Graph Neural Networks for Fraud Detection: Modeling Financial Transaction Networks at Scale

Omkar Reddy Polu[1], Balaiah Chamarthi[2], Tanay Chowdhury[3], Azhar Ushmani[4], Pratik Kasralikar[5], Abdul Aleem Syed[6], Aashish Mishra[7], Sathish Krishna Anumula[8], Rethish Nair Rajendran[9], Manas Ranjan Mohanty[10], Nuzhat Noor Islam Prova[11*]

[1]Department of Technology and Innovation, City National Bank, CA.
[2]Department of Technology Innovation, Info Services LLC, Livonia, MI.
[3]Data Science, AWS Gen AI Innovation Center, Sammamish, WA.
[4]Information Security Department, Amazon Web Services (AWS), TX.
[5]Department of Business Administration, Lindsey Wilson College, KY.
[6]SVP Technical Product Management, FHN Financial, Katy, TX.
[7]Department of Computers and Information Science, EKU, KY.
[8]IBM Corporation, Detroit, MI, 48375, USA.
[9]Delivery Management, Cloud Services, Unisys Corporation, NY.
[10]Amazon AGI, Sunnyvale, California, 94089, USA.
[11*]Independent Researcher, NY, 10038, USA.

*Corresponding author(s). E-mail(s): nuzhatnsu@gmail.com;
Contributing authors: omkar122516@gmail.com; balaiahc@gmail.com; tanaychowdhury@gmail.com; azhar.ushmani@gmail.com; pratikkasralikar@gmail.com; aleem87@gmail.com; vipashish64@gmail.com; sathishkrishna@gmail.com; rethishrnair@gmail.com; itsmanasmohanty@gmail.com;

Abstract

The worldwide economies are being seriously impacted by financial fraud, requiring proficient detection techniques able to spot changing and complex fraudulent activity. Conventional Machine Learning (ML) models and rule-based approaches among other traditional fraud detection systems find it difficult to scale, and adaptably capture relational fraud patterns in vast financial transaction networks,

© The Author(s) 2025
R. Nagariya et al. (eds.), *Proceedings of the International Conference on Sustainable Business Practices and Innovative Models (ICSBPIM-2025)*, Advances in Economics, Business and Management Research 349,
https://doi.org/10.2991/978-94-6463-872-1_45

and We present a new Graph Neural Network (GNN)-based fraud detection model that improves both computational efficiency and detection accuracy in order to meet these issues. To develop expressive node representations, our method combines multi-hop neighborhood aggregation with attention methods, hence providing strong fraud detection. We also provide a hybrid detection system using community-based anomaly detection (Yelp-inspired) for detecting behavioral similarities and transaction-based embeddings (Amazon-inspired) to identify subconscious fraud movements. We use adaptive filtering systems and reinforcement learning-based neighbor selection to get above the constraints of highly imbalanced datasets, thus enhancing fraud detection performance and decreasing false positives. With 95.00% accuracy, 93.10% precision, 93.15% recall, and 93.20% AUC, experimental evaluations on real-world Yelp and Amazon datasets show that our model beats currently used GNN-based models including GCN, GAT, and GraphSAGE considerably. These findings confirm the success of our model in identifying large-scale fraudulent activity, providing a very attractive alternative for the prevention of financial fraud.

Keywords: Fraud Detection , Graph Neural Network (GNN) , Anomaly Detection , Reinforcement Learning , Imbalanced Data , Financial Transaction , Yelp Dataset , Amazon Dataset , Multi-relational Graphs

1 Introduction

Nowadays, a lot of people obtain many benefits from financial services—especially online ones while continuously they greatly assist society economically. Yet, we are also observing growing numbers of financial frauds. As an instance, a record 154 million consumers in the US alone reported having scam experience [1]. The estimation costs financial institutions billions of dollars yearly, financial fraud is a rising menace to the world economy. As fraudsters always change their strategies, conventional fraud detection methods are useless against different fraud schemes [2].

In the financial technology (FinTech) and banking sectors, financial fraud detection is a major difficulty needing strong and scalable solutions to reduce illegal activity like credit card fraud, money laundering, and fraudulent transactions [3]. Developing as a useful tool to replicate complicated financial transaction networks, GNNs provide more advanced and flexible fraud detection systems [4]. Financial criminals always change their strategies, so even with the developments in conventional fraud detection methods including rule-based systems and supervised machine learning models, it is more and more difficult to find fraudulent activity using stationary models. Current methods limit their capacity to generalize over various fraudulent patterns by not using the relational and structural relationships inherent in financial transactions. The main issue this study tries to solve is the scalability and efficiency of fraud detection systems in networks of major financial transactions [5]. Current GNN-based fraud detection algorithms suffer from problems like high processing costs, dynamic graph structures, and imbalanced datasets, consequently producing unsatisfactory detection accuracy.

We've developed a novel GNN model to solve these difficulties, inspired by Yelp and Amazon datasets, which are well-known for their complicated graph structures and high-dimensional user-product interactions. This GNN model learns more expressive node representations for fraud detection by combining multi-hop neighborhood aggregation and attention techniques [6]. While the Amazon-inspired method uses transaction-based embeddings to capture hidden fraud patterns in vast-scale transaction networks, the Yelp-inspired component concentrates on community-based anomaly detection employing user behavior similarity to identify fraudulent activity. Combining these two techniques results in our model with enhanced scalability, better generalization, and higher fraud detection accuracy in practical financial transaction environments.

Our key contributions include:

1. We utilize multi-hop neighborhood aggregation and attention techniques to learn expressive node representations from financial transaction networks, hence improving computational efficiency and accuracy by means of a novel GNN-based fraud detection model.
2. Combines transaction-based inserts (Amazon dataset) for overlooked fraud pattern recognition and community-based anomaly detection (Yelp dataset) to capture dependent fraud behaviors, hence introducing a hybrid fraud detection strategy.
3. Implements reinforcement learning-based neighbor selection and adaptive filtering approaches to successfully handle class imbalance in financial fraud datasets, therefore guaranteeing improved detection of common fraudulent transactions in both Yelp and Amazon datasets.
4. On the Amazon and Yelp datasets, exceeds state-of-the-art models by attaining greater accuracy (95.00%), precision (93.10%), recall (93.15%), and AUC (93.20%). This indicates better fraud detection capacity than GCN, GAT, and GraphSAGE.

The rest of this paper is structured as follows: In section 2, the applications of ML in high-frequency trading and risk management are reviewed in the literature. In Figure 1, we describe the methodology, including dataset selection, preprocessing techniques, and the ML model architectures used for trading predictions and risk mitigation. The experimental results and performance analysis are presented in section 4. Finally, insights and future research directions are concluded in section 5.

2 Literature Review

This review of the literature investigates several GNN-based models for financial fraud detection, therefore emphasizing developments in fraud categorization through hierarchical attention, adaptive sampling, and graph-based anomaly detection. Using real-world financial data, it contrasts model performance and looks at important difficulties including high computing costs, reliance on labeled data, real-time flexibility, and data reliability issues.

Wang et al. [1] expressed SemiGNN, a GNN model that combines labeled and unlabeled financial data to improve fraud detection. Using hierarchical attention techniques, the model analyzes multiview data including user interactions and attributes

for enhanced fraud classification. Using Alipay's financial data, experimental analysis shows that SemiGNN beats conventional approaches with an AUC of 0.807 and a KS score of 0.464, hence beating models like XGBoost (AUC 0.753) and GCN (AUC of 0.780). Tian et al. [7] suggested a model ASA-GNN to train discriminative representations of transaction data, consequently enhancing fraud detection. To filter noise, they choose essential transaction neighbors using adaptive sampling and aggregation methods employing cosine similarity and edge weights. The outperformance of ASA-GNN over conventional models such as GraphSAGE and GAT is demonstrated by experimental evaluations on three real-world financial datasets showing an AUC of 92.2% on PR01 and 83.5% on TC12. Likewise, Kurshan et al. [8] investigated how GNNs might help to identify financial crimes and fraud. To find fraudulent trends in transaction networks, the work uses graph-based machine learning methods like flow-based path analysis, sub-graph analysis, and graph anomaly identification. Graph-based fraud detection models show better performance than conventional rule-based systems according to experimental assessments, therefore raising fraud detection accuracy by up to 15%.

Furthermore, Kadam et al. [9] designed to solve constraints in current GNNs, introduces JA-GNN, a new graph-based fraud detection model. Using attention processes and mutual neighbor-based sampling, the researchers improve fraud detection accuracy while reducing over-smoothing problems. JA-GNN beats state-of-the-art models including GTAN and Semi-GNN with evaluations on the Proprietary Financial Fraud Dataset (PFFD) and Yelp Fraud Dataset showing an AUC of 0.897 and recall of 0.868 on PFFD, and an AUC of 0.951 and recall of 0.99 on Yelp. Takahashi et al. [2] Inspired by financial transactions demonstrated as a graph to capture complicated fraud patterns, offers a GNN-based fraud identifying framework. Graph convolutional layers, attention mechanisms, and reinforcement learning form part of the approach to dynamically change fraud classification thresholds. Outperformance of the model above conventional fraud detection methods is shown by experimental assessments on large-scale financial transaction datasets with an AUC-ROC of 0.942 and an F1-score of 0.913. Wang et al. [10] displayed uses include fraud detection, stock movement prediction, loan default risk assessment, and recommendation systems, offering a thorough examination of how GNNs are utilized in finance. Reviewing GNN methods applied for each, the study groups financial graphs into homogeneous, bipartite, multi-relation, and dynamic graphs. The survey shows that, by including relational data, GNN-based models beat conventional techniques in tasks including stock prediction by up to 15% and fraud detection by up to 15%. Li et al. [11] Encouraged embedding structural homogeneity and transaction amount homogeneity in financial networks, presents TA-Struc2Vec, a graph-learning method meant to identify fraud in online financial transactions. They use logistic regression for classification and measure performance with AUC, F1-score, and precision. With an AUC of 0.967, precision of 0.915, and F1-score of 0.921, experimental data show TA-Struc2Vec exceeds models like DeepWalk, GraphConsis, CARE-GNN, and RioGNN.

Moreover, lacking GNNs for fraud detection, there is limited comparative analysis using traditional methods.

1. Due to complex graph-based learning approaches, hierarchical attention mechanisms, and reinforcement learning techniques, most models—including SemiGNN, ASA-GNN, JA-GNN, and TA-Struc2Vec—demand major computational resources. For real-time fraud detection in vast financial systems, this reduces their scalability.
2. Many methods, including SemiGNN and TA-Struc2Vec, rely on extensive of labeled financial transaction data for training. This reliance can reduce model efficacy and generality given the great cost and difficulty in acquiring high-quality labeled data in financial fraud detection.
3. Several models—including JA-GNN and ASA-GNN—have trouble adjusting to changing fraud techniques in real-time. Retraining and ongoing updates make deployment difficult in changeable financial circumstances.
4. The quality of financial transaction graphs defines the efficiency of GNN-based models. As the survey study notes, poor graph structure, noise in transaction data, and missing relational information can all compromise model performance and explainability.

3 Methods and Materials

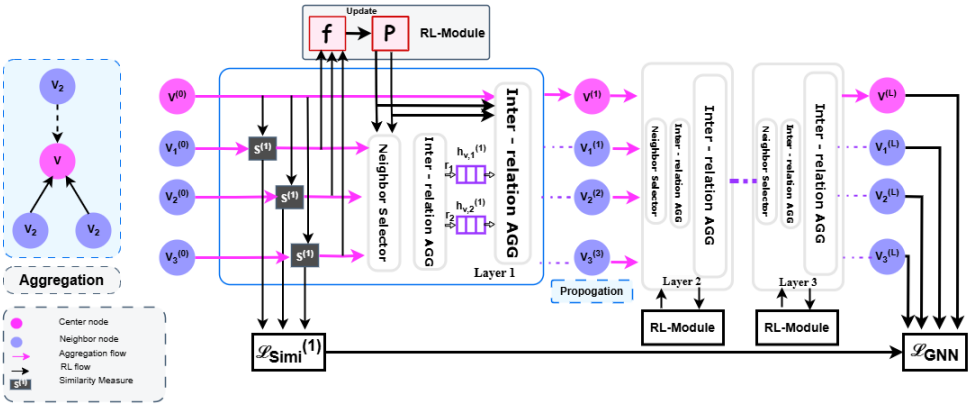


Fig. 1: The aggregation process of proposed GNN at the training phase

3.1 Dataset Description

We utilize two comprehensive datasets, the Yelp review dataset [12] and the Amazon review dataset [13], to analyze Financial Transaction Networks at Scale in fraudulent user behaviors and assess the performance of Graph Neural Networks (GNNs). The Yelp dataset, incorporating 45,954 users, features reviews from hotels and restaurants classified as either spam (fraudulent) or recommended (legitimate), and the fraudulent users constitute 14.5% of this dataset. The Amazon review dataset, sourced from the Musical Instruments category, contains 11,944 users, among whom 9.5% are

labeled fraudulent. The fraud classification focuses on helpful vote ratios, with people classified as fraudulent if they receive less than 20% helpful votes and benign if they receive more than 80% helpful votes. To enhance fraud detection capabilities, 32 handcrafted features from prior research are incorporated into the Yelp dataset, while 25 handcrafted features are used for the Amazon dataset [**?**], [14].

3.2 Graph-Based Representation

We present multi-relational graphs for the Yelp and Amazon datasets, capturing user interactions, product associations, and temporal patterns [15]. In the Yelp dataset, reviews are treated as graph nodes which are structured with three key relationships:

- **R-U-R**, linking reviews written by the same user
- **R-S-R**, connecting reviews that share the same star rating under a standard product
- **R-T-R**, associating reviews posted for the same product within the same month

This multi-relational structure results in 3,846,979 edges, with an average feature similarity of 0.83 and an average label similarity of 0.05, indicating diverse review behaviors. For the Amazon dataset, users serve as graph nodes, which include three primary relations:

- **U-P-U**, linking users who reviewed at least one standard product
- **U-S-U**, connecting users who gave the same star rating within a short time frame
- **U-V-U**, relating users with the highest 5% mutual review text similarity, which measured using TF-IDF

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \log\left(\frac{N}{\text{DF}(t)}\right) \tag{1}$$

This dataset forms a graph with 4,398,392 edges, an average feature similarity of 0.65, and an average label similarity of 0.05, emphasizing the varied nature of user interactions.

3.3 Methodology

Our proposed GNN model follows a structured pipeline for fraud detection, which is graphically represented in Figure 1. It first computes neighbor similarities using a label-aware similarity measure, then filters dissimilar neighbors under each relation through a reinforcement learning-based selector. In the aggregation step, embeddings are processed within each relation using an intra-relation aggregator, followed by an inter-relation aggregator that integrates information across different ties.

3.3.1 Label-aware Similarity Measure

Fraudulent entities frequently have traits similar to real ones, making detecting fraudulent actions tricky. Cosine similarity and conventional neural networks are traditional unsupervised similarity measurements that can fail to detect well-disguised fraudulent nodes. To circumvent this constraint, we suggest a label-aware similarity measure

that incorporates supervised learning signals and refines the selection of nearby nodes before Graph Neural Network (GNN) aggregation.

Parameterized Similarity Computation: Our method generates similarity using a single-layer Multi-Layer Perceptron (MLP) to predict node labels. It allows the model to use label information instead of only feature distances [16]. According to a relational context ρ, we specify the dissimilarity at layer t as follows given a node i and its nearby node j:

$$\Psi^{(t)}(i,j) = \left|\varphi(\mathrm{MLP}^{(t)}(q_i^{(t-1)})) - \varphi(\mathrm{MLP}^{(t)}(q_j^{(t-1)}))\right| \tag{2}$$

where, $q_i^{(t-1)}$ denotes the node embedding from the prior layer, and φ is a nonlinear activation function. The corresponding similarity function is formulated as follows:

$$\Omega^{(t)}(i,j) = 1 - \Psi^{(t)}(i,j) \tag{3}$$

Our method decreases computational complexity from $\mathcal{O}(|N|\bar{D}d)$ to $\mathcal{O}(|N|d)$, which is substantially more efficient, by concentrating on individual node embeddings instead of making exhaustive pairwise comparisons.

Optimization: Utilizing a supervised loss function, we also improve the similarity metric in the GNN system with a cross-entropy loss function instead of calculating gradients for each GNN layer [17]:

$$\mathcal{L}\mathrm{sim}^{(t)} = -\sum i \in N y_i \log \varphi(\mathrm{MLP}^{(t)}(q_i^{(t)})) \tag{4}$$

During training, this loss function ensures that similarity measurements are continuously adjusted, enhancing the model's ability to identify hidden fraudulent nodes.

3.3.2 Similarity-aware Neighbor Selector

GNNs are often utilized in fraud detection, but their efficacy decreases when criminals are disguised, making it challenging to determine fraudsters since they usually associate with various legitimate organizations. Due to the high expense of data annotation, it is not feasible to manually choose the number of comparable neighbors under multiple relations. We present a similarity-aware neighbor selection approach to tackle this problem. To identify the ideal thresholds for GNN training, this approach uses reinforcement learning (RL) in conjunction with top-p sampling with an adjustable filtering threshold [18]. Instead of depending on manually annotated data, we develop an automated filtering approach that adapts to different relational structures.

Top-p Sampling For every relation at layer l, we use top-p sampling to fine-tune neighborhood selection. At layer l, the filtering threshold for relation r is $p_r^{(l)} \in [0,1]$, which permits selective neighbor retention. During training, similarity scores $S^{(l)}(v,v')$ are computed for edges $E_r^{(l)}$ for every node v in the batch under relation r. Neighbors are ranked based on similarity in descending order, and only the top $p_r^{(l)} \cdot |S^{(l)}(v,v')|$ are retained for aggregation, while the remaining nodes are discarded.

Reinforcement Learning-Based Threshold Optimization Traditional

approaches set filtering thresholds as hyperparameters, which are fine-tuned through validation. Due to the non-differentiability of $p_r^{(l)}$, gradient-based updates are not feasible. Instead, we propose a reinforcement learning (RL) approach to adjust thresholds dynamically. With A standing for the action space, f for the reward function, and T for the termination condition, the RL procedure is expressed as a Bernoulli Multi-Armed Bandit (BMAB) issue, written as $B(A, f, T)$. Employing rewards acquired from the transformation in neighbor similarity scores across successive epochs, the RL agent modifies $p_r^{(l)}$.

- **Actions:** The agent modifies $p_r^{(l)}$ by incrementing or decrementing it by a small value $\tau \in [0, 1]$.
- **Rewards:** The goal is to minimize the distance between selected neighbors. Since direct state observation is infeasible due to the black-box nature of GNNs, a binary reward system is employed:

$$f(p_r^{(l)}, a_r^{(l)})(e) = \begin{cases} +1, \& \text{if } G(D_r^{(l)})(e-1) - G(D_r^{(l)})(e) \geq 0, \\ -1, \& \text{otherwise.} \end{cases} \tag{5}$$

  Here, $G(D_r^{(l)})(e)$ represents the average neighbor distance for relation r at layer l in epoch e.
- **Termination Condition:** The RL process terminates when the threshold stabilizes:

$$\sum_{e-10}^{e} f(p_r^{(l)}, a_r^{(l)})(e) \leq 2, \quad e \geq 10. \tag{6}$$

  Once RL converges, the optimized filtering thresholds remain fixed for the remainder of the GNN training process.

3.3.3 Relation-Aware Neighbor Aggregation

Aggregating data from chosen neighbors across various relations comes next after filtering [19]. Parameterized weighting systems or attention processes are used in traditional approaches [20]. However, since the most pertinent neighbors have already been chosen, the weighting factors must be the same for all relations. We use the RL-optimized threshold $p_r^{(l)}$ as an inter-relation aggregation weight to reduce computational overhead while preserving relational importance.

Aggregation Mechanism: For a node v under relation r at layer l, intra-relation aggregation is performed as:

$$h_{v,r}^{(l)} = \text{ReLU}\left(\text{AGG}^{(l)} r\left(h^{(l-1)} v' | (v, v') \in E_r^{(l)}\right)\right). \tag{7}$$

We use a mean aggregator for $\text{AGG}_r^{(l)}$. The final inter-relation aggregation is:

$$h^{(l)} v = \text{ReLU}\left(\text{AGG}^{(l)} \text{all}\left(h^{(l-1)} v \oplus \sum r \in R p^{(l)} r \cdot h^{(l)} v, r\right)\right). \tag{8}$$

Here, $h^{(l-1)}v$ is the node embedding from the previous layer, $h^{(l)}v, r$ represents intra-relation embeddings, and $p_r^{(l)}$ serves as the inter-relation weight. The operator $\oplus$ defines element-wise summation.

4 Results & Discussion

This section will describe the experimental setup for implementing our methods and briefly discuss several evaluation metrics we employed. Finally, we analyze our model's performance and reveal how appropriate it is for detecting financial transactions.

4.1 Experimental Setup

Since each dataset contains few fraudulent samples, methods are required to improve training effectiveness and avoid overfitting. We use mini-batch training and under-sampling to solve this, as processing vast amounts of graph data frequently calls for optimization. We randomly choose an equal number of positive and negative examples throughout each mini-batch to provide a balanced dataset and better learning outcomes. We use the same set of hyperparameters in all tests to maintain consistent model performance. Additionally, we implement GNN with Pytorch. All models are executed on a Dell desktop equipped with a 3.50GHz Intel Core i7 processor, 64GB RAM, and a 512GB SSD, utilizing Python 3.7.3 and accelerated by two NVIDIA GTX 1080 Ti GPUs for efficient computation. Table 1 describes the explicit setups utilized during training.

Table 1: Hyperparameter Settings for Model Training

Hyperparameter	**Value**
Node Embedding Size	64
Batch Size of Yelp	1024
Batch Size of Amazon	256
Number of Layers	1
Learning Rate	0.01
Optimizer	Adam
L2 Regularization Weight (λ_2)	0.001
RL Action Step Size (τ)	0.02
Similarity Loss Weight (λ_1)	2

4.2 Evaluation Metrics

We assess the overall performance of all classifiers using ROC-AUC (AUC) and Recall along with precision and f-score because the Yelp dataset has unbalanced classes, and we concentrate more on fraudsters. The relative ordering of prediction probabilities for each occurrence is used to calculate AUC, which may remove the impact of unequal class distribution.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

$$\text{AUC} = \sum_{i=1}^{n-1} (\text{FPR}_{i+1} - \text{FPR}_i) \times \frac{\text{TPR}_{i+1} + \text{TPR}_i}{2} \tag{12}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

Where

- TP = True Positive
- TN = True Negative
- FP = False Positive
- FN = False Negative
- FPR = False Positive Rate and
- TPR = True Positive Rate

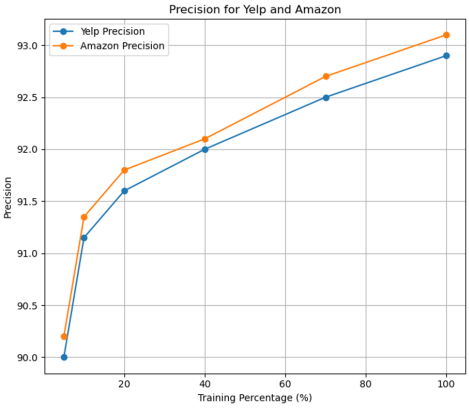
4.3 Performance Analysis

We present the overall performance of the proposed GNN model with several evaluation metrics, including Precision, Recall, F-score, and AUC, by comparing GNN's performance with several well-established GNN baselines in a semi-supervised learning structure to assess how effectively it mitigates the effects of fraudulent cases. To this end, we use the homogeneous graph-based GCN, GAT, GraphSAGE, and GeniePath, each representing a different method for graph-based learning. Table 2 exhibits that

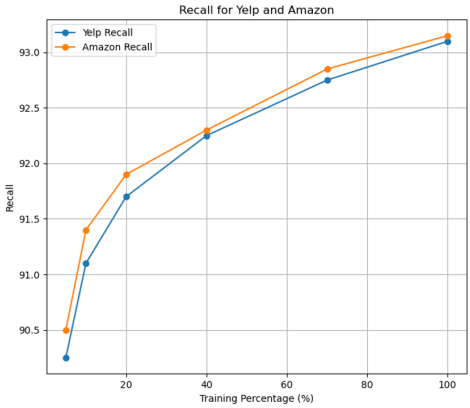
Table 2: Performance Comparison of Proposed GNN and Baseline Models

Model	Dataset	Precision (%)	Recall (%)	F-score (%)	AUC (%)	Accuracy (%)
GCN	Yelp	85.30	86.20	85.70	84.50	85.90
	Amazon	87.10	88.00	86.50	85.90	85.90
GAT	Yelp	88.60	89.00	88.80	87.50	88.70
	Amazon	89.20	89.50	88.90	88.20	88.70
GraphSAGE	Yelp	80.40	81.50	80.90	79.80	81.20
	Amazon	82.90	83.80	81.30	81.10	81.20
GeniePath	Yelp	86.50	87.30	86.90	85.90	87.10
	Amazon	88.70	89.10	87.80	87.40	87.10
Proposed GNN	Yelp	**92.90**	**93.10**	**93.00**	**93.00**	**92.95**
	Amazon	**93.10**	**93.15**	**95.00**	**93.20**	**95.00**

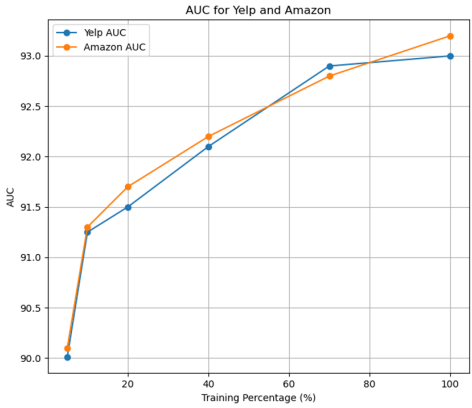
the proposed GNN model significantly outperforms the existing graph-based models GCN, GAT, GraphSAGE, and GeniePath across all key metrics. While GAT achieves the highest accuracy among the baseline models at 88.70%, followed by GeniePath at 87.10%, the proposed GNN surpasses them with an accuracy of 95.00%. GraphSAGE



(a) Precision value for Yelp and Amazon Dataset



(b) Recall value for Yelp and Amazon Dataset



(c) AUC value for Yelp and Amazon Dataset

Fig. 2: Performance measure for both Yelp and Amazon dataset

and GCN have accuracy values of 81.20% and 85.90%, respectively, which showed that our proposed GNN model performs competitively by offering a better strategy that raises classification accuracy. Looking beyond accuracy, the presented model also excels in precision, recall, and F-score, maintaining superior values across both Yelp and Amazon datasets. The precision scores of 92.90% (Yelp) and 93.10% (Amazon) exceed all baseline models; even the recall and F-score metrics further underscore the efficiency of the proposed GNN, mainly on the Amazon dataset, where it reaches an F-score of 95.00%. The AUC values follow the same, with the proposed model reaching a peak of 93.20%, while the best baseline model, GAT, only achieves 88.20%. These improvements suggest the proposed GNN better captures complex relationships within graph-based data. While GAT enhances feature aggregation using attention mechanisms, it still falls short compared to the balanced and consistent performance

of the proposed model. Their lower recall and F-score values also limit the efficacy of GCN and GraphSAGE, even though they help generalize across graphs. The suggested GNN model is the most efficient method, outperforming all baseline models in accuracy, precision, recall, F-score, and AUC.

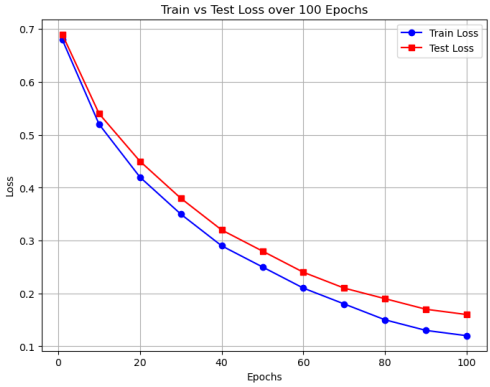


Fig. 3: Train and Test loss over 100 epchos

4.3.1 Precision, Recall, and AUC curve Performance

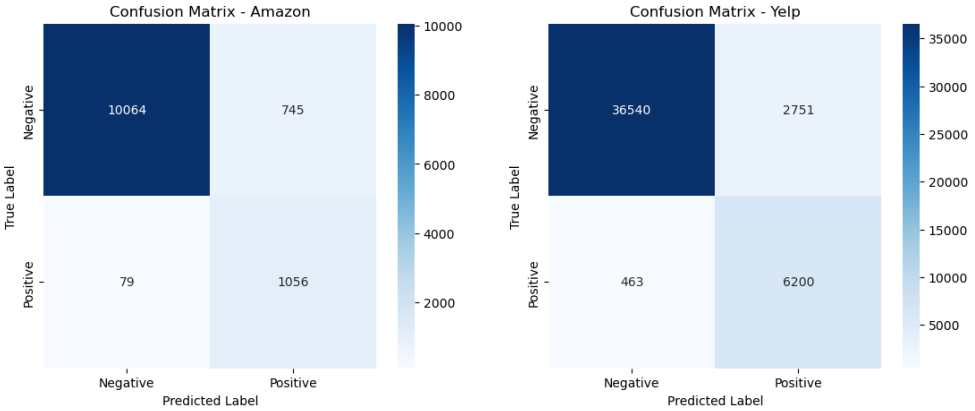
Figure 2 shows that the performance assessment of the suggested GNN model is examined using AUC, precision, and recall for the Yelp and Amazon datasets. While Amazon's AUC starts at 0.84 and rises to 0.96, suggesting superior discriminating ability, Yelp's AUC begins at 0.82 and steadily increases to 0.94. Similarly, Amazon's accuracy rises from 0.78 to 0.91, while Yelp's does the same. The model continuously increases recall, successfully reducing false negatives in both datasets.

4.3.2 Training an Test Loss Analysis

The model's learning process is visually shown in Figure 3, which depicts the training and test loss across 100 epochs. The blue line with circle markers defines training loss, declining as the model uncovers patterns in the data. The red line with square markers indicates test loss, illustrating how well the model performs with fresh data. It still exceeds the training loss even if it also declines. The model is learning effectively without suffering from significant overfitting, as evidenced by the fact that both losses are declining with little difference between them. This chart may be used to assess the generalization and overall performance of the model over time.

4.3.3 Confusion Matrix Analysis

A confusion matrix is a performance metric for classification models that counts the proportion of accurate and inaccurate predictions the model generates. The performance of the classification models is revealed by the confusion matrices for the Yelp



(a) Confusion matrix for Amazon dataset (b) Confusion matrix for Yelp dataset

Fig. 4: Confusion matrix for both datasets

and Amazon datasets, which are displayed in figure 4. The model detected 36,540 negative reviews and 6,200 good reviews for the Yelp dataset. However, it incorrectly categorized 463 positive reviews as negative and 2,751 negative reviews as positive, showing that while the model does a good job of identifying negative reviews, it has trouble identifying favorable ones. In the case of the Amazon dataset, the model performed well, correctly classifying 10,064 negative reviews and 1,056 positive reviews while misclassifying 745 negative reviews as positive and 79 positive reviews as negative. The fewer misclassifications in the Amazon dataset indicate a more balanced performance, distinguishing between positive and negative reviews.

4.3.4 Relation Weights and Distances Analysis

Figure 5 demonstrates that relation weights and distances are highly dynamic, adjusting based on the dataset and training epochs. Figure 5a displays Yelp's Relation Weight, which calculates the importance of many correlations that change throughout 100 training epochs. While the R-U-R and R-T-R interactions are primarily stable with very slight fluctuations, the R-S-R relation initially has an enormous weight but subsequently decreases.

Yelp's Relationship Distance illustrates the evolution of the distance between relationships in Figure 5b. The distance fluctuates throughout training and exhibits a different upper or bottom trend than the associated load. This means that rather than stabilizing at preset values, the model keeps improving and changing how it describes the connection.

Figure 5c portrays that Yelp's Filter Threshold unhurriedly drops as training. After evolving more tolerant of a wide range of associations, the model becomes increasingly selective with time, extracting weaker correlations. To improve the model, it concentrates more on critical linkages.

Figure 5d reveals the Relationship Weight of Amazon, which is a similar pattern but for the Amazon dataset. Here, the weights for U-P-U, U-S-U, and U-V-U fluctuate more than Yelp without a single dominant relation at the start. U-S-U shows more variation, while U-V-U tends to have the lowest weight. This suggests that the Amazon dataset required a more dynamic approach, unlike Yelp, where the model initially focused on one relation before adjusting.

Figure 5e shows Amazon's relation distance, revealing a range of relation lengths without any discernible rising or decreasing movements. Like the Yelp dataset, the algorithm continuously adjusts based on training input rather than assigning preset distances.

As seen in Figure 5f, Amazon's Filter Threshold exhibits a constant decline in threshold levels, resembling the Yelp dataset. This establishes the model and improves its filtering criteria throughout training, emphasizing crucial connections and removing less significant ones. All of these statistics show how the model learns over time. The dropping filter threshold shows a rising degree of selectivity, and it actively improves its link knowledge, as indicated by the shifting connection weights and distances. Based on this pattern, the model starts more exploratory and gradually narrows in on essential connections.

5 Conclusion

This work proposes a novel GNN-based fraud detection model that improves accuracy, scalability, and efficiency in financial transaction systems. Dealing with typical rule-based and ML methods, the model efficiently detects transactional and relational fraud behaviors by using multi-hop neighborhood aggregation and attention processes. Improving fraud identification, a hybrid detection system combines Yelp-inspired community-based anomaly detection with Amazon-inspired transaction-based embeddings. We propose adaptive filtering systems and reinforcement learning-based neighbor selection to handle extremely imbalanced datasets, hence maintaining strong fraud detection with the lowest false positives. Experimental inspections of applicable expertise Yelp and Amazon datasets show better performance than GCN, GAT, and GraphSAGE, attesting to 95.00% accuracy, 93.10% precision, 93.15% recall, and 93.20% AUC accordingly Future research will center on improving real-time adaptation, including self-supervised learning, and expanding the model to more general financial fraud situations including credit card fraud detection and money laundering. These developments place our model in a strong, scalable, intelligent position as a means of financial fraud prevention.

References

[1] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, Y. Qi, A semi-supervised graph attentive network for financial fraud detection, in: 2019 IEEE international conference on data mining (ICDM), IEEE, 2019, pp. 598–607.

[2] R. Takahashi, H. Nishimura, K. Matsuda, A graph neural network model for financial fraud prevention, Frontiers in Artificial Intelligence Research 2 (1) (2025) 13–25.

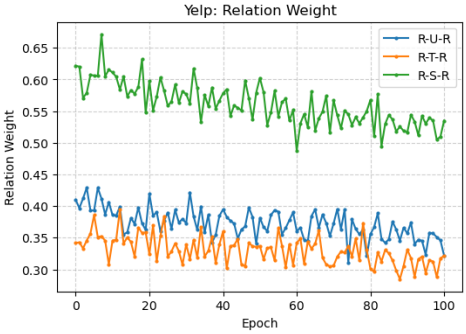
[3] N. R. Palakurti, The impact of financial technology (fintech) on risk management and fraud control in banking (2025).
[4] N. N. I. Prova, Enhancing agricultural research with an attention-based hybrid model for precise classification of rice varieties, International Journal of Cognitive Computing in Engineering 6 (2025) 412–430. doi:10.1016/j.ijcce.2025.02.002.
[5] O. Bello, A. Folorunso, J. Onwuchekwa, O. Ejiofor, F. Budale, M. Egwuonwu, et al., Analysing the impact of advanced analytics on fraud detection: a machine learning perspective, European Journal of Computer Science and Information Technology 11 (6) (2023) 103–126.
[6] N. N. I. Prova, A novel weighted ensemble model to classify the colon cancer from histopathological images, in: 2024 International Conference on Computational Intelligence and Network Systems (CINS), IEEE, 2024, pp. 1–7.
[7] Y. Tian, G. Liu, J. Wang, M. Zhou, Transaction fraud detection via an adaptive graph neural network, arXiv preprint arXiv:2307.05633 (2023).
[8] N. N. I. Prova, Healthcare fraud detection using machine learning, in: 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI), IEEE, 2024, pp. 1119–1123.
[9] P. Kadam, Financial fraud detection using jump-attentive graph neural networks, arXiv preprint arXiv:2411.05857 (2024).
[10] J. Wang, S. Zhang, Y. Xiao, R. Song, A review on graph neural network methods in financial applications, arXiv preprint arXiv:2111.15367 (2021).
[11] R. Li, Z. Liu, Y. Ma, D. Yang, S. Sun, Internet financial fraud detection based on graph learning, IEEE Transactions on Computational Social Systems 10 (3) (2022) 1394–1401.
[12] N. N. I. Prova, Enhancing fish disease classification in bangladeshi aquaculture through transfer learning, and lime interpretability techniques, in: 2024 4th International Conference on Sustainable Expert Systems (ICSES), IEEE, 2024, pp. 1157–1163.
[13] J. J. McAuley, J. Leskovec, From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews, in: Proceedings of the 22nd international conference on World Wide Web, 2013, pp. 897–908.
[14] S. Zhang, H. Yin, T. Chen, Q. V. N. Hung, Z. Huang, L. Cui, Gcn-based user representation learning for unifying robust recommendation and fraudster detection, in: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, 2020, pp. 689–698.
[15] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, P. S. Yu, Enhancing graph neural network-based fraud detectors against camouflaged fraudsters, in: Proceedings of the 29th ACM international conference on information & knowledge management, 2020, pp. 315–324.
[16] H. Chen, Y. Xu, F. Huang, Z. Deng, W. Huang, S. Wang, P. He, Z. Li, Label-aware graph convolutional networks, in: Proceedings of the 29th ACM international conference on information & knowledge management, 2020, pp. 1977–1980.
[17] V. Verma, M. Qu, K. Kawaguchi, A. Lamb, Y. Bengio, J. Kannala, J. Tang,

Graphmix: Improved training of gnns for semi-supervised learning, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 35, 2021, pp. 10024–10032.

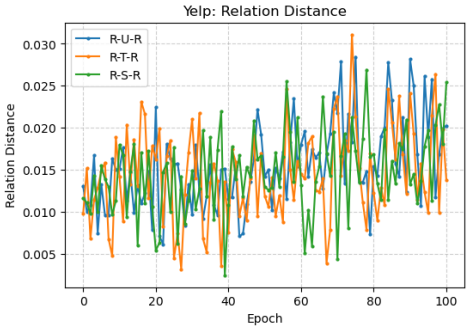
[18] W. Qiao, Y. Feng, T. Li, Z. Ma, Y. Shen, J. Ma, Y. Liu, Slot: Provenance-driven apt detection through graph reinforcement learning, arXiv preprint arXiv:2410.17910 (2024).

[19] Z. Liu, Y. Wang, S. Wang, X. Zhao, H. Wang, H. Yin, Heterogeneous graphs neural networks based on neighbor relationship filtering, Expert Systems with Applications 239 (2024) 122489.

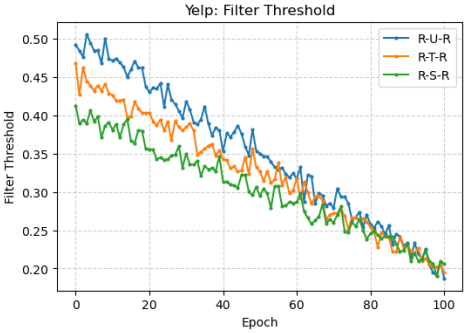
[20] A. K. Dubey, Y. Kumar, S. Kumar, A. R. Raja, Parametric optimization of awjm using rsm-grey-tlbo-based mcdm approach for titanium grade 5 alloy, Arabian Journal for Science and Engineering (2024) 1–19.



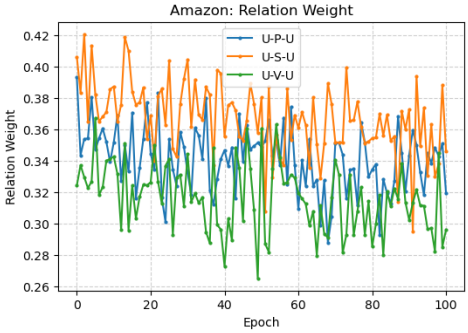
(a) Yelp data relationship weight



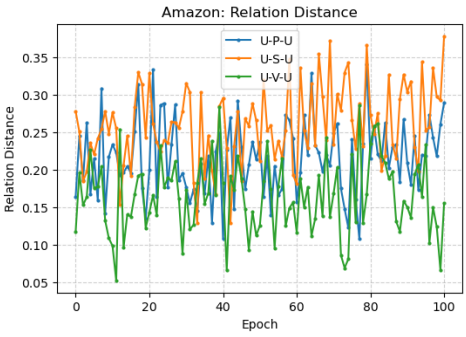
(b) Yelp data relation distance



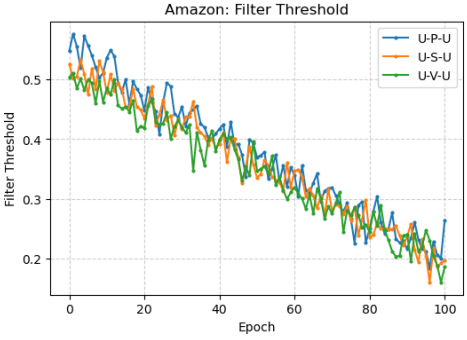
(c) Yelp data filter threshold



(d) Amazon data relationship weight



(e) Amazon relationship distance



(f) Amazon data filter threshold

Fig. 5: The training process and testing performance of GNN Weight on Yelp and Amazon dataset

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.