

Hybrid IF–GNN–IF Framework for Suspected Fraud Detection in Bank Switching Transactions

Zumam Muddin¹, Bambang Jokonowo², Afiyati³

^{1,2,3}Data Science Department, Faculty of Computer Science, Mercu Buana University, Jakarta, Indonesia

Received:

June 30, 2026

Revised:

July 21, 2026

Accepted:

September 8, 2026

Published:

October 24, 2026

Corresponding Author:

Author Name*:

Zumam Muddin

Email*:

56225110021@student.mercu
 buana.ac.id

DOI:

10.63158/journalisi.v8i5.1799

© 2026 Journal of
 Information Systems and
 Informatics. This open
 access article is distributed
 under a (CC-BY License)



Abstract. Fraud detection in financial transactions remains a major challenge in the banking industry, particularly in bank switching systems that process high transaction volumes and involve complex interactions among transaction entities. This study proposes a hybrid Isolation Forest–Graph Neural Network–Isolation Forest (IF–GNN–IF) framework to identify suspected fraudulent transactions in bank switching data. In the first stage, Isolation Forest detects potentially anomalous transactions and produces anomaly scores that are used to construct pseudo-labels. In the second stage, a Graph Neural Network learns relational patterns from a transaction graph by integrating original transaction attributes with the pseudo-labels. In the final stage, the learned graph embeddings are combined with the original transaction features and evaluated using a second Isolation Forest to generate final anomaly scores. The framework was evaluated using 71,873 bank switching transaction records and achieved an accuracy of 0.9668, precision of 0.9200, recall of 0.3681, and an F1-score of 0.5258. The high precision indicates that the framework effectively identifies suspicious transaction patterns. However, the relatively low recall suggests that some suspected fraud cases remain undetected. Therefore, further validation using confirmed fraud labels is necessary to assess the practical effectiveness and generalizability of the proposed framework in real-world banking environments.

Keywords: Fraud detection; bank switching; Isolation Forest; Graph Neural Network; anomaly detection.

1. INTRODUCTION

The rapid growth of digital banking and electronic payment services has increased the volume and complexity of transactions processed through bank switching systems. This growth makes it increasingly difficult to identify suspicious transaction patterns, particularly when anomalies arise from interactions among accounts, cards, merchants, issuers, and terminals. Conventional transaction-level monitoring may overlook transactions that appear normal individually but form abnormal patterns collectively. Therefore, an analytical approach capable of identifying both transaction-level anomalies and relational patterns is needed to strengthen suspected-fraud monitoring and prevention in bank switching environments [1], [2].

Isolation Forest (IF) is widely used for anomaly detection because it can identify abnormal observations without requiring confirmed fraud labels, making it suitable for financial transaction datasets where reliable fraud labels are limited [1], [5]. However, IF primarily evaluates individual transactions based on feature distributions and does not explicitly capture relationships among entities. As a result, transactions that appear normal in isolation but collectively form abnormal interaction patterns may remain undetected, potentially increasing false negatives and associated financial, investigative, regulatory, and customer-trust risks [1], [3]. In contrast, Graph Neural Networks (GNNs), particularly Graph Convolutional Networks (GCNs), can model relationships among entities in transaction networks and learn representations that capture network-based behavioral patterns [4], [5].

Although anomaly detection and graph-based learning have been widely investigated for fraud-related applications, most existing approaches apply these techniques independently or combine them only in limited ways. Recent studies have also demonstrated the applicability of machine learning and GNNs to financial transaction data; however, many approaches rely on supervised learning with confirmed fraud labels or focus on single-stage detection [6], [7], [8], [9]. Consequently, there remains limited investigation of a sequential IF-GNN-IF framework, in which the first IF generates anomaly-based pseudo-labels, the GNN incorporates relational information into transaction representations, and the second IF performs anomaly detection on the resulting graph-enhanced representations. This sequential approach is designed to

address both the limited availability of confirmed fraud labels and the inability of transaction-level anomaly detection to capture relational patterns.

The output of the proposed framework is intended to support the prioritization of transaction alerts for further investigation, rather than provide definitive fraud decisions. Anomaly scores and rankings can help investigators focus their resources on transactions requiring greater attention and support more efficient monitoring within the bank switching environment. Accordingly, this study aims to develop and evaluate a graph-based anomaly detection framework for prioritizing suspected fraudulent transactions in bank switching data under limited availability of confirmed fraud labels. Specifically, the study investigates :

- 1) RQ1: whether a sequential IF–GNN–IF framework can enrich anomaly representations for suspected fraud detection.
- 2) RQ2: whether incorporating graph-based relational embeddings improves the identification of suspicious transactions compared with Isolation Forest based solely on transaction features.
- 3) RQ3: whether the proposed framework can support the prioritization of suspected fraudulent transactions when only Isolation Forest-generated pseudo-labels are available for evaluation.

The framework is evaluated using a real-world bank switching transaction dataset containing 71,873 transactions, with performance assessed using appropriate classification and anomaly-detection metrics, including recall.

2. METHODS

This study proposes a Hybrid Isolation Forest–Graph Neural Network–Isolation Forest (IF–GNN–IF) framework for detecting suspected fraudulent transactions in a bank switching system. The proposed framework combines statistical anomaly detection with graph-based representation learning to capture both anomalous transaction characteristics and complex relational patterns among transaction entities, including card identifiers, issuers, acquirers, merchants, terminals, and transaction timestamps. Isolation Forest is selected because it can efficiently identify anomalous observations without requiring labeled data [1], while Graph Neural Networks (GNNs) are employed to learn

structural relationships within the transaction graph and enrich the representation of suspicious transaction behavior [2], [3], [5].

The dataset consists of 71,873 bank switching transaction records, which are processed through a sequential three-stage computational framework. In the first stage, Isolation Forest performs coarse-grained anomaly detection by assigning an anomaly score to each transaction. The resulting anomaly scores are used to generate pseudo-labels for suspected anomalous transactions and are retained as additional features for the subsequent learning stage. In the second stage, transaction records are transformed into a graph representation, where nodes correspond to individual transactions and edges represent sequential and similarity-based relationships among transactions. Subsequently, a Graph Convolutional Network (GCN) is trained—guided by pseudo-labels generated by the first-stage isolation forest—to learn graph embeddings that capture transaction relational patterns [10], [11]. In the third stage, the learned graph embeddings are concatenated with the original normalized transaction features and re-evaluated using a second Isolation Forest to produce refined anomaly scores and prioritize suspected transactions for further investigation.

Unlike conventional fraud classification approaches, the proposed framework does not require confirmed fraud labels during model training. Instead, it focuses on suspected-fraud prioritization by integrating feature-based anomaly detection with graph-based relational learning under limited label availability. Similar graph-enhanced anomaly detection strategies have demonstrated promising performance in financial transaction analysis [7], [8], [9], [12], [13].

The overall methodology flow of the proposed framework is illustrated in Figure 1. The methodology is conducted through several stages, beginning with preprocessing and feature engineering, followed by initial anomaly detection using Isolation Forest, graph construction, GNN-based embedding learning, and final anomaly detection using a second Isolation Forest. As shown in Figure 1, the initial stage begins with preprocessing and feature engineering to clean the data, handle imbalanced values [14], and develop relevant features for the analysis process. The preprocessing and transaction behavior feature extraction stages aim to improve the quality of data representation before anomaly detection and graph learning processes [10], [15], [16].

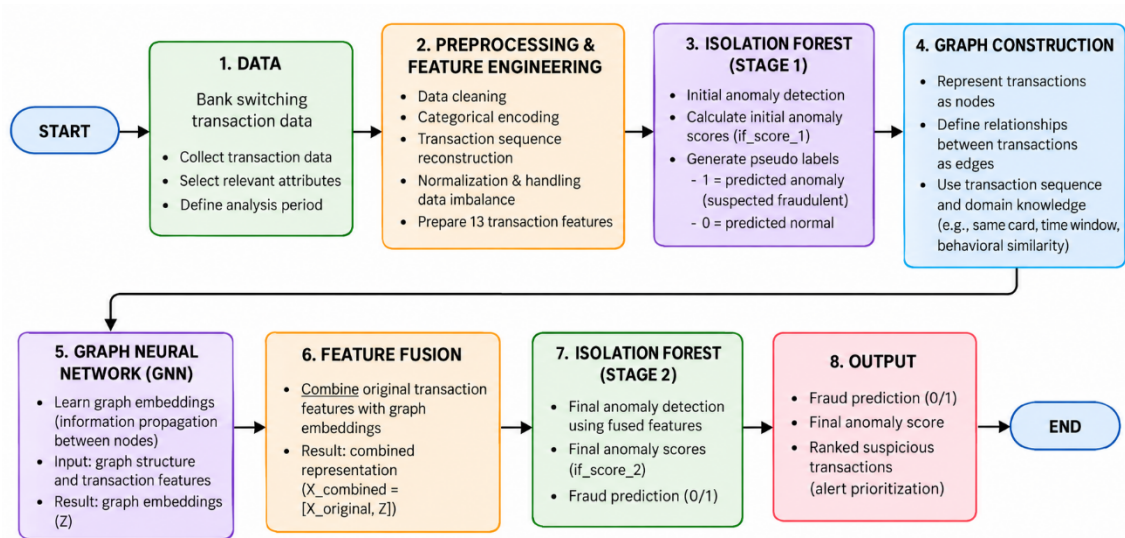


Figure 1. Metodology Flow

Next, initial detection is performed using Isolation Forest to generate anomaly scores based on individual transaction characteristics. Isolation Forest was chosen because it can detect anomalies without requiring labeled data and has efficient computational complexity for large-scale datasets [1], [17]. The results of this stage are used as the basis for constructing a graph that represents the relationships between transactions in the form of nodes and edges [3], [15].

In the next stage, embedding learning is carried out using a Graph Neural Network (GNN). The information propagation process between nodes in the GNN is performed using Equation (1).

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)}) \quad (1)$$

The resulting embedding is then combined with the original transaction features to form a more comprehensive data representation. The final step is performed using a second Isolation Forest to generate the final anomaly score. Integrating the graph embedding with transaction features allows the model to capture both attribute-based anomalies and relationship-based anomalies between transactions [3], [9], [10].

2.1. Dataset and Preprocessing

The dataset used is switching transaction data containing information such as transaction time, nominal amount, issuer, acquirer, terminal code or merchant, and card number are shown in Table 1. These attributes are commonly used in financial transaction suspected fraud detection systems because they can represent transaction characteristics and relationships between entities within a financial network [12], [13], [16].

Table 1. Dataset

id	timestamp	terminal_code	ref_number	card_number	issuer	acquirer	amount
3696401119	2025-03-22 09:49:21	11911236	515339096057	557692BZZEIEBIEE	Bank ADF	Bank XYZ	1785000
3696844863	2025-03-22 11:51:51	COR79175	202503222631	627486ZZZZZZZZB	Bank ACD	Bank ACD	4031772
3696283477	2025-03-22 12:07:52	00000000A0444001	006287806498	606058HZZZFAIHB	Bank ANO	Bank XYZ	50000
3696738429	2025-03-22 12:23:27	COR79333	202503222642	627486ZZZZZZZZB	Bank ACD	Bank ACD	993500
3696281491	2025-03-22 13:14:09	000000514	000000000518	627653ZAAGEFZBCE	Bank AGH	Bank AGH	5000000
3696405446	2025-03-22 13:42:29	6285701381876	223344927177	564417ZZEZIIFEAF	Bank AMN	Bank AMN	100000
3696403284	2025-03-22 15:06:03	11917132	515351637851	622011BZZZCACCAC	Bank AFG	Bank XYZ	2000000
3696359070	2025-03-22 15:28:33	COR80222	202503222675	627486ZZZZZZZZB	Bank ACD	Bank ACD	25000000
3697062638	2025-03-22 15:30:44	COR80237	202503222679	627486ZZZZZZZZB	Bank ACD	Bank ACD	9000000
3696385168	2025-03-22 16:08:03	11898648	515354303071	557692BZCAHEFGCC	Bank ADF	Bank XYZ	1500000
3696393016	2025-03-22 16:24:59	10974726	515355100693	622011BZZZCZGEID	Bank AFG	Bank XYZ	35000
3696378733	2025-03-22 16:27:59	11364022	515355246505	622011BZZZCEBIIB	Bank AFG	Bank XYZ	100000
3696397104	2025-03-22 17:24:49	000000199	000000000385	627653ZAAZAFEAFF	Bank AGH	Bank AGH	20000
3696867920	2025-03-22 19:22:26	COR81293	202503222709	627486ZZZZZZZZB	Bank ACD	Bank ACD	1093500
3696323224	2025-03-22 21:04:12	MB00016019	130483982610	627142GAZBBZZZCH	Bank AST	Bank AST	48708000
3696285616	2025-03-22 21:26:54	MB00016019	130486415344	627142GAZBBZZZCH	Bank AST	Bank AST	47955000
3696405525	2025-03-22 21:28:20	077725050	000000907228	627653ZBZGCBHGIZ	Bank AGH	Bank AGH	1000000
3696298581	2025-03-22 22:49:07	MB00016019	130484733808	627142GAZBBZZZCH	Bank AST	Bank AST	47683000
3696289493	2025-03-22 22:55:49	MB00016019	130484777954	627142GAZBBZZZCH	Bank AST	Bank AST	47729000
3696358099	2025-03-22 23:04:18	MB00016019	130487239090	627142GAZBBZZZCH	Bank AST	Bank AST	48903000

The preprocessing stages are performed sequentially. First, feature engineering is performed on the timestamp attribute to generate hour, day, and weekday features. Temporal features have proven effective in capturing unusual transaction behavior patterns and improving suspected fraud detection capabilities [13], [18], [19]. Second, categorical features such as issuer, acquirer, and merchant are converted into numeric form using label encoding so they can be processed by machine learning models [20]. Third, transaction behavior features are added to capture abnormal patterns, namely the time difference between transactions, the average and standard deviation of the transaction nominal on a rolling basis, the frequency of rapid transactions, and changes

in transaction nominals. This transaction behavior-based feature engineering is widely used in suspected fraud detection to identify irregularities in user activity [6], [12], [13]. Fourth, all features are normalized using StandardScaler to have a uniform scale so that each feature makes a balanced contribution during the model training process [20]. This normalization is stated in Equation 2.

$$z = \frac{x - \mu}{\sigma} \quad (2)$$

where x is the original feature value, μ is the mean of the feature in the training data, and σ is the standard deviation of the feature in the training data. The values z represent the normalized features. This process ensures that the data distribution has a uniform scale before being used in the next modeling stage [20].

2.2. Anomaly Detection Isolation Forest Stage 1 and Pseudo-Label Generation

Isolation Forest is an unsupervised learning method used to detect anomalies based on the concept of data isolation. This method operates on the assumption that anomalous data tends to be easier to separate than normal data because it has characteristics that differ from the majority of observations [1]. This approach is widely applied to financial transaction fraud detection because it does not require labeled data and is capable of handling large-scale datasets [13], [21].

An Isolation Forest consists of a collection of binary trees called isolation trees (iTree). Each tree is constructed by randomly selecting features and randomly assigning split values within a range of feature values. The splitting process is performed recursively until a node contains only one data point or the tree reaches its maximum depth [1]. For each observation x , the path length is calculated, which $h(x)$ is the number of steps required to isolate the data in the tree. In general, anomalous data has a shorter [1], [22]. path length than normal data because it is separated more quickly from the majority of the data. The average path length of the entire tree is expressed in Equation 3.

$$E(h(x)) = \frac{1}{t} \sum_{i=1}^t h_i(x) \quad (3)$$

where $E(h(x))$ is the average length of the observation path x , t is the number of trees used in the model, and $h_i(x)$ is the length of the observation path on the t th tree i [1]. To normalize the path length value, the normalization function is used as shown in Equation 4.

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \quad (4)$$

where $c(n)$ is the path length normalization factor, n is the number of samples, and $H(n)$ is the harmonic number which can be approximated using Equation 5.

$$H(n) \approx \ln(n) + \gamma \quad (5)$$

with $\gamma \approx 0.5772$ which is Euler's constant.

The anomaly score for each observation is calculated using Equation 6.

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \quad (6)$$

where $s(x, n)$ is the observation anomaly score x , $E(h(x))$ is the average path length, and $c(n)$ is the normalization factor. Score values close to 1 indicate that the data is highly anomalous, while values close to 0.5 indicate normal data. In the context of suspected fraud detection, transactions with extreme values, such as very large transaction amounts or unusual transaction times, tend to have shorter path lengths, resulting in higher anomaly scores. Conversely, normal transactions within dense data clusters require more separation processes and therefore have longer path lengths. Therefore, Isolation Forest is effective for detecting point anomalies and behavioral deviations in financial transactions [1], [13].

The first-stage Isolation Forest (IF_1) provides an initial anomaly signal used as a pseudo-label. For each transaction i , a decision score s_i is computed using the IF_1 decision function, where lower (more negative) scores represent a higher likelihood of being an anomaly. To convert these continuous scores into binary pseudo-labels, a threshold T_p is determined based on a predefined low percentile p of the decision-score distribution. The pseudo-label p_i for transaction i is defined as Equation 7.

$$p_i = \begin{cases} 1, & \text{if } s_i \leq T_p \text{ (or } IF_1(x_i) = -1) \\ 0, & \text{if } s_i > T_p \text{ (or } IF_1(x_i) = 1) \end{cases} \quad (7)$$

where x_i denotes the normalized feature vector of transaction i . Under this thresholding rule, transactions with decision scores at or below T_p are assigned $p_i = 1$ (indicating a suspected anomaly within the lowest score percentile), while the remaining transactions are assigned $p_i = 0$. These pseudo-labels are not treated as verified ground-truth fraud labels; rather, they serve as a weak supervision signal to guide the subsequent graph representation learning.

2.3. Graph Construction

To capture relationships between transactions, data is represented in a graph consisting of nodes and edges. In this study, nodes represent transactions, while edges represent relationships between transactions. Graph representation is used to model transaction interaction patterns so that relationships between entities can be analyzed in a more complex manner than a single-attribute-based approach [2], [3]. Graph-based approaches have proven effective in detecting anomalous patterns and fraudulent activities involving hidden relationships between entities in financial transaction networks [3], [5].

Graph construction is performed using two approaches, namely sequence-based edges and similarity-based edges. The sequence-based edge approach is used to connect transactions based on the time sequence on the same card. If two transactions originate from identical cards and occur sequentially within a certain time span, then the two transactions will be connected in the graph. This approach allows the model to capture sequential transaction behavior patterns and changes in user activity over time, which are often found in cases of financial transaction fraud [12], [13], [14], [17], [23]. The relationship between transactions can be expressed using Equation 8.

$$E_{seq}(i, j) = \begin{cases} 1, & \text{jika } card_i = card_j \text{ dan } t_j > t_i \\ 0, & \text{lainnya} \end{cases} \quad (8)$$

with $E_{seq}(i, j)$ is the relationship between transactions i and j based on time sequence, $card_i$ and $card_j$ is the card identity in the transaction, while t_i and t_j indicates the transaction time.

In addition, a similarity-based edge approach is used to connect transactions that have a certain level of similarity based on the difference in transaction nominal and transaction time difference. Similarity-based relationships aim to identify groups of transactions that exhibit similar characteristics and have the potential to form suspicious activity patterns in the transaction network [2], [5]. The similarity value between transactions is calculated using Equation 9.

$$S(i, j) = \alpha |amt_i - amt_j| + \beta |t_i - t_j| \quad (9)$$

where $S(i, j)$ is the similarity value between transaction i and j , amt_i and amt_j is the nominal transaction, t_i and t_j is the transaction time, while α and β are the weights for each similarity component. Through this approach, the resulting graph is able to represent transaction behavior patterns in a more complex way. The combination of sequential and similarity-based relationships allows the model to capture both local and global interactions between transactions, allowing hidden relationships that are difficult to identify using conventional individual feature-based methods to be learned more shows potential for prioritizing suspected transactions by the Graph Neural Network [3], [12].

2.4. Graph Neural Network (GNN)

Graph Neural Network (GNN) is used to learn node representation by considering the graph structure and relationships between nodes. This study uses a Graph Convolutional Network (GCN) architecture that aggregates information from neighboring nodes through a message passing process. GCN was chosen because it is able to efficiently integrate feature information and graph structure to learn node representation in complex networks [2], [11]. This approach has been widely applied in anomaly and fraud detection in financial transaction networks [3], [5]. A graph is represented as a pair of sets of nodes and edges as shown in Equation 10.

$$G = (V, E) \quad (10)$$

where V is the set of nodes that represent transactions and E is the set of edges that show the relationships between transactions. The features of each node are represented in the form of a Feature Metric in Equation 11.

$$X \in \mathbb{R}^{N \times F} \quad (11)$$

where N is the number of nodes and F is the number of features in each transaction. The propagation process in the GCN layer is carried out using Equation 12.

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)}) \quad (12)$$

with $H^{(l)}$ is the node representation in the l th layer, $W^{(l)}$ is the learned weight Metric, $\tilde{A} = A + I$ is the adjacency Metric with self-loop, $\tilde{D}$ is the degree Metric of $\tilde{A}$, and σ is the activation function [11]. In the initial layer, the node representation is determined using Equation 13.

$$H^{(0)} = X \quad (13)$$

The activation function used is Rectified Linear Unit (ReLU) as shown in Equation 14.

$$ReLU(z) = \max(0, z) \quad (14)$$

This function is used to help the model capture non-linear relationships between nodes. After going through several propagation layers, the final representation of the node or embedding is obtained as in Equation 15.

$$Z = H^{(L)} \quad (15)$$

with Z is a node embedding containing feature information and graph structure. This embedding is used as an additional feature in the advanced anomaly detection stage. To guide the representation learning process using the pseudo-labels, an auxiliary linear classifier is applied to the GCN embeddings is used as shown in Equation 16.

$$\hat{p}_i = \sigma(w^T z_i + b) \quad (16)$$

where z_i is the embedding of transaction i , w and b are learnable classifier parameters, and σ is the sigmoid activation function. The training objective is defined using binary cross-entropy is used as shown in Equation 17.

$$\mathcal{L}_{\text{GCN}} = -\frac{1}{N} \sum_{i=1}^N [p_i \log(\hat{p}_i) + (1 - p_i) \log(1 - \hat{p}_i)] \quad (17)$$

The loss encourages the GCN to generate embeddings that contain information relevant to the anomaly patterns identified during the initial screening stage. Because the supervisory signal originates from the Isolation Forest rather than verified fraud labels, the GCN stage is characterized as pseudo-label-guided or weakly supervised representation learning.

The binary cross-entropy objective is appropriate for the proposed embedding-learning stage because the purpose of the GCN is not to produce the final fraud decision, but to learn graph-aware latent representations that encode patterns associated with the initial anomaly signal. By minimizing the discrepancy between the predicted pseudo-labels and the IF1-derived pseudo-labels, the GCN is encouraged to organize transactions in the latent space according to their graph context and initial anomaly characteristics. Consequently, the resulting embeddings contain relational information that can complement the original transaction features.

In the context of fraud detection, each node is influenced not only by its own features but also by those of its neighbors. This allows the model to detect anomalous patterns that arise from interactions between transactions, such as repeated transactions within a short period of time, similar nominal patterns, and suspicious relationships between accounts. Thus, GNNs are able to detect relational anomalies that are difficult to detect using feature-based methods alone [3], [5], [13].

After GCN training, the auxiliary classifier is not used for final transaction classification. Instead, the trained GCN is used to extract the node embedding Metric Z . The embedding is concatenated with the original normalized transaction features as shown in Equation 18. The enriched feature representation is subsequently supplied to the second-stage Isolation Forest as shown in Equation 19

$$X_{\text{combined}} = [X_{\text{scaled}} \parallel Z] \quad (18)$$

$$S_{\text{final}} = \text{IF}_2(X_{\text{combined}}) \quad (19)$$

where S_{final} denotes the final anomaly score. Thus, the first-stage Isolation Forest provides the initial pseudo-label signal, the GCN transforms transaction and graph information into a relational representation, and the second-stage Isolation Forest performs the final anomaly detection and suspected-transaction prioritization.

Once the GCN training process concludes, the auxiliary classifier function is not used during the final transaction classification stage. The trained GCN model serves solely as a feature extractor to generate the node embedding Metric Z . Specifically, the second GCN layer extracts a 32-dimensional vector representation for each transaction, resulting in the following Metric as shown in Equation 20

$$\mathbf{Z} \in \mathbb{R}^{N \times 32} \quad (20)$$

with N representing the total transaction nodes. This relational embedding representation is then combined side by side (concatenation) with the original normalized transaction features ($X_{\text{"norm"}}$) as follows as shown in Equation 21

$$\mathbf{X}_{\text{combined}} = \text{Concat}(\mathbf{Z}, \mathbf{X}_{\text{norm}}) \quad (21)$$

Since the original normalized transaction feature Metric contains 13 features, the concatenation of the 13-dimensional transaction feature vector and the 32-dimensional GCN embedding produces a 45-dimensional enriched representation for each transaction.

2.5. Isolation Forest Stage 2 and Final Anomaly Detection

The 45-dimensional enriched feature Metric is subsequently processed by a second Isolation Forest (IF2). Beyond operating on the graph-enhanced feature space, IF2 serves a fundamentally different functional role compared to IF1. While IF1 functions as an uncalibrated preliminary screening filter aimed at generating coarse pseudo-labels to guide GCN representation learning, IF2 acts as the primary decision-making engine. Specifically, IF2 is tasked with computing anomaly scores and producing the final operational anomaly predictions by evaluating the enriched transaction representation that incorporates both individual transaction attributes and graph-derived information.

2.6. Model Configuration and Graph Statistics

To ensure reproducibility, the proposed Hybrid IF–GNN–IF framework was implemented using fixed model configurations throughout all experiments. The first-stage Isolation Forest was employed as a coarse anomaly detector using 100 isolation trees ($n_estimators = 100$), $contamination = 0.02$, $max_samples = "auto"$, and $random_state = 42$. The anomaly score produced in this stage was stored as `if_score_1` and subsequently used both for pseudo-label generation and as an additional feature in the graph learning process. Transactions whose anomaly scores belonged to the lowest anomaly-score percentile were assigned pseudo-labels representing suspected anomalous transactions. These pseudo-labels were used only for model training and evaluation because confirmed fraud labels were unavailable.

The Graph Neural Network was implemented using a two-layer Graph Convolutional Network (GCN). Each transaction was represented as a graph node with input features consisting of normalized transaction attributes and engineered behavioral features. The first GCN layer transformed the input features into a 32-dimensional hidden representation, followed by a Rectified Linear Unit (ReLU) activation function. The second GCN layer generated the final node embeddings. The model is optimized using the Adam optimizer with a learning rate of 0.01 over 50 training epochs, employing a full-batch learning strategy since the graph fits entirely within memory. The training loss curve over these 50 epochs is shown in Figure 2. The proposed framework performs weakly supervised GCN representation learning, utilizing pseudo-labels derived from the first-stage Isolation Forest. The embedding vectors generated by the GCN are concatenated with normalized original transaction features to form an enriched feature representation. These combined features are then processed by the second stage—an Isolation Forest—which produces the final anomaly score and a ranking of suspected fraud cases.

Graph construction combines both sequence-based and similarity-based relationships. Sequence-based edges connect consecutive transactions originating from the same card within a predefined temporal window of 120 seconds. Similarity-based edges are established according to transaction amount difference and transaction time difference. The similarity score is computed using equal weights ($\alpha = 0.5$ and $\beta = 0.5$) for the amount and temporal components. This dual-edge construction enables the graph to capture

both sequential transaction behavior and similarity among suspicious transaction patterns.

The final transaction graph contains 71,873 nodes, which corresponds to the total number of transaction records used in this study. Graph statistics, including the number of edges, average node degree, graph density, connected components, and the size of the largest connected component, are shown in Table 2. These values were obtained directly from the graph construction process prior to GNN training.

Categorical attributes, including issuer, acquirer, and merchant, were transformed using label encoding solely to provide numerical identifiers required by the machine learning implementation. The encoded values were not interpreted as ordinal variables during graph learning. Instead, graph connectivity was determined by explicit transaction relationships based on card identity, transaction sequence, transaction amount similarity, and temporal similarity. Consequently, the encoded numerical values did not introduce artificial ordering into the graph representation.

Table 2. Graph Statistics

Item	Amount
Number of Nodes	71,873
Number of Edges	18,039
Average Degree	0.5020
Graph Density	0.00000698
Connected Components	55,479
Largest Connected Component	633

2.7. Tools and Technology Used

This research utilizes several tools and technologies to support data processing, machine learning modeling, and visualization of analysis results. The technology selection was tailored to the needs of large-scale transaction data processing and the implementation of a hybrid Isolation Forest and Graph Neural Network (GNN) method. Apache NiFi is used as an ETL (Extract, Transform, Load) tool to manage transaction data flows. NiFi plays a role in data ingestion, data cleansing, and pipeline automation processes so that data

retrieval and transformation can be carried out in a structured and real-time manner. The data processing flow in NiFi can be represented using Equation 22.

$$D_{out} = T(E(D_{in})) \quad (22)$$

with D_{in} is input data, E is the extraction process, T is the data transformation process, and D_{out} is the processed data.

The Greenplum Database is used as the primary storage system to store historical switching transaction data. This database supports large-scale data storage, transaction query execution, and provides a primary data source for machine learning analysis. Python is used as the primary programming language for model development and data analysis. Libraries used include Pandas for data manipulation, NumPy for numerical computation, Scikit-learn for Isolation Forest implementation and data preprocessing, PyTorch for deep learning, PyTorch Geometric for Graph Neural Network implementation, and Matplotlib for visualizing analysis results. During the evaluation and visualization phase, Power BI is used to present anomaly detection results in an interactive dashboard. This dashboard is used for monitoring anomalies, visualizing fraud patterns, and compiling analysis reports, facilitating interpretation and decision-making.

Details regarding execution time and hardware specifications are as follows. The IF-GNN-IF framework was executed on an HP EliteBook 830 G10 (13-inch) notebook running Microsoft Windows 11 Pro, equipped with a 13th Gen Intel® Core™ i7-1355U processor (10 cores, 12 logical processors), 16 GB of LPDDR5 RAM (4800 MT/s), Intel® Iris® Xe Graphics, and an NVMe SSD. The measured graph construction time was 121.45 seconds, while GCN training for 50 epochs took 8.66 seconds. The inference time for the second Isolation Forest stage (IF2)—which encompasses prediction and anomaly scoring—was 2.36 seconds. These measurements provide an overview of the computational load of the proposed framework under the specified experimental hardware configuration.

3. RESULTS AND DISCUSSION

This study conducted the detection of suspected anomalous transactions using a hybrid method, namely the Isolation Forest (IF) algorithm followed by the Graph Neural Network

(GNN) algorithm for identifying anomalies in bank switching transactions. Isolation Forest was applied to identify deviations based on transaction characteristics, followed by the GNN algorithm to identify deviations based on transaction relationships represented in graph form. The results of this process were anomaly scores and prediction labels for each transaction. The results show that transactions characterized by unusual amounts, high transaction frequency within a short period, and specific relationships with other suspected anomalous transactions were associated with higher anomaly scores. The resulting anomaly scores and prediction labels were obtained for each transaction after the application of the hybrid IF–GNN framework.

3.1. Data Distribution and Pseudo-Labels

The first key finding relates to data distribution and pseudo-label formation . Since the dataset lacks an explicit fraud label, a pseudo-label approach was applied based on initial detection results using the Isolation Forest algorithm . Of the 71,873 transaction data sets, 68,279 were classified as normal (0) and 3,594 as fraud (1). These results indicate that the proportion of fraud is only around 5% of the total data, making the dataset imbalanced . This imbalanced condition aligns with the characteristics of real-world fraud , which is rare but has a significant impact. This finding answers the research question regarding fraud data representation and serves as a basis for further modeling. These results align with the literature stating that anomaly detection in financial transactions generally faces the problem of extreme class imbalance. The implication is that the method used in the next stage must be able to handle this imbalance to avoid bias towards the majority class.

3.2. GNN Model Training Results

Based on the GNN model training results shown in Figure 2, the loss graph shows a significant decrease from the initial epoch to a very small value at the final epoch. The graph's characteristics include a sharp decrease at the beginning of training, a gradual decline after the 10th epoch, and a stable value approaching zero at the final epoch. This indicates that the model has successfully achieved convergence, the embedding representation between nodes is stable, and there is no divergence or instability during the training process. However, it should be noted that the loss function used is unsupervised, so a small loss value does not directly reflect the model's ability to detect fraud.

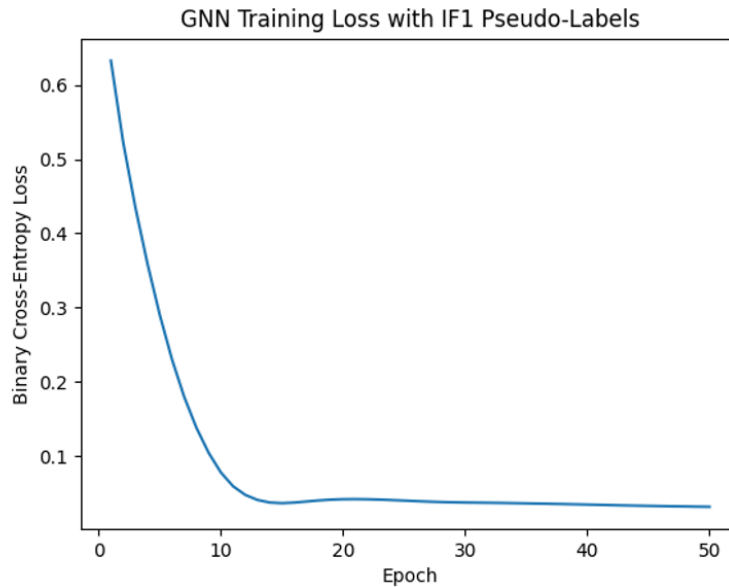


Figure 2. Hasil Training Model GNN

The Graph Neural Network primarily enriches transaction representations by incorporating relational information among issuers, merchants, card numbers, and temporal interactions. In the current experiment, the hybrid framework mainly affected the anomaly-score distribution rather than substantially increasing recall. The observed improvement was reflected in the higher separation between suspicious and normal transactions, contributing to high precision while maintaining a relatively conservative detection threshold.

3.3. Model Evaluation

Since confirmed fraud labels were unavailable, model evaluation was performed against pseudo-labels generated by the first-stage Isolation Forest. Therefore, the reported metrics should be interpreted as the capability of the proposed framework to prioritize suspicious transactions rather than to classify confirmed fraud cases. Based on the test results, the graph statistics shown in Table 3 were obtained.

Table 3. Graph Statistics

Item	Amount
Number of Nodes	71,873
Number of Edges	18,039

Item	Amount
Average Degree	0.5020
Graph Density	0.00000698
Connected Components	55,479
Largest Connected Component	633

The graph contains 71,873 transaction nodes and 18,039 edges, resulting in 55,479 connected components, while the largest connected component contains 633 nodes. This fragmented structure is primarily a consequence of the selective edge construction strategy. Sequence-based edges connect transactions within the same card sequence, while similarity-based edges require the same card and sufficiently close transaction characteristics within the specified search window. Consequently, transactions with limited or no qualifying relationships remain isolated or form small components. The resulting graph therefore represents localized transaction relationships rather than a single globally connected network. This structure allows the GNN to learn relational patterns within connected transaction groups, while transactions in separate components do not exchange information directly.

Table 4. Confusion Metric

Category	Mark	Meaning
True Negative (TN)	68,164	Normal detected correctly
False Positive (FP)	115	False alarm
False Negative (FN)	2,271	Fraud not detected
True Positive (TP)	1,323	Fraud detected

The confusion metric results indicate that the model successfully detected the majority of normal transactions and generated a low number of false alarms. However, a number of suspected anomalous transactions remained undetected. A comprehensive comparison of metrics between IF Stage 1 and Hybrid IF–GNN–IF—based on Accuracy, Precision, Recall, F1-score, Balanced Accuracy, Specificity, False Negative Rate, ROC-AUC, PR-AUC, and Precision@100—is presented in Table 5.

Table 5. Model Evaluation Metrics

Metric	IF Stage 1	Hybrid IF-GNN-IF
Accuracy	0.9700	0.9668
Precision	1.0000	0.9200
Recall	0.4001	0.3681
F1-score	0.5715	0.5258
Balanced Accuracy	0.7001	0.6832
Specificity	1.0000	0.9983
False Negative Rate	0.5999	0.6319
ROC-AUC	1.0000	0.9778
PR-AUC	1.0000	0.7619
Precision@100	1.0000	1.0000
Precision@500	1.0000	1.0000
Precision@1000	1.0000	0.9480

A precision value of 0.9200 indicates that most transactions predicted as fraudulent are indeed anomalous, resulting in a low false alarm rate for the model. A recall value of 0.3681 indicates that the model is still unable to optimally detect all suspected anomalous transactions. An F1-score of 0.5258 indicates that the balance between precision and recall still needs improvement. Meanwhile, an accuracy of 97% cannot be used as a primary indicator due to the imbalanced nature of the dataset used. Balanced Accuracy 0.6832: Much more representative than standard accuracy. Specificity 0.9983: Very good at recognizing normal transactions. False Negative Rate 0.6319: High, consistent with low recall. ROC-AUC 0.9778: Very good, indicating strong anomaly score ranking. PR-AUC 0.7619: Very good for an imbalanced dataset. Precision@100 and Precision@500: 1.0000: All 100 transactions with the highest anomaly scores are pseudo-labeled positive. Precision@1000 0.9480: 94.8% of the 1,000 transactions with the highest anomaly scores are pseudo-labeled positive

The relatively low recall suggests that the selected anomaly threshold favors precision over sensitivity. Lowering the anomaly threshold would likely increase recall by detecting more suspicious transactions but would also increase false positives. Therefore, threshold optimization represents an important direction for future work. A Precision@1000 of 0.9480 indicates that approximately 94 of the top 100 ranked transactions correspond to

the positive reference category used for evaluation." This indicates that the ranking can potentially reduce the initial investigation workload by directing analysts toward a small set of high-priority suspicious transactions.

3.3.2 Model Stability Analysis

To evaluate the performance variation of the proposed Hybrid IF-GNN-IF framework under different random initializations, the evaluation was repeated using five random seeds (1, 5, 10, 20, and 42) while maintaining the same model configuration and dataset. The evaluation results for each random seed are presented in Table 6.

Table 6. Performance Stability Across Random Seeds

Seed	Accuracy	Precision	Recall	F1-score
1	0.966218	0.905424	0.362270	0.517488
5	0.967053	0.926287	0.370618	0.529412
10	0.965884	0.897079	0.358932	0.512719
20	0.966302	0.907510	0.363105	0.518680
42	0.966803	0.920028	0.368114	0.525835

Across the five random seeds, the accuracy values ranged from 0.965884 to 0.967053, precision values ranged from 0.897079 to 0.926287, recall values ranged from 0.358932 to 0.370618, and F1-score values ranged from 0.512719 to 0.529412. The mean and standard deviation of each evaluation metric are presented in Table 7.

Table 7. Mean and Standard Deviation

Metric	Mean $\pm$ Std
Accuracy	0.9665 $\pm$ 0.0005
Precision	0.9113 $\pm$ 0.0117
Recall	0.3646 $\pm$ 0.0047
F1-score	0.5208 $\pm$ 0.0067

A sensitivity analysis regarding thresholds was also performed, utilizing specific percentages (0.5%, 1%, 2%, 3%, and 5%) to illustrate the changes observed across different anomaly thresholds; the results are presented in Table 8.

Table 8. Threshold Sensitivity Analysis

Threshold Percentile	Threshold Score	Accuracy	Precision	Recall	F1-score	Balanced Accuracy	Specificity	False Negative Rate
0.5 %	-0.0882	0.9550	10.000	0.1002	0.1821	0.5501	10.000	0.8998
1%	-0.0683	0.9596	0.9765	0.1967	0.3275	0.5982	0.9998	0.8033
2%	0.0000	0.9668	0.9200	0.3681	0.5258	0.6832	0.9983	0.6319
3%	0.0478	0.9691	0.8183	0.4911	0.6138	0.7427	0.9943	0.5089
5%	0.0964	0.9679	0.6789	0.6789	0.6789	0.8310	0.9831	0.3211

The evaluation results of the proposed Hybrid IF–GNN–IF framework are presented based on the obtained classification and ranking metrics. The framework achieved an accuracy of 0.9668, precision of 0.9200, and recall of 0.3681. The corresponding F1-score was 0.5258, while the ROC-AUC value was 0.9778. The obtained precision value was higher than the recall value. The evaluation also produced the corresponding numbers of true positive, false positive, true negative, and false negative classifications, as reported in Table 4. The resulting anomaly scores were used to rank transactions according to their anomaly level. The ROC-AUC value was obtained from the model's prediction scores across different classification thresholds. The evaluation metrics at the selected operating threshold are presented in Table 5, while the threshold sensitivity results are presented in Table 8.

3.4. Model Comparison

The performance comparison between the initial Isolation Forest model and the hybrid IF + GNN model is shown in Table 9.

Table 9. Comparison of Anomaly Detection Models

Model	Mean Score	Std Dev	% Anomaly
IF Stage 1	0.22	0.06	2.0%
IF + GNN	0.21	0.06	2.0%

Based on Table 9, the Hybrid IF–GNN–IF framework shows a decrease in the mean anomaly score from 0.22 in IF Stage 1 to 0.21 after incorporating GNN-based representations. This change indicates a slight shift in the overall anomaly score distribution after graph-based feature enrichment. The standard deviation remains

unchanged at 0.06, suggesting that the overall variability of anomaly scores is relatively stable between the two stages. The anomaly percentage also remains at 2.0%, as both Isolation Forest stages use the same contamination parameter of 0.02. Therefore, the contribution of the GNN is not reflected in an increase in the proportion of detected anomalies, but in the enriched representation used by the second-stage Isolation Forest. The effectiveness of the hybrid framework is therefore better assessed using the detailed evaluation metrics, including Precision, Recall, ROC-AUC, PR-AUC, and Precision@1000, rather than by the mean anomaly score alone.

3.5. Visualization of Anomaly Detection Results

The anomaly scores and prediction results generated by the proposed Hybrid IF–GNN–IF framework were visualized using an interactive Power BI dashboard. The dashboard presents the distribution of anomalous transactions by issuer, transaction number, anomaly score, and prediction status.

As shown in Figure 3, the dashboard displays the distribution of anomalous transactions across different issuers. The number of anomalous transactions recorded for each issuer is presented in the visualization, with Bank AST showing the highest number of anomalous transactions among the issuers included in the dataset. The dashboard also presents transaction frequencies and anomaly scores for individual cards.

The visualization shows several transaction patterns associated with higher anomaly scores, including high transaction frequency within a short timeframe, concentration of anomalous transactions among particular issuers, and anomaly scores that deviate from the observed distribution. The corresponding transaction details and prediction statuses are displayed in the dashboard for further examination.

Figure 3 therefore provides a visual representation of the anomaly scores and prediction results generated by the proposed framework. The quantitative performance of the model is evaluated separately using the classification metrics presented in the preceding evaluation results.

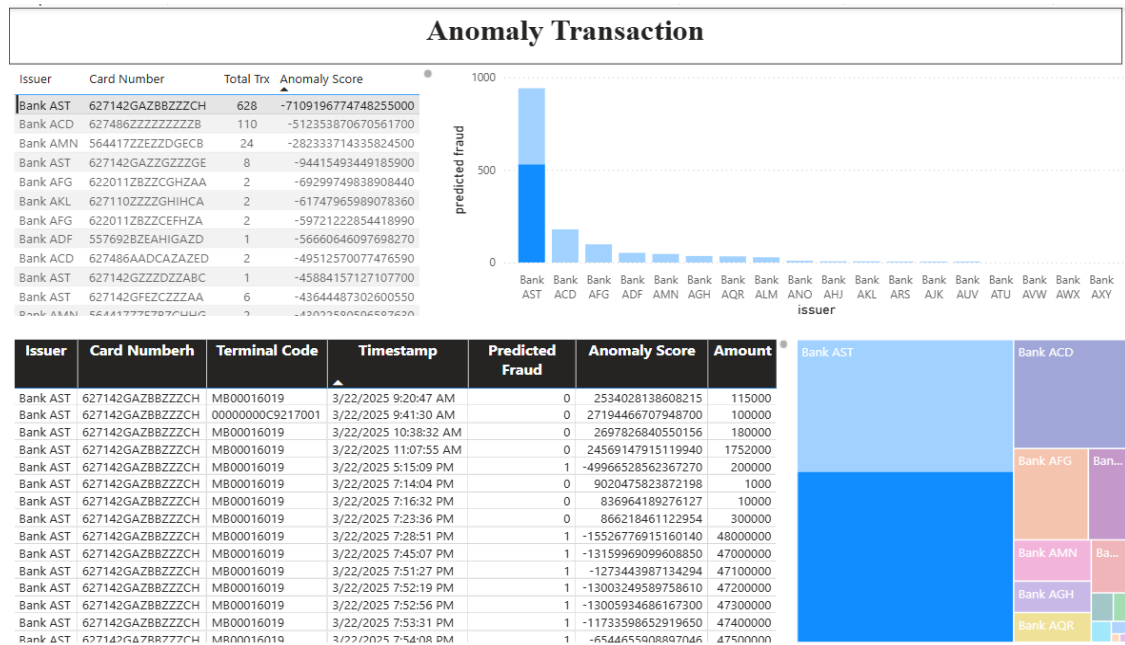


Figure 3. Anomalous Transaction Visualization Dashboard

As shown in Figure 4, a Power BI dashboard was developed to visualize the outputs of the proposed Hybrid IF–GNN–IF framework. The dashboard summarizes key information, including the total number of processed transactions (71,873), the number of transactions identified as suspicious (1,438), the proportion of anomalous transactions (2%), the total transaction value, the cumulative transaction value associated with suspicious transactions, and the proportion of anomalous transaction value relative to the total transaction value.

The dashboard also provides an issuer-level summary containing the number of processed transactions, cumulative transaction amount, and anomaly scores for each issuer. The corresponding values are presented for each issuer in the dashboard. In addition, the dashboard provides a temporal visualization of transaction amounts by hour for normal and suspicious transactions.

The temporal visualization shows the hourly distribution of transaction amounts for the two transaction categories. Suspicious transaction amounts are concentrated during specific periods of the day, while normal transaction amounts are distributed across the observed operational hours. The dashboard also presents the corresponding transaction counts, amounts, anomaly scores, and prediction statuses for the analyzed transactions.

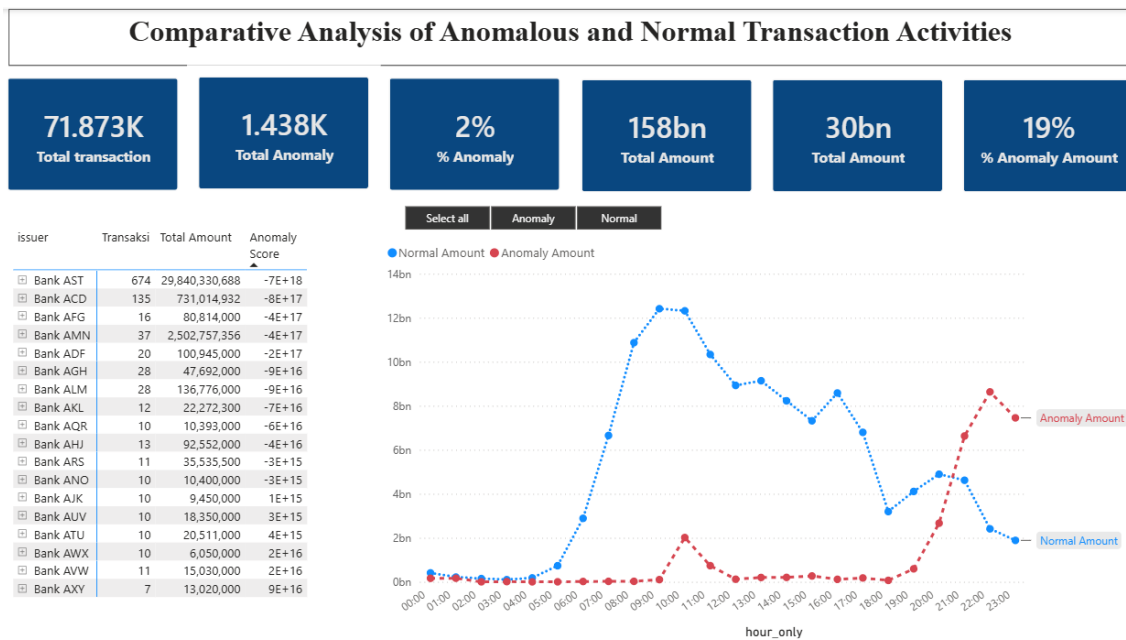


Figure 4. Comparative Anomaly Analysis Table

The final anomaly scores generated by the Hybrid IF–GNN–IF framework were used to rank transactions according to their anomaly levels. Transactions with higher anomaly scores were placed at higher positions in the resulting ranking. As illustrated in Figures 3 and 4, transactions associated with specific issuers, high transaction frequencies, and unusually large transaction amounts appeared among the highest-ranked suspicious transactions. The resulting ranking and anomaly scores were presented through the Power BI dashboard.

3.6. Computational Cost

The computational execution of the proposed Hybrid IF–GNN–IF framework consists of three main stages: (1) first-stage Isolation Forest, (2) graph construction and GNN training, and (3) second-stage Isolation Forest. The execution time and computational resource utilization recorded for each stage are presented in Table 10. Among the three stages, graph construction and GNN training recorded the highest computational requirements, while the first- and second-stage Isolation Forest processes recorded lower computational requirements.

Table 10. Detail Runtime

Item	Time
IF1 inference time	1.6571 Second
Graph construction time	133.0235 Second
GCN training time	10.7849 Second
IF2 inference time	2.8249 Second

3.7. Discussion

The proposed Hybrid IF–GNN–IF framework combines feature-based and relationship-based anomaly detection to identify suspected anomalous patterns in bank switching transactions. Isolation Forest captures deviations in individual transaction characteristics, while the Graph Neural Network (GNN) incorporates relationships among transaction entities, including banks, merchants, card identifiers, and transaction times. This combination is consistent with previous studies on graph-based financial anomaly detection, which indicate that relational information can complement conventional feature-based analysis by capturing dependencies among transaction entities. The resulting anomaly scores can therefore be used to rank suspected transactions and support the prioritization of transactions requiring further investigation.

The evaluation results should be interpreted in the context of the labeling strategy used in this study. Because the framework relies on pseudo-labels generated by the first-stage Isolation Forest rather than independently verified fraud labels, the identified transactions cannot be regarded as confirmed fraudulent transactions. The anomaly scores are more appropriately viewed as a ranking mechanism for suspected transactions, with higher-ranked transactions presented to fraud analysts for further examination. In an analyst-in-the-loop implementation, the dashboard can support the review process by presenting anomaly scores and relevant transaction information, while the final determination remains the responsibility of human analysts.

Several limitations should be considered. The model was evaluated using the complete dataset of 71,873 transactions used during representation learning and anomaly detection; therefore, the reported results do not represent independent out-of-sample generalization performance. In addition, the quality of the pseudo-labels and the selection of anomaly thresholds may affect the resulting detection performance.

Quantitative comparisons with alternative methods, such as Local Outlier Factor, One-Class SVM, Autoencoder, and standalone GNN, were also outside the scope of this study. Future research should evaluate the framework using temporally or transaction-wise partitioned test data and independently verified fraud labels when available. Further improvements may include threshold optimization, additional transaction-behavior features, model tuning, semi-supervised learning, and the integration of contextual information such as device characteristics and transaction location.

Privacy and data governance are also important considerations for operational implementation. In this study, sensitive attributes, including card numbers, issuer identifiers, and acquirer identifiers, were hashed during data extraction before further processing. This procedure preserved the relational information required for graph construction while reducing direct exposure of sensitive transaction identifiers. For future operational deployment, maintaining records of model versions, processing timestamps, anomaly scores, and analyst decisions could further support traceability and auditability.

4. CONCLUSION

The proposed Hybrid Isolation Forest–Graph Neural Network–Isolation Forest (IF–GNN–IF) framework addresses the research aim by combining transaction-level anomaly detection with graph-based relational learning to prioritize suspected anomalous transactions in bank switching data. Using pseudo-labels generated by the first-stage Isolation Forest, the framework achieved a Precision of 0.9200, Recall of 0.3681, F1-score of 0.5258, ROC-AUC of 0.9778, PR-AUC of 0.7619, and Precision@100 of 1.0000, indicating that the framework can effectively rank transactions that are highly consistent with the generated pseudo-labels, although its relatively low recall limits its suitability for comprehensive fraud screening. Because the framework has not been validated against confirmed fraud labels or independent expert-reviewed cases, these results represent suspected-anomaly prioritization performance rather than confirmed fraud detection performance. Therefore, the proposed framework is most appropriately positioned as an offline decision-support and alert-prioritization mechanism that assists fraud analysts in selecting transactions for further investigation, rather than as an automated fraud-blocking system. Future research should validate the framework using confirmed fraud

labels and expert review, while improving recall through threshold optimization and cost-sensitive learning, and further addressing graph representation, interpretability, scalability, and production-level validation.

REFERENCES

- [1] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation forest," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, pp. 413–422, 2008, doi: 10.1109/ICDM.2008.17.
- [2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021, doi: 10.1109/TNNLS.2020.2978386.
- [3] X. Ma *et al.*, "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12012–12038, Dec. 2023, doi: 10.1109/TKDE.2021.3118815.
- [4] S. Priyadarshi, J. Kumar, and N. Arunachalam, "Financial fraud detection using graph neural networks," in *Proc. 2025 IEEE 4th World Conf. Appl. Intell. Comput. (AIC), 2025*, pp. 537–541, 2025, doi: 10.1109/AIC66080.2025.11212157.
- [5] H. Kim, B. S. Lee, W. Y. Shin, and S. Lim, "Graph anomaly detection with graph neural networks: Current status and challenges," *IEEE Access*, vol. 10, pp. 111820–111829, 2022, doi: 10.1109/ACCESS.2022.3211306.
- [6] P. Mahesh, M. Suresh Chinnathampy, M. Parthiban, C. Jeevan Kumar, R. Aravinth, and G. Karthick, "Credit card fraud detection in banking using machine learning," in *Proc. 2025 Int. Conf. Recent Adv. Elect., Electron., Ubiquitous Commun., Comput. Intell. (RAEEUCCI), 2025*, doi: 10.1109/RAEEUCCI63961.2025.11048184.
- [7] Y. Sun, "A method for blockchain financial transaction fraud detection based on graph neural networks," in *Proc. 2025 IEEE Int. Conf. Commun. Netw. Comput. (CNC)*, pp. 71–76, 2025, doi: 10.1109/CNC68716.2025.11484570.
- [8] O. Omid, M. Kargari, and S. Khosravi, "Fraud detection in banking transaction network using a hybrid deep learning method," in *Proc. Int. Conf. Web Res. (ICWR)*, pp. 541–546, 2025, doi: 10.1109/ICWR65219.2025.11006253.
- [9] U. K. Chilakalapalli, P. Sachdev, B. Mohan, and A. Khandelwal, "Graph neural networks for real-time transaction fraud detection in payment networks," in *Proc. 2026 Int. Conf. Smart Appl., Commun. Netw. (SmartNets), 2026*, doi: 10.1109/SMARTNETS69662.2026.11604901.

- [10] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.
- [11] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Representations (ICLR)*, Sep. 2016, [Online]. Available: <https://arxiv.org/pdf/1609.02907>
- [12] Z. Shao, X. Wang, E. Ji, S. Chen, and J. Wang, "GNN-EADD: Graph neural network-based e-commerce anomaly detection via dual-stage learning," *IEEE Access*, vol. 13, pp. 8963–8976, 2025, doi: 10.1109/ACCESS.2025.3526239.
- [13] R. S. Menezes and R. H. Filho, "Investigating the robustness of graph neural networks to data drift: A case study on financial transaction data," *IEEE Access*, vol. 13, pp. 164302–164312, 2025, doi: 10.1109/ACCESS.2025.3611817.
- [14] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proc. Int. Conf. Inf. Knowl. Manage. (CIKM)*, pp. 315–324, 2020, doi: 10.1145/3340531.3411903.
- [15] H. Dong, G. Frusque, Y. Zhao, E. Chatzi, and O. Fink, "NNG-Mix: Improving semi-supervised anomaly detection with pseudo-anomaly generation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 6, pp. 10635–10647, 2025, doi: 10.1109/TNNLS.2024.3497801.
- [16] D. Wang *et al.*, "A semi-supervised graph attentive network for financial fraud detection," in *Proc. 2019 IEEE Int. Conf. Data Mining (ICDM)*, pp. 598–607, 2019, doi: 10.1109/ICDM.2019.00070.
- [17] M. Jin *et al.*, "A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 10466–10485, 2024, doi: 10.1109/TPAMI.2024.3443141.
- [18] F. Khaled Alarfaj and S. Shahzadi, "Enhancing fraud detection in banking with deep learning: Graph neural networks and autoencoders for real-time credit card fraud prevention," *IEEE Access*, vol. 13, pp. 20633–20646, 2025, doi: 10.1109/ACCESS.2024.3466288.
- [19] A. Correa Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, "Feature engineering strategies for credit card fraud detection," *Expert Syst. Appl.*, vol. 51, pp. 134–142, Jun. 2016, doi: 10.1016/j.eswa.2015.12.030.
- [20] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011, [Online]. Available: <http://scikit-learn.sourceforge.net>.

- [21] C. Chen *et al.*, "InfDetect: A large scale graph-based fraud detection system for e-commerce insurance," in *Proc. 2019 IEEE Int. Conf. Big Data (Big Data)*, pp. 1765–1773, 2019, doi: 10.1109/BIGDATA47090.2019.9006115.
- [22] X. Zhuang and J. Wu, "Anomaly detection in complex systems using extended isolation forest and denoising autoencoder," *IEEE Sens. J.*, vol. 26, no. 5, pp. 7408–7418, 2026, doi: 10.1109/JSEN.2026.3655746.
- [23] S. Motie and B. Raahemi, "Financial fraud detection using graph neural networks: A systematic review," *Expert Syst. Appl.*, vol. 240, p. 122156, Apr. 2024, doi: 10.1016/j.eswa.2023.122156.