module results in a clear and consistent performance improvement across all evaluation metrics. Compared with the baseline, the Baseline w/ AdaAdvSAGE variant achieves notable gains in Accuracy (+0.85%), Precision (+5.0%), Recall (+24.2%), F1-score (+0.19), and AUC (+0.0481). These substantial improvements indicate that the synergistic effect of adversarial training and adaptive feature selection effectively enhances the model's generalization capability and feature-level discriminability when dealing with complex, high-dimensional transactional data. In contrast, removing AdaAdvSAGE from the complete HMOA-GNN framework causes a dramatic deterioration in performance, underscoring the indispensable role of AdaAdvSAGE in stabilizing the learning process and mitigating overfitting under severe class imbalance conditions.

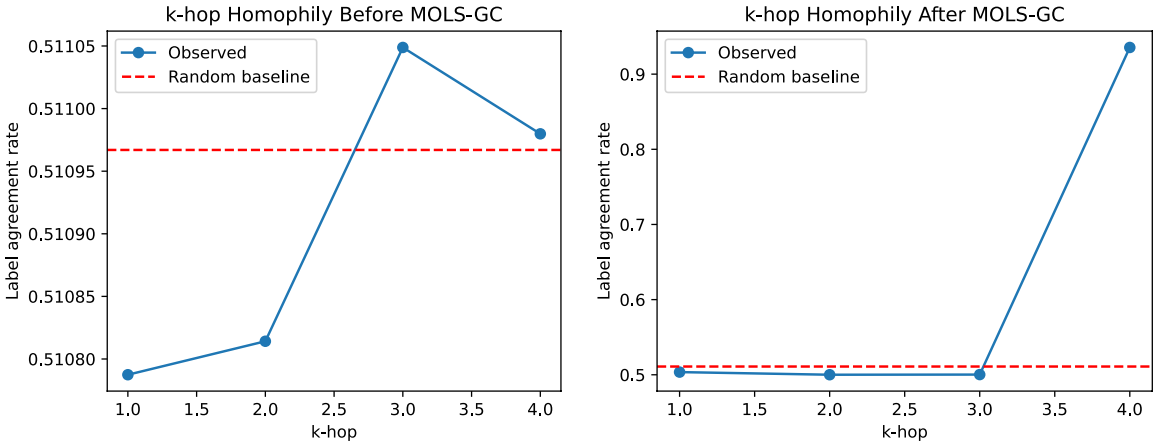
For the Simulated dataset, the incorporation of AdaAdvSAGE significantly enhances the detection capability of the baseline model, increasing the F1-score from 0.0000 to 0.6372. Within the HMOA-GNN framework, AdaAdvSAGE contributes to a more balanced relationship between precision and recall, with the former reaching 0.7049 compared to 0.7500 without the module, and the latter improving from 0.6774 to 0.6935. This balance indicates that the module stabilizes performance across different evaluation dimensions. A slight reduction in AUC, from 0.9177 to 0.8931, is observed after integration, which can be attributed to the adversarial perturbation mechanism. This mechanism deliberately introduces controlled noise during training to enhance model generalization rather than directly optimizing for AUC performance.

In summary, these ablation results substantiate that AdaAdvSAGE serves as a pivotal component in the HMOA-GNN framework, enhancing its discriminative power, resilience, and robustness. Through adversarial regularization and adaptive feature selection, AdaAdvSAGE enables the model to effectively resist noise and imbalance while maintaining strong feature expressiveness. Although minor metric fluctuations are observed in certain datasets, they reflect a deliberate trade-off between overfitting suppression and global generalization, which ultimately contributes to more reliable fraud detection performance in real-world transactional environments.

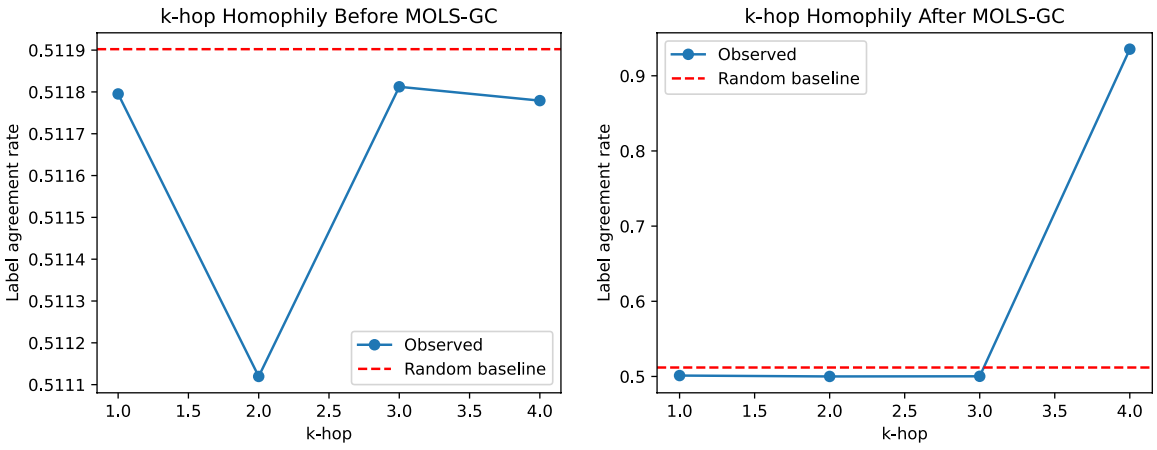
In addition to the quantitative evaluation presented above, we further conducted a qualitative analysis to visually examine the influence of the adaptive feature selection mechanism within the AdaAdvSAGE module. While the numerical results in Table 4 demonstrate its strong impact on enhancing the model's stability and



(a) Label Agreement Rates across k-hop Neighborhoods on European



(b) Label Agreement Rates across k-hop Neighborhoods on IEEE-CIS



(c) Label Agreement Rates across k-hop Neighborhoods on Simulated

Figure 5. Enhancement of k-hop homophily before and after MOLS-GC across three datasets.

generalization ability, the underlying effect on feature representation can be more intuitively understood through visualization. To this end, we employed t-SNE and UMAP to project the learned feature embeddings of transactions before and after applying adaptive feature selection. These visual comparisons aim to reveal how the proposed mechanism reshapes the latent space structure, enhances inter-class separability, and suppresses noise or redundant dimensions that obscure fraudulent behavior patterns.

Figures 6 and 7 depict the distribution of transactions in the latent space before and after applying the proposed adaptive feature selection method, visualized using t-SNE and UMAP. The results are presented on three benchmark datasets: (a) European Cardholders Transaction dataset, (b) IEEE-CIS Fraud Detection dataset:, and (c) Simulated Credit Card Transactions dataset. Prior to feature selection, the data points corresponding

Dataset	Algorithm	Evaluation Metrics				
		Accuracy	Precision	Recall	F1-score	AUC
European	Baseline	0.9745	0.0000	0.0000	0.0000	0.5000
	Baseline w/ AdaAdvSAGE	0.9745(–)	0.0000(–)	0.0000(–)	0.0000(–)	0.5000(–)
	HMOA-GNN	0.9965	0.9400	0.9216	0.9307	0.9975
	HMOA-GNN w/o AdaAdvSAGE	0.9965(–)	1.0000(↑)	0.8824(↓)	0.9278(↓)	0.9507(↓)
IEEE-CIS	Baseline	0.9815	0.8718	0.5152	0.6476	0.9099
	Baseline w/ AdaAdvSAGE	0.9900(↑)	0.9259(↑)	0.7576(↑)	0.8333(↑)	0.9580(↑)
	HMOA-GNN	0.9805	0.6627	0.8333	0.7383	0.9445
	HMOA-GNN w/o AdaAdvSAGE	0.1450(↓)	0.0361(↓)	1.0000(↑)	0.0696(↓)	0.5452(↓)
Simulated	Baseline	0.9685	0.0000	0.0000	0.0000	0.9386
	Baseline w/ AdaAdvSAGE	0.9795(↑)	0.7059(↑)	0.5806(↑)	0.6372(↑)	0.9032(↓)
	HMOA-GNN	0.9815	0.7049	0.6935	0.6992	0.9177
	HMOA-GNN w/o AdaAdvSAGE	0.9830(↑)	0.7500(↑)	0.6774(↓)	0.7119(↑)	0.8931(↓)

Table 4. Ablation results of AdaAdvSAGE module.

to fraudulent (red) and legitimate (green) transactions exhibit considerable overlap, rendering the two classes difficult to distinguish. After applying adaptive feature selection, the visualizations obtained through t-SNE and UMAP reveal markedly improved separation, with fraudulent transactions forming more compact and clearly distinguishable clusters. The consistent improvements observed across different visualization techniques and datasets demonstrate that the proposed method effectively identifies feature relevance and enhances the discriminative capacity of the latent representation, thereby substantiating its contribution to fraud detection.

Comparative experiment

Baseline methods

- *DAE* [49]: This study proposes a denoising autoencoder neural network (DAE) that integrates oversampling with noise reduction, leveraging misclassification costs to enhance minority class classification accuracy in imbalanced datasets.
- *DevNet* [50]: This study introduces a neural deviation learning framework for anomaly detection that directly optimizes anomaly scores using a few labeled anomalies and prior probability, achieving data-efficient training and superior anomaly scoring.
- *GA-RF*[51]: This study proposes a machine learning-based CCFD framework that utilizes a genetic algorithm for feature selection, integrated with an ensemble of diverse classifiers including Decision Tree, Random Forest, Logistic Regression, Artificial Neural Network, and Naive Bayes. By combining evolutionary feature optimization with multi-model learning, the framework enhances the robustness and adaptability of fraud detection across varying data characteristics.
- *SOBT*[27]: This study proposes a CCFD model that integrates a fraud feature-boosting mechanism with a spiral oversampling balancing technique (SOBT), employing a compound grouping elimination strategy to reduce feature redundancy and a multifactor synchronous embedding mechanism to enhance feature decision-making, enhances the model's capacity to represent and discriminate fraudulent patterns in imbalanced transaction data.
- *GTAN*[52]: This study proposes a gated temporal attention network (GTAN) for CCFD, constructing a temporal transaction graph and leveraging a Gated Temporal Attention Network for representation learning and risk propagation, enhances the model's ability to capture complex patterns of fraudulent behavior under limited supervision.
- *TFD*[53]: This study proposes a Transformer-based CCFD (TFD) model that applies data balancing and feature refinement, leveraging self-attention to capture long-range dependencies in tabular transactions, achieving superior detection performance over XGBoost and TabNet.

Comparative experiment of European Cardholders Transaction dataset

To investigate the impact of different sampling strategies on fraud detection performance, we sampled a subset of the European Cardholders Transaction dataset, ensuring that the original class imbalance ratio was preserved. A fixed random seed (set to 42) was used to guarantee reproducibility. Five sampling methods were then employed to address the class imbalance problem: DEHS, random undersampling, SMOTE, H-SMOTE and K-means-based undersampling. After sampling, we first applied our MOLS-GC method to construct graph structures that conform to the homophily assumption. These constructed graphs were then used as input to train the AdaAdvSAGE model on each processed dataset, thereby evaluating the impact of different sampling strategies on the overall framework performance. The experimental results are summarized in Table 5.

Table 5 presents the experimental results of applying different sampling algorithms to the European Cardholders Transaction dataset. Since fraud detection datasets are typically imbalanced, accuracy alone is insufficient to measure the actual effectiveness of the algorithm. Therefore, we added metrics such as recall, F1-score, and AUC, which are more suitable for imbalanced test data. The results demonstrate that after processing the training data

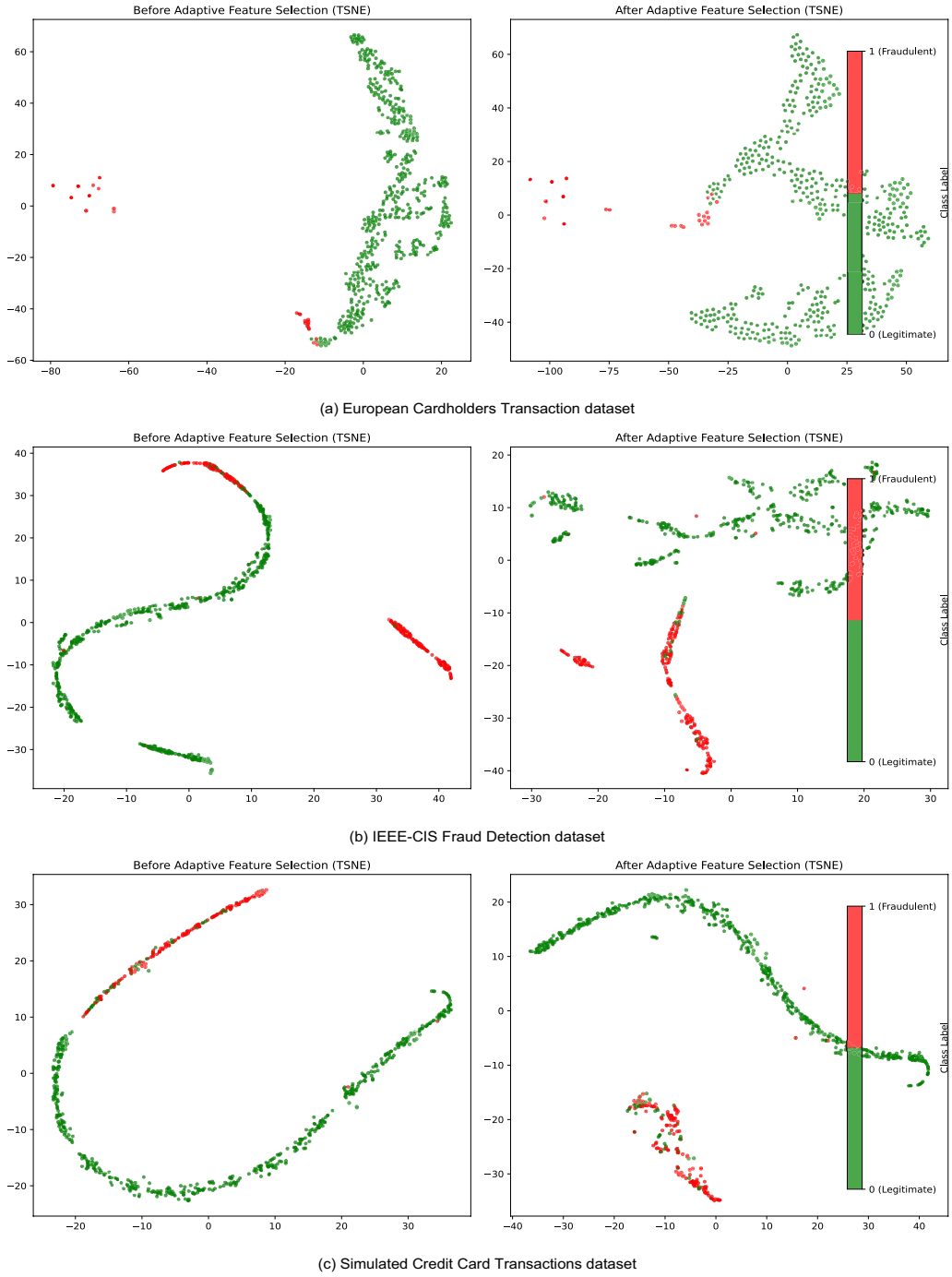


Fig. 6. Transaction distribution before and after Adaptive Feature Selection with t-SNE visualization.

using the DEHS method, the model achieved the best performance in terms of accuracy, precision, F1-score, and AUC, while maintaining competitive recall. Compared with random undersampling and K-means-based undersampling methods, DEHS yielded substantial improvements across all metrics. Compared with SMOTE and H-SMOTE, DEHS exhibited notably higher precision and F1-score, while achieving a recall comparable to SMOTE and only slightly lower than H-SMOTE. Although H-SMOTE reached the highest recall, it suffered from a noticeable drop in precision, indicating a higher false positive rate. In contrast, the model trained on data processed with K-means-based undersampling performed poorly on nearly all metrics, particularly with extremely low accuracy, precision, and F1-score, suggesting that the model classified an excessive number of transactions as fraudulent, thereby generating a very high false positive rate.

As mentioned in the previous section, F1-score considers both false positive and false negative rates, achieving a balance between the two. Therefore, using F1-score as an evaluation metric can avoid extreme situations. Figure 8 shows the F1-scores for different sampling methods in the European Cardholders Transaction dataset.

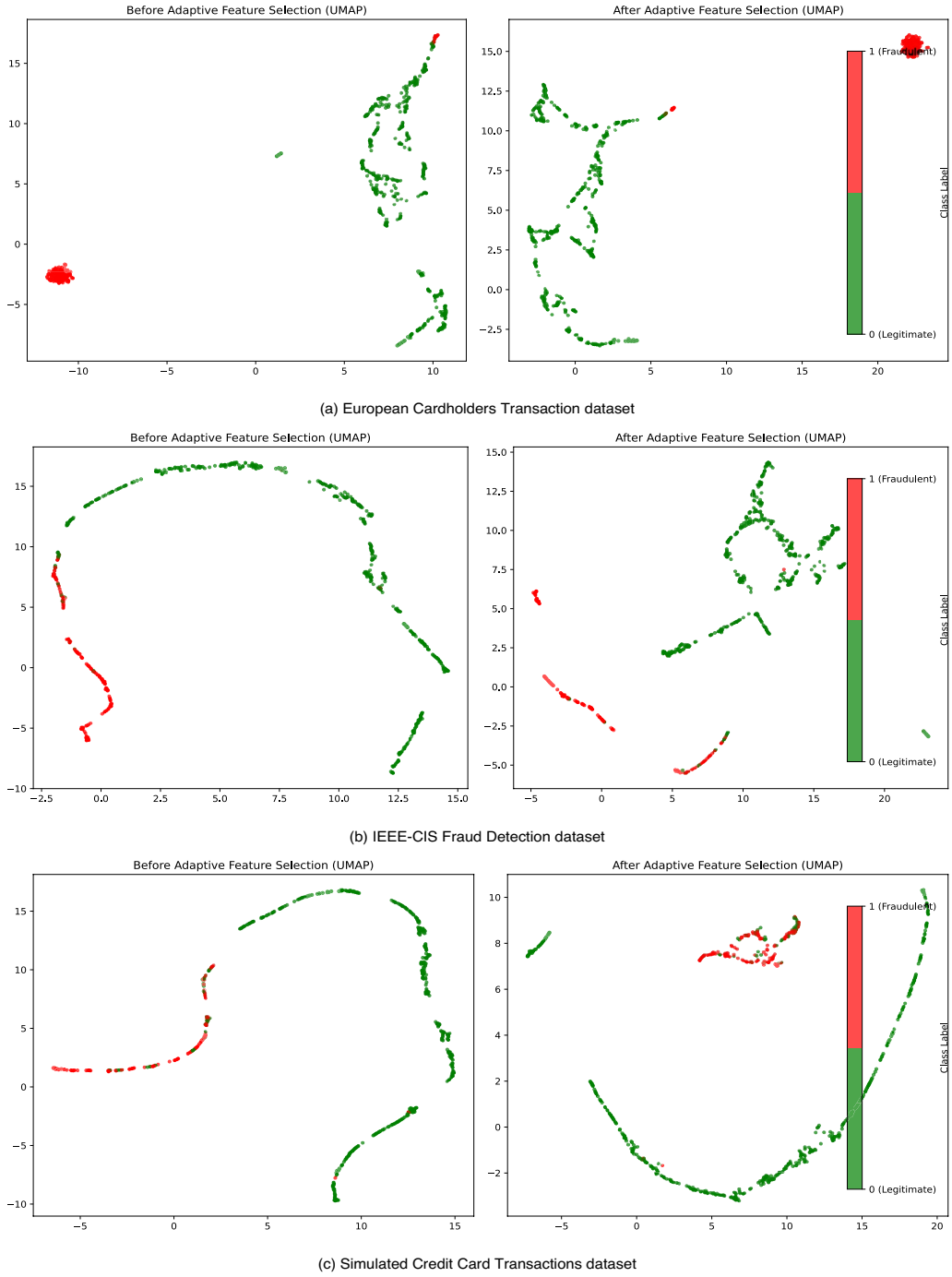


Fig. 7. Transaction distribution before and after Adaptive Feature Selection with UMAP visualization.

Metrics	DEHS(Ours)	Random undersampling	SMOTE	H-SMOTE	K-means-based undersampling
Accuracy	**0.9965**	0.9870	0.9950	0.9950	0.1810
Precision	**0.9400**	0.6866	0.8868	0.8596	0.0274
Recall	0.9216	0.9020	0.9216	**0.9608**	0.9020
F1-score	**0.9307**	0.7797	0.9038	0.9074	0.0532
AUC	**0.9975**	0.9557	0.9777	0.9891	0.5014

Table 5. Performance comparison of DEHS and other sampling methods on European Cardholders Transaction dataset. Bold values indicate the best performance.

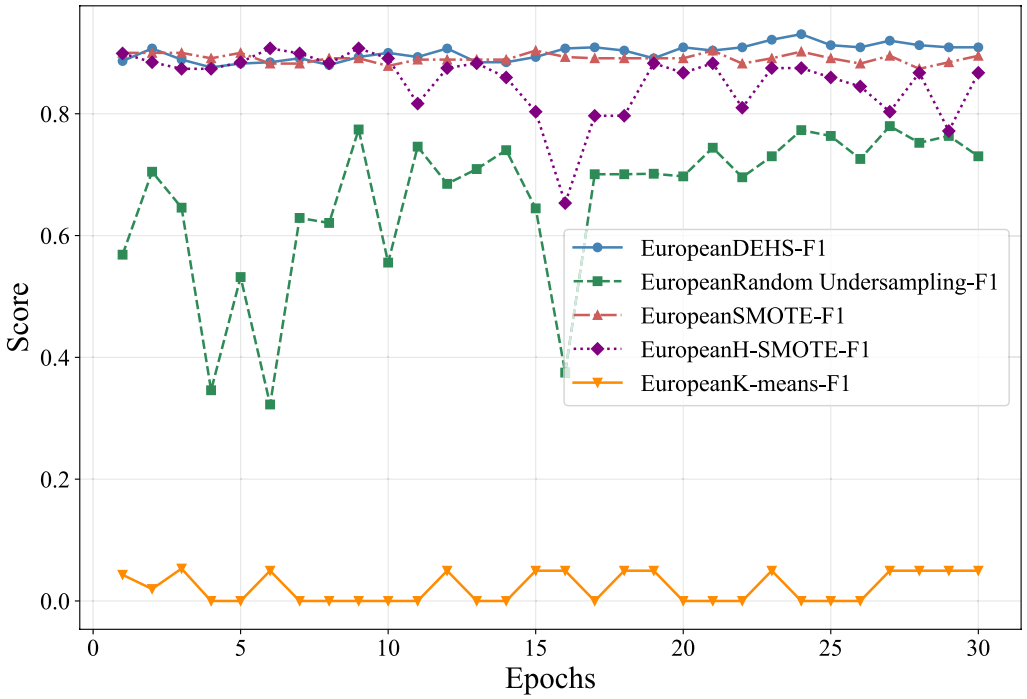


Fig. 8. F1-scores for different sampling methods on European Cardholders Transaction dataset.

Ref	Algorithm	Accuracy	Precision	Recall	F1-score	AUC
Zou et al.[49]	DAE	0.8285	0.1075	0.7843	0.1891	0.8808
Pang et al.[50]	DevNet	0.9710	0.4706	0.8431	0.6010	0.9014
Ileberi et al.[51]	GA-RF	**0.9995**	0.9369	0.7647	0.8421	0.9544
Ni et al.[27]	SOBT	**0.9995**	0.9153	0.8095	0.8592	0.9081
Xiang et al.[52]	GTAN	0.9655	0.6427	0.6290	0.6355	0.8169
Chang et al.[53]	TFD	0.9883	0.7033	0.8889	0.7853	0.9452
Ours	HMOA-GNN	0.9965	**0.9400**	**0.9216**	**0.9307**	**0.9975**

Table 6. Performance comparison of HMOA-GNN and existing CCFD methods on European Cardholders Transaction dataset. Bold values indicate the best performance.

From the figure, it is clearly evident that, based on the F1-score, our DEHS method significantly outperforms random undersampling and K-means-based undersampling methods, and is slightly better than the SMOTE and H-SMOTE sampling methods for the European Cardholders Transaction dataset.

As shown in Table 6, our proposed HMOA-GNN achieves a recall of 0.9216, which significantly outperforms all compared baselines. The most competitive baseline in terms of recall is TFD[53], achieving a score of 0.8889, followed by DevNet[50] with a recall of 0.8431. Compared to TFD, HMOA-GNN achieves an absolute improvement of 0.0327 and a relative increase of approximately 3.7% in recall. Compared to DevNet, the absolute gain is 0.0785, corresponding to a relative improvement of 9.3%. This notable enhancement indicates that HMOA-GNN is more effective in identifying fraudulent transactions, especially under imbalanced data conditions where recall is critical. In addition to recall, HMOA-GNN also achieves the highest F1-score (0.9307) and AUC (0.9975), further demonstrating its superiority in balancing precision and recall.

Comparative experiment of IEEE-CIS Fraud Detection dataset

To evaluate the effectiveness of various sampling strategies in CCFD, we sampled a subset of the IEEE-CIS Fraud Detection dataset and applied standard preprocessing, ensuring that the original class imbalance ratio was preserved. A fixed random seed (set to 42) was used to guarantee reproducibility. We then applied five sampling techniques individually, namely DEHS, random undersampling, SMOTE, H-SMOTE and K-means-based undersampling, to address the class imbalance issue. After sampling, we first applied our MOLS-GC method to construct graph structures that conform to the homophily assumption. These constructed graphs were then used as input to train the AdaAdvSAGE model on each processed dataset, thereby evaluating the impact of different sampling strategies on the overall framework performance. The experimental results are summarized in Table 7.

Metrics	DEHS(Ours)	Random undersampling	SMOTE	H-SMOTE	K-means-based undersampling
Accuracy	**0.9805**	0.8900	0.9645	0.9675	0.9065
Precision	**0.6627**	0.2270	0.4793	0.5043	0.2590
Recall	0.8333	0.9697	0.8788	0.8939	**0.9848**
F1-score	**0.7383**	0.3678	0.6203	0.6448	0.4101
AUC	0.9445	0.9528	**0.9645**	0.9285	0.9529

Table 7. Performance comparison of DEHS and other sampling methods on IEEE-CIS Fraud Detection dataset. Bold values indicate the best performance.

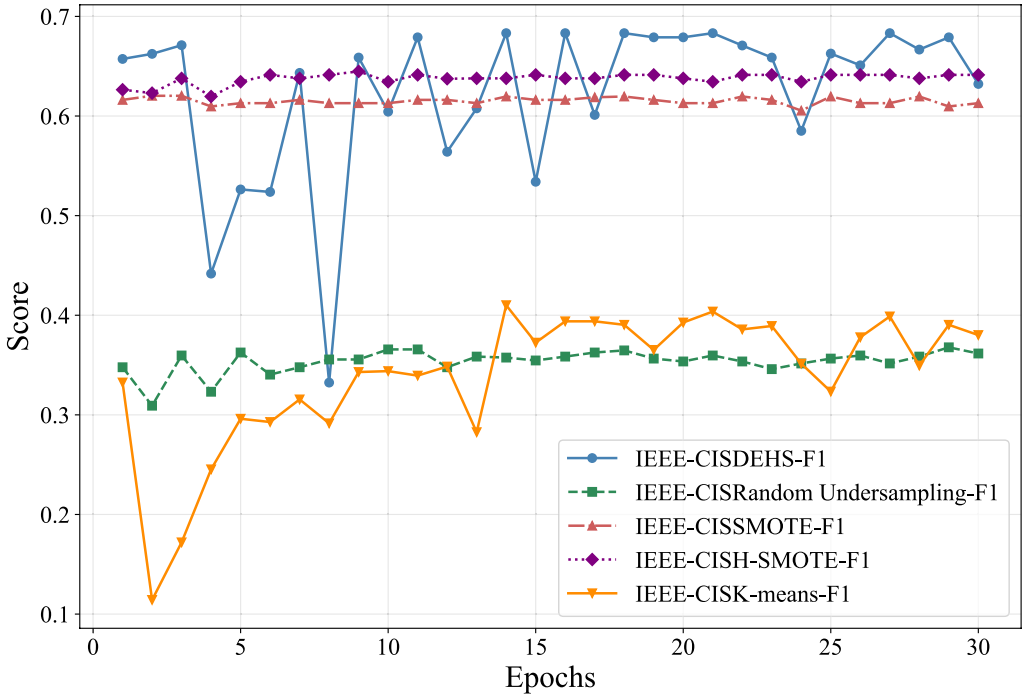


Fig. 9. F1-scores for different sampling methods on IEEE-CIS Fraud Detection dataset.

Table 7 lists the experimental results of applying different sampling algorithms to the IEEE-CIS Fraud Detection dataset. The results show that after processing the training data with the DEHS method, it outperforms existing methods such as random undersampling, SMOTE, H-SMOTE and K-means-based undersampling under various evaluation metrics. In most metrics, the model trained on data processed with the DEHS method performed better than the models trained using random undersampling and K-means-based undersampling methods. Although the models trained on data processed with random undersampling and K-means-based undersampling methods had recall rates of 0.9697 and 0.9848, higher than that of DEHS, their scores in other metrics like precision and F1-score were lower. In particular, their precision scores were only 0.2270 and 0.2590, indicating a very high false positive rate. The model trained on data processed with the SMOTE method had a slightly higher AUC than the DEHS method, but its precision and F1-score still did not reach the level of DEHS. The model trained on data processed with the H-SMOTE method showed improvements over SMOTE across multiple metrics; however, its lower AUC indicates that these gains in local performance came at the expense of slightly reduced global discriminative ability.

It is important to emphasize that accuracy and recall alone are insufficient to fully reflect the detection capability of a model on imbalanced datasets. The excellent performance of the DEHS method in important metrics such as precision, F1-score, and AUC demonstrates its advantage in balancing false positives and false negatives, proving its superiority in fraud detection tasks.

Figure 9 shows the F1-scores under different sampling methods in the IEEE-CIS Fraud Detection dataset. It is evident from the figure that, based on the F1-score, our DEHS method significantly outperforms random undersampling and K-means-based undersampling methods in the IEEE-CIS Fraud Detection dataset, and achieves better model performance than the SMOTE and H-SMOTE sampling methods.

As shown in Table 8, our proposed HMOA-GNN achieves a recall of 0.8333, which significantly outperforms all compared baselines. The most competitive baseline in terms of recall is TFD [53], achieving a score of 0.8300, followed by GTAN [52] with a recall of 0.7172. Compared to TFD, HMOA-GNN achieves an absolute improvement of 0.0033 and a relative increase of approximately 0.4% in recall. Compared to GTAN, the absolute gain is 0.1161,

Ref	Algorithm	Accuracy	Precision	Recall	F1-score	AUC
Zou et al.[49]	DAE	0.9030	0.2091	0.6970	0.3217	0.8957
Pang et al.[50]	DevNet	0.7398	0.0471	0.5976	0.0873	0.6739
Ileberi et al.[51]	GA-RF	**0.9872**	**0.9863**	0.5620	0.7160	**0.9613**
Ni et al.[27]	SOBT	0.9787	0.8819	0.4442	0.5908	0.7223
Xiang et al.[52]	GTAN	0.9700	0.7019	0.7172	0.7092	0.8762
Chang et al.[53]	TFD	0.9027	0.2318	0.8300	0.3624	0.9409
Ours	HMOA-GNN	0.9805	0.6627	**0.8333**	**0.7383**	0.9445

Table 8. Performance comparison of HMOA-GNN and existing CCFD methods on IEEE-CIS Fraud Detection dataset. Bold values indicate the best performance.

Metrics	DEHS(Ours)	Random undersampling	SMOTE	H-SMOTE	K-means-based undersampling
Accuracy	**0.9815**	0.8520	0.9695	0.9665	0.8715
Precision	**0.7049**	0.1638	0.5052	0.4752	0.1761
Recall	0.6935	**0.9194**	0.7903	0.7742	0.8548
F1-score	**0.6992**	0.2780	0.6164	0.5890	0.2920
AUC	**0.9177**	0.8758	0.9045	0.8845	0.8648

Table 9. Performance comparison of DEHS and other sampling methods on Simulated Credit Card Transactions dataset. Bold values indicate the best performance.

corresponding to a relative improvement of 16.2%. Furthermore, HMOA-GNN attains the highest F1-score (0.7383) among all evaluated methods, highlighting its superior capability in minimizing false negatives while preserving a well-balanced trade-off between precision and recall. This characteristic is particularly desirable in real-world CCFD applications, where both sensitivity and specificity are critical. Additionally, its AUC of 0.9445 is competitive with top-performing models, further demonstrating the effectiveness and robustness of our approach.

Comparative experiment of Simulated Credit Card Transactions dataset

To investigate the impact of different sampling strategies on fraud detection performance, we extracted a subset of the Simulated Credit Card Transactions dataset while preserving the original class imbalance ratio. A fixed random seed (set to 42) was used to ensure reproducibility. Five sampling methods were employed to address the class imbalance problem, namely DEHS, random undersampling, SMOTE, H-SMOTE, and K-means-based undersampling. After sampling, our MOLS-GC method was applied to construct graph structures consistent with the homophily assumption. These constructed graphs were subsequently used as inputs to train the AdaAdvSAGE model on each processed dataset, thereby enabling an evaluation of how different sampling strategies influence the overall framework performance. A summary of the experimental results is provided in Table 9.

As shown in Table 9, DEHS achieves the highest accuracy (0.9815), precision (0.7049), F1-score (0.6992), and AUC (0.9177), while its recall (0.6935) is lower than that of the other sampling strategies. Random undersampling produces the highest recall (0.9194), but with substantially reduced precision (0.1638). SMOTE and H-SMOTE yield relatively higher recall values (0.7903 and 0.7742, respectively), although the generation of synthetic samples around class boundaries likely introduces noise that decreases precision. K-means-based undersampling achieves a recall of 0.8548, yet precision remains low (0.1761), which may be attributed to the cluster-based reduction of majority samples that biases the classifier toward minority predictions. In contrast, the results of DEHS indicate that its sampling process retains more informative structural patterns while introducing less noise compared to SMOTE and its variants, which is beneficial for constructing graphs under the homophily assumption and for enabling the AdaAdvSAGE model to learn more discriminative representations.

Although H-SMOTE was originally designed to overcome the limitations of SMOTE by addressing intra-class imbalance and small disjuncts through histogram-based binning, its performance on the Simulated Credit Card Transactions dataset is observed to be inferior to that of SMOTE. This may be attributed to its reliance on histogram binning, which can generate synthetic samples that are not fully consistent with the underlying data distribution, as well as its lack of a noise filtering mechanism. Moreover, the amplification of small disjuncts may introduce additional noise in fraud detection tasks, thereby reducing overall effectiveness compared to the simpler interpolation strategy of SMOTE.

Figure 10 shows the F1-scores under different sampling methods in the Simulated Credit Card Transactions dataset. The results indicate that our proposed DEHS method yields substantially better performance than random undersampling and K-means-based undersampling, and exhibits a distinct advantage over the SMOTE and H-SMOTE sampling methods.

As presented in Table 10, our proposed HMOA-GNN achieves the highest F1-score (0.6992) on the Simulated Credit Card Transactions dataset, surpassing all compared baselines. In terms of accuracy, HMOA-GNN

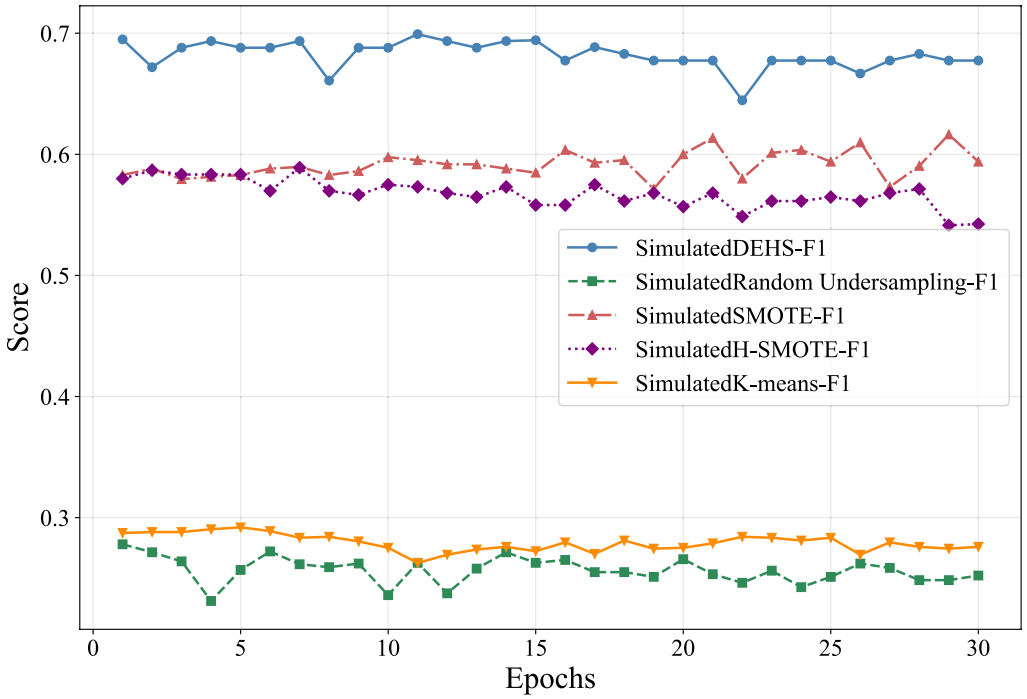


Fig. 10. F1-scores for different sampling methods on Simulated Credit Card Transactions dataset.

Ref	Algorithm	Accuracy	Precision	Recall	F1-score	AUC
Zou et al.[49]	DAE	0.8790	0.1809	0.8226	0.2965	0.9293
Pang et al.[50]	DevNet	0.8116	0.1339	**0.9274**	0.2340	0.9210
Ileberi et al.[51]	GA-RF	**0.9815**	0.7551	0.5968	0.6667	0.9504
Ni et al.[27]	SOBT	0.9800	**0.9348**	0.4300	0.5890	**0.9760**
Xiang et al.[52]	GTAN	0.8838	0.6581	0.4063	0.5535	0.8379
Chang et al.[53]	TFD	0.9807	0.8889	0.4800	0.6234	0.9672
Ours	HMOA-GNN	**0.9815**	0.7049	0.6935	**0.6992**	0.9177

Table 10. Performance comparison of HMOA-GNN and existing CCFD methods on Simulated Credit Card Transactions dataset. Bold values indicate the best performance.

and GA-RF both achieve the best result (0.9815), slightly outperforming TFD (0.9807), which demonstrates the stability of HMOA-GNN in overall predictive correctness. Regarding recall, DevNet reports the highest value (0.9274), albeit at the cost of extremely low precision (0.1339), resulting in a poor F1-score (0.2340). By contrast, HMOA-GNN achieves a recall of 0.6935, which, although lower than DevNet, strikes a more favorable balance between sensitivity and specificity. More notably, compared with GA-RF (0.5968), HMOA-GNN attains an absolute gain of 0.0967, corresponding to a relative improvement of 16.2%, underscoring its superior coverage of fraudulent transactions.In terms of F1-score, HMOA-GNN achieves the best performance (0.6992), outperforming GA-RF (0.6667) with an absolute increase of 0.0325 (4.9% relative) and TFD (0.6234) with an absolute gain of 0.0758 (12.2% relative). This highlights its strong capability to capture fraudulent cases while mitigating false alarms. For precision, SOBT achieves the highest score (0.9348), followed by TFD (0.8889), whereas HMOA-GNN (0.7049) maintains a more balanced trade-off with higher recall and F1-score. For AUC, SOBT and TFD remain competitive with scores of 0.9760 and 0.9672, respectively, while HMOA-GNN achieves 0.9177, still demonstrating robust discriminative ability.

Taken together, the experiments across the European Cardholders Transaction, IEEE-CIS Fraud Detection, and Simulated Credit Card Transactions datasets consistently demonstrate that the primary strength of HMOA-GNN lies in its well-balanced overall performance. In particular, its superior F1-score highlights the model's ability to simultaneously control false positives and false negatives, thereby ensuring robust detection in highly imbalanced real-world scenarios. While certain baselines, such as DevNet, achieve higher recall at the cost of markedly reduced precision, and others, such as SOBT or GA-RF, emphasize precision while sacrificing recall or generalization, HMOA-GNN maintains a stable trade-off across accuracy, precision, recall, F1-score, and AUC. This stability stems from the synergistic design of its three core components: the DEHS module, which effectively balances imbalanced data by generating informative minority samples while filtering noise; the

MOLS-GC module, which constructs homophily-consistent graphs tailored to the transaction domain; and the AdaAdvSAGE model, which integrates adversarial training with adaptive feature aggregation to enhance representation learning. The superiority of HMOA-GNN across multiple datasets and evaluation metrics underscores its robustness, adaptability, and practical reliability compared with models that excel only in isolated metrics but lack holistic effectiveness. These results confirm that HMOA-GNN offers a comprehensive and generalizable solution for real-world CCFD.

Conclusion

We propose HMOA-GNN, a multi-strategy enhanced adaptive adversarial GNN framework with hierarchical hybrid sampling and metric optimization for CCFD. It improves data quality via center selection and boundary refinement, and addresses class imbalance through density-aware sampling and noise filtering. A metric learning-based latent mapper and KNN-driven graph construction ensure homophily in transaction graphs. Additionally, AdaAdvSAGE adaptively emphasizes informative features during GNN message passing and aggregation. Extensive experiments validate that HMOA-GNN achieves superior performance and robust generalization across diverse metrics and scenarios.

In future work, we aim to further optimize the computational efficiency of the proposed method to enhance its applicability in streaming and real-time transaction scenarios. Meanwhile, we will improve the interpretability of fraud detection outcomes to promote transparency and trustworthiness. Moreover, we plan to investigate heterogeneous and temporal graph modeling to better capture complex entity relationships and the dynamic nature of fraud patterns in credit card transactions.

Data availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Received: 2 September 2025; Accepted: 31 October 2025
Published online: 02 December 2025

References

1. J, D. Credit card fraud. https://en.wikipedia.org/wiki/Credit_card_fraud (2025).
2. Liu, G. *Petri Nets: Theoretical Models and Analysis Methods for Concurrent Systems* (Springer, 2022).
3. Vorobyev, I. & Krivitskaya, A. Reducing false positives in bank anti-fraud systems based on rule induction in distributed tree-based models. *Comput. Secur.* **120**, 102786. https://doi.org/10.1016/j.cose.2022.102786 (2022).
4. Gianini, G. et al. Managing a pool of rules for credit card fraud detection by a game theory based approach. *Futur. Gener. Comput. Syst.* **102**, 549–561. https://doi.org/10.1016/j.future.2019.08.028 (2020).
5. Xie, Y. et al. Learning transactional behavioral representations for credit card fraud detection. *IEEE Trans. Neural Netw. Learn. Syst.* **35**, 5735–5748. https://doi.org/10.1109/TNNLS.2022.3208967 (2024).
6. Cherif, A. et al. Credit card fraud detection in the era of disruptive technologies: A systematic review. *J. King Saud Univ. Comput. Inf. Sci.* **35**, 145–174. https://doi.org/10.1016/j.jksuci.2022.11.008 (2023).
7. Akouhar, M., Ouhssini, M., El Fatini, M., Abarda, A. & Agherrabi, E. Dynamic oversampling-driven Kolmogorov–Arnold networks for credit card fraud detection: An ensemble approach to robust financial security. *Egypt. Inf. J.* **31**, 100712. https://doi.org/10.1016/j.eij.2025.100712 (2025).
8. Zhang, J., Man, X., Zhao, H. & Ni, L. Counterfactual graph convolution with quantified discrepancies for fraud detection. *Expert Syst. Appl.* **297**, 129281. https://doi.org/10.1016/j.eswa.2025.129281 (2026).
9. Li, Z., Huang, M., Liu, G. & Jiang, C. A hybrid method with dynamic weighted entropy for handling the problem of class imbalance with overlap in credit card fraud detection. *Expert Syst. Appl.* **175**, 114750. https://doi.org/10.1016/j.eswa.2021.114750 (2021).
10. Xie, Y. *et al.* A feature extraction method for credit card fraud detection. In *2019 2nd International Conference on Intelligent Autonomous Systems (ICoIAS)* 70–75. https://doi.org/10.1109/ICoIAS.2019.00019 (2019).
11. Lebichot, B. et al. Assessment of catastrophic forgetting in continual credit card fraud detection. *Expert Syst. Appl.* **249**, 123445. https://doi.org/10.1016/j.eswa.2024.123445 (2024).
12. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357. https://doi.org/10.1613/jair.953 (2002).
13. Maldonado, S., Vairetti, C., Fernandez, A. & Herrera, F. Fw-smote: A feature-weighted oversampling approach for imbalanced classification. *Pattern Recogn.* **124**, 108511. https://doi.org/10.1016/j.patcog.2021.108511 (2022).
14. Tarawneh, A. S., Hassanat, A. B., Altarawneh, G. A. & Almuhaimeed, A. Stop oversampling for class imbalance learning: A review. *IEEE Access* **10**, 47643–47660. https://doi.org/10.1109/ACCESS.2022.3169512 (2022).
15. Kumar, A., Singh, D. & Yadav, R. S. Entropy and improved k-nearest neighbor search based under-sampling (enu) method to handle class overlap in imbalanced datasets. *Concurr. Comput. Pract. Exp.* **36**, e7894. https://doi.org/10.1002/cpe.7894 (2024).
16. Sun, Z., Ying, W., Zhang, W. & Gong, S. Undersampling method based on minority class density for imbalanced data. *Expert Syst. Appl.* **249**, 123328. https://doi.org/10.1016/j.eswa.2024.123328 (2024).
17. Buda, M., Maki, A. & Mazurowski, M. A. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Netw.* **106**, 249–259. https://doi.org/10.1016/j.neunet.2018.07.011 (2018).
18. Xu, Y., Wang, J., Guang, M., Yan, C. & Jiang, C. Multistructure graph classification method with attention-based pooling. *IEEE Trans. Comput. Soc. Syst.* **10**, 602–613. https://doi.org/10.1109/TCSS.2022.3169219 (2023).
19. Guang, M., Yan, C., Xu, Y., Wang, J. & Jiang, C. Graph convolutional networks with adaptive neighborhood awareness. *IEEE Trans. Pattern Anal. Mach. Intell.* **46**, 7392–7404. https://doi.org/10.1109/TPAMI.2024.3391356 (2024).
20. Xu, Y., Wang, J., Guang, M. & Jiang, C. Graph multi-convolution and attention pooling for graph classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **46**, 10546–10557. https://doi.org/10.1109/TPAMI.2024.3443253 (2024).
21. Zhou, J. et al. Graph neural networks: A review of methods and applications. *AI Open* **1**, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001 (2020).
22. Wu, Z. et al. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **32**, 4–24. https://doi.org/10.1109/TNNLS.2020.2978386 (2021).
23. Zhu, J. *et al.* Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems* (eds. Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. & Lin, H.), vol. 33, 7793–7804 (Curran Associates, Inc., Red Hook, 2020).

24. Defferrard, M., Bresson, X. & Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems* (eds. Lee, D., Sugiyama, M., Luxburg, U., Guyon, I. & Garnett, R.), vol. 29 (Curran Associates, Inc., Red Hook, 2016).
25. Hamilton, W., Ying, Z. & Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems* (eds. Guyon, I. *et al.*, vol. 30 (Curran Associates, Inc., Red Hook, 2017).
26. Zhu, H. et al. Nus: Noisy-sample-removed undersampling scheme for imbalanced classification and application to credit card fraud detection. *IEEE Trans. Comput. Soc. Syst.* **11**, 1793–1804. https://doi.org/10.1109/TCSS.2023.3243925 (2024).
27. Ni, L., Li, J., Xu, H., Wang, X. & Zhang, J. Fraud feature boosting mechanism and spiral oversampling balancing technique for credit card fraud detection. *IEEE Trans. Comput. Soc. Syst.* **11**, 1615–1630. https://doi.org/10.1109/TCSS.2023.3242149 (2024).
28. Li, T., Wang, Y., Liu, L., Chen, L. & Chen, C. P. Subspace-based minority oversampling for imbalance classification. *Inf. Sci.* **621**, 371–388. https://doi.org/10.1016/j.ins.2022.11.108 (2023).
29. Lin, C., Tsai, C.-F. & Lin, W.-C. Towards hybrid over-and under-sampling combination methods for class imbalanced datasets: An experimental study. *Artif. Intell. Rev.* **56**, 845–863. https://doi.org/10.1007/s10462-022-10186-5 (2023).
30. Guo, J., Wu, H., Chen, X. & Lin, W. Adaptive sv-borderline smote-svm algorithm for imbalanced data classification. *Appl. Soft Comput.* **150**, 110986. https://doi.org/10.1016/j.asoc.2023.110986 (2024).
31. Alamri, M. & Ykhlef, M. Hybrid undersampling and oversampling for handling imbalanced credit card data. *IEEE Access* **12**, 14050–14060. https://doi.org/10.1109/ACCESS.2024.3357091 (2024).
32. Cheng, D., Zou, Y., Xiang, S. & Jiang, C. Graph neural networks for financial fraud detection: A review. *Front. Comp. Sci.* **19**, 199609. https://doi.org/10.1007/s11704-024-40474-y (2025).
33. Wang, X. & Zhang, M. How powerful are spectral graph neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, vol. 162 of *Proceedings of Machine Learning Research* (eds. Chaudhuri, K. *et al.*, 23341–23362 (PMLR, Brooklyn, 2022).
34. Li, Z., Yang, X. & Jiang, C. Multi-view graph-based hierarchical representation learning for money laundering group detection. *IEEE Trans. Inf. Forensics Secur.* **20**, 2035–2050. https://doi.org/10.1109/TIFS.2025.3529321 (2025).
35. Yang, Y. et al. Fmvpci: A multiview fusion neural network for identifying protein complex via fuzzy clustering. *IEEE Trans. Syst. Man Cybern. Syst.* **55**, 6189–6202. https://doi.org/10.1109/TSMC.2025.3578348 (2025).
36. Su, X. et al. Interpretable identification of cancer genes across biological networks via transformer-powered graph representation learning. *Nat. Biomed. Eng.* **9**, 371–389. https://doi.org/10.1038/s41551-024-01312-5 (2025).
37. Yang, Y. et al. Link-based attributed graph clustering via approximate generative bayesian learning. *IEEE Trans. Syst. Man Cybern. Syst.* **55**, 5730–5743. https://doi.org/10.1109/TSMC.2025.3572738 (2025).
38. Liu, J., Cheng, D. & Jiang, C. Preferential selective-aware graph neural network for preventing attacks in interbank credit rating. *IEEE Trans. Neural Netw. Learn. Syst.* **36**, 11414–11427. https://doi.org/10.1109/TNNLS.2024.3519169 (2025).
39. Qiao, L., Zhang, L., Chen, S. & Shen, D. Data-driven graph construction and graph learning: A review. *Neurocomputing* **312**, 336–351. https://doi.org/10.1016/j.neucom.2018.05.084 (2018).
40. Carneiro, M. G. & Zhao, L. Analysis of graph construction methods in supervised data classification. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)* 390–395. https://doi.org/10.1109/BRACIS.2018.00074 (2018).
41. Jiang, B., Zhang, Z., Lin, D., Tang, J. & Luo, B. Semi-supervised classification with graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 11305–11312 (2016).
42. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 770–778. https://doi.org/10.1109/CVPR.2016.90 (2016).
43. Kaggle. Credit card fraud detection. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud (2018).
44. Kaggle. Ieee-cis fraud detection. https://www.kaggle.com/code/pavan1512/ieee-cis-fraud-detection/data (2022).
45. Kaggle. Simulated credit card transactions. https://www.kaggle.com/datasets/kartik2112/fraud-detection/data (2020).
46. Liaw, L. C. M., Tan, S. C., Goh, P. Y. & Lim, C. P. A histogram smote-based sampling algorithm with incremental learning for imbalanced data classification. *Inf. Sci.* **686**, 121193. https://doi.org/10.1016/j.ins.2024.121193 (2025).
47. MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, vol. 5, 281–298 (University of California Press, 1967).
48. Zhu, J. *et al.* Beyond homophily in graph neural networks: Current limitations and effective designs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20 (Curran Associates Inc., Red Hook, 2020).
49. Zou, J., Zhang, J. & Jiang, P. Credit card fraud detection using autoencoder neural network (2019). arXiv:1908.11553.
50. Pang, G., Shen, C. & van den Hengel, A. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 353–362 (Association for Computing Machinery, New York, 2019). https://doi.org/10.1145/3292500.3330871
51. Ileberi, E., Sun, Y. & Wang, Z. A machine learning based credit card fraud detection using the ga algorithm for feature selection. *J. Big Data* **9**, 24. https://doi.org/10.1186/s40537-022-00573-8 (2022).
52. Xiang, S. et al. Semi-supervised credit card fraud detection via attribute-driven graph representation. *Proc. AAAI Conf. Artif. Intell.* **37**, 14557–14565. https://doi.org/10.1609/aaai.v37i12.26702 (2023).
53. Yu, C. *et al.* Credit card fraud detection using advanced transformer model. In *2024 IEEE International Conference on Metaverse Computing, Networking, and Applications (MetaCom)* 343–350, https://doi.org/10.1109/MetaCom62920.2024.00064 (2024).

Author contributions

All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by L.N., X.L., and Y.Z.. J.Z., as the corresponding author, contributed to the methodology development and provided critical guidance throughout the research process. H.Q. and X.M. assisted with data analysis and interpretation. The first draft of the manuscript was written by X.L., and all authors commented on and revised previous versions of the manuscript. All authors read and approved the final manuscript.

Funding

This work was supported in part by the National Science Foundation of Shandong Province, China under Grant ZR2023LZH018 and ZR2022MF338, Open Project of Tongji University Embedded System and Service Computing of Ministry of Education of China under Grant ESSCKF2022-02.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to J.Z.