Leveraging Graph-Based Learning for Credit Card Fraud Detection: A comparative study of traditional, deep learning and graph-based approaches.

DATS 6501 – Data Science Capstone

Sunisha Harish,
Chirag Lakhanpal

supervised by
Amir Jafari

Abstract

Credit card fraud results in staggering financial losses amounting to billions of dollars annually, impacting both merchants and consumers. Contemporary solutions heavily rely on machine learning (ML) and deep learning (DL) methods to handle such tasks. While these methods have been effective in many aspects of fraud detection, they may not always be sufficient for credit card fraud detection as they aren't adaptable to detect complex relationships when it comes to transaction In the context of fraud detection, the ability of Graph Neural Networks (GNN's) to aggregate information contained within the local neighbourhood of a transaction enables them to identify larger patterns that may be missed by just looking at a single transaction. In this research, we conduct a thorough analysis to evaluate the effectiveness of GNNs in improving fraud detection over traditional Machine Learning and Deep Learning methods. We first build an heterogenous graph architecture with the source, transaction, and destination as our nodes. Next, we leverage Relational Graph Convolutional Network to learn the representations of nodes in our graph and perform node classification on the transaction node. Our experimental results demonstrate that GNN outperforms traditional and deep learning methods.

Table of contents

1 Introduction

With the growing upsurge in credit card payments, credit card fraud has become the most common form of identity theft. Credit card fraud occurs when someone uses another person's credit card or credit card information to buy something or access an account without permission. According to Federal Trade Commission Data [1] a loss of more than $10 billion to fraud was reported by the consumers in 2023. This data signifies a 14% increase over losses documented in 2022. There are different types of credit card frauds that can occur [2]. One is the fraud committed over phone or online where the fraudster has the card details but not the card. Another way is credit card skimming, where a fake or cloned card id created using someone else's information. Fraudsters can also get a hold of a customer's personal information and use them to access their credit card account or apply for a new card in their name. Fraud can also occur through a lost or stolen credit card. Although the ratio of fraud transactions is very low, credit card fraud undermines consumer trust, leading to reduced spending and increased costs for businesses, ultimately hindering economic growth and stability. It also imposes financial burdens on individuals and businesses, disrupts operations, and entails legal and regulatory complexities.

In this project, we present our solution to tackle the problem of credit card fraud using machine learning, deep learning, and graph neural networks. We are using the Tabformer dataset provided by IBM [7] to demonstrate this workflow. The TabFormer dataset is a synthetic credit card transaction dataset, consisting of around 24 million unique transactions and around 0.1% of fraudulent samples. The rest of the paper is organized as follows: Section 2 covers the background and related work done on credit card fraud detection. Section 3 describes the methodology used. Section 4 covers the experimentation and results. The last section concludes our work with the scope for future work.

2 Problem Statement

Credit card fraud presents a substantial challenge, resulting in significant financial losses for businesses and individuals annually. Research aims to explore the potential of Graph Neural Networks (GNNs) in enhancing fraud detection by leveraging their capacity to analyze transaction patterns. Detecting fraud is complicated by imbalanced data and evolving tactics employed by fraudsters, necessitating ongoing updates to detection systems. The study endeavours to conduct a comprehensive analysis, comparing the performance of GNNs with traditional machine learning and deep learning methods to advance fraud detection capabilities.

3 Related Work

Fraud detection has been a great motivation for researchers to find a solution. Numerous methods have been proposed and tested to detect and prevent fraud. Their primary function is to utilize historical data to automatically detect fraudulent transactions. While they excel at identifying established fraud patterns, they struggle when confronted with unfamiliar types of fraud. Our literature review focused on researching the work performed on credit card fraud detection using traditional, deep learning and graph neural networks.

Methods using machine learning models have been compared, these models have achieved much success [3]. Classical algorithms such as, Support Vector Machines (SVM), Decision Tree (DT), Logistic Regression (LR) and Random Forest (RF) have proven useful. Random Forest and AdaBoost Algorithm were compared [4] after applying SMOTE to balance the dataset and the RF algorithm achieved better precision, recall and F1-score. Paper [5] benchmarked several machine learning models along with MLP Classifier on a real-world credit card fraud dataset and found that the Random Forest classifier performed best after applying oversampling and feature selection techniques to the highly imbalanced data. These studies proved that classical models perform very well when data imbalance is handled using proper techniques. Classical models tend to bias towards the majority class and overfit the minority class and hence perform poorly on unbalanced data. Ensemble techniques such as XGBoost demonstrate a superior ability to effectively handle imbalanced data distributions compared to traditional machine learning models, as evidenced by their considerably better performance metrics reported in the paper [6]. XGBoost uses gradient boosting which builds ensemble models in a sequential manner and handles weighted instances, allowing it to better adapt to the imbalanced class distributions.

In recent years, deep learning approaches have also been extensively explored for credit card fraud detection, in addition to traditional machine learning models. Deep learning is an actively researched area that shows potential in enhancing fraud detection performance, especially with larger future datasets. Paper [8] compared machine learning algorithms along with Bi-LSTM and Bi-GRU and much better results were achieved with the deep learning methods. Methods such as CNN and LSTM have been widely used for image classification and Natural Language Processing respectively due to their ability to handle massive datasets. CNN and LSTM have proved to be quite efficient in handling large datasets and have also delivered promising results for fraud detection [9]. LSTM as a sequence learner has been compared with Random Forest (a static learner) on online and offline transactions and their behaviour has been studied [10].

Hybrid approaches combining deep learning with other techniques have been explored to counter class imbalance. CCFD-Net used a hybrid architecture and the residual neural network (Res-Net) [11]. The study showed that hybrid architecture produces much better results and can be used for numerical data classification. Traditional and deep learning models have also been used to create ensemble methods where the results of individual classifiers have been combined to get the final classifier results [12] [13].

However, the research in using deep learning models for credit card classification is not as extensive enough as traditional models. Moreover, traditional models like XGBoost are much efficient at handling imbalanced datasets than deep learning architectures.

Classical models may produce great results, but they require extensive feature engineering, and it is difficult to model the relationships between different entities to capture complex fraud patterns. They focus on direct relationships potentially overlooking fraudulent patterns embedded with broader complexities.

Graph neural networks (GNNs) have gained significant traction as a powerful machine learning framework for modelling data with underlying graph/network structures. A GNN is an effective framework that can learn graph representations by modelling the relationships of non-Euclidean graph data. The inherently tabular nature of transactional data can be effectively represented as a graph structure by constructing nodes that encapsulate the various entities involved, such as credit card accounts, merchants, and transactions themselves. The relationships and interactions between these entities can then be captured by introducing edges that connect the corresponding nodes.

Graph neural network methods such as GCN and GAT (Graph Attention Networks) have been applied on transaction data to detect fraud [14]. Graph Neural Networks improves the accuracy of fraud predictions by using more relevant features information from the network. Some researchers have constructed a graph using each transaction record as a node and defining a set of logical propositions as conditions to create an edge between the transactions [15] [16]. Paper [15] uses this concept and defined the logical propositions by connecting transactions that have the same MAC and take place at the same time interval. Using homogenous graphs like this have proven to perform better [14] but however they have certain limitations: (1) The applicability is not ideal in real world scenarios with a lot of missing values in data. (2) The logical propositions defined on one dataset cannot be easily applied to others.

Paper [17] picks centre nodes using a label-balanced sampler, chooses the neighbourhoods of minority and majority class nodes by oversampling and undersampling respectively based on a parameterized distance function, and then aggregates messages from the selected neighbours and relations to obtain final node representations for training. Some other works have tried modelling graphs with transactions as edges and the source and destination as nodes [18]. Edge features have been exploited and used to train the graph neural networks [19]. These methods may lead to loss of rich transactional metadata. Edge features are typically treated as static attributes, lacking the ability to model dynamic or temporal edge properties that evolve over time.

However, these GNN methods have some flaws in terms of applicability for different datasets while using logical propositions and the loss of features when exploited as edge features. The absence of standardized and consistent definitions for credit card transaction datasets across different scenarios poses a challenge, resulting in a lack of robustness and generalizability of the models and techniques developed using these datasets.

4 Solution and Methodology

In this section, we first describe our approach to construct the graph. We then discuss about the RGCN model architecture and how we use it to classify our transactions.

4.1 Graph Construction

In response to the limitations observed in both homogenous graphs, where transactions solely serve as nodes, and in graphs where transactions are edges between source (card ID) and destination (merchant ID) nodes—primarily concerning static attributes and the loss of features—we've implemented a solution: the construction of a heterogeneous graph [21] [23] with learnable parameters. In this graph representation, the focal point is the transaction node, while the source and destination nodes correspond to the card ID and merchant ID, respectively. This approach imbues the transaction node with transaction-specific features, also known as node features, thereby enriching the graph's representational power. The source and destination nodes, i.e., card and merchant IDs, intentionally lack inherent features and are treated as learnable parameters. This strategic design allows for greater flexibility and adaptability within the model. Our classification task centres on predicting the transaction node, thereby making it a node classification problem. This framework not only addresses the shortcomings of existing graph representations but also paves the way for more effective and versatile analysis within complex transactional networks.

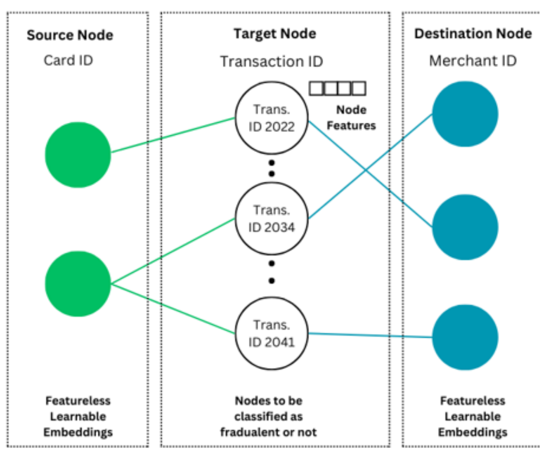

Figure 1: Architecture of our constructed heterogenous graph.

4.2 Message Passing in GNN’s

GNNs are known for their ability to learn structural information. Usually, nodes with similar features or properties are connected to each other (this is true in the social media setting). The GNN exploits this fact and learns how and why specific nodes connect to one other while some do not. To do so, the GNN looks at the neighbourhoods of nodes.

The Neighbourhood $\mathcal{N}_i$ of a node i is defined as the set of nodes j connected to i by an edge. The equation is as follows:

$$\mathcal{N}_{i} = \{j : e_{ij} \in E\} \qquad (1)$$

Once we obtain the transformed messages, we aggregate them. Aggregations can be done using Sum, Mean, Min or Max. The equation for aggregation using Sum is as follows:

$$Sum = \sum_{j \in N_i} W_j \, x_j \qquad (2)$$

After aggregating messages from the neighbouring nodes, the GNN layer needs to update the feature representation of the source node i itself. The updated representation should not only encode the node's own features but also incorporate the information from its neighbours captured through the aggregated messages. This is achieved by combining the source node i 's original feature vector with the aggregated neighbourhood messages.

4.3 Heterogenous RGCN

R-GCNs are an extension of standard Graph Convolutional Networks (GCNs) designed to handle graphs with different types of relationships or edges between nodes (known as relational or heterogeneous graphs) [22]. The core idea is to perform message passing and neighbourhood aggregation while taking into account the different relation types present in the graph. The message-passing equation is as follows:

$$h_i^{(l+1)} = \sigma\left(\sum_{r \in R} \sum_{j \in N_r(i)} W_r^{(l)} h_j^{(l)}\right) \qquad (3)$$

Here: $h_i^{(l+1)}$is the feature representation of node i at the l-th+1 layer of the GCN, R is the set of all relation types (edge types) in the heterogeneous graph, $N_r(i)$ is the set of neighbor nodes of node i under the relation r. $W_r^{(l)}$ is a learnable weight matrix specific to the relation type r at layer l. σ is a non-linear activation function like ReLU.

The message passing happens in two key steps:

i)Per-edge-type Message Computation and Aggregation: For each relation type $r \in$ R, the model computes messages by applying the relation-specific weight $W_r^{(l)}$to the representations $h_i^{(l+1)}$ of all neighbouring nodes $j \in N_r(i)$. These weighted messages are then aggregated (summed up) to capture the neighbourhood information for node i under the relation r.

ii)Type-wise Reduction: After computing and aggregating messages for each relation type r independently, the results are combined (reduced) by taking a sum across all relation types.

So, the overall computation involves a) Applying relation-specific transformations to neighbour representations. b) Aggregating these transformed messages per relation type. c) Combining the per-relation aggregates into a single representation for the target node.

This allows the R-GCN to effectively propagate and integrate information along different types of edges/relations in the heterogeneous graph during the message passing process. The relation-specific weights $W_r^{(l)}$ enable the model to learn distinct transformation parameters for each edge type, capturing their unique neighborhood interaction patterns.

By performing this type-aware message passing, R-GCN can learn rich node representations that fuse both node features and multi-relational graph structure information present in heterogeneous graphs like credit card transaction networks.

We apply the R-GCN model to the constructed heterogeneous graph to learn representations for classifying the transaction nodes of interest. The R-GCN performs relational message passing by transforming node representations from neighbouring nodes under different relation types (e.g. card-to-transaction, merchant-to-transaction) using relation-specific weight matrices. These transformed neighbour messages are then aggregated through a normalized sum operation into updated node representations. For the target transaction node, its embedding is computed by combining its initial node features with the aggregated multi-relational neighbourhood messages passing through a non-linear activation. This relational message passing, and aggregation process is parallelized across all nodes in the graph.

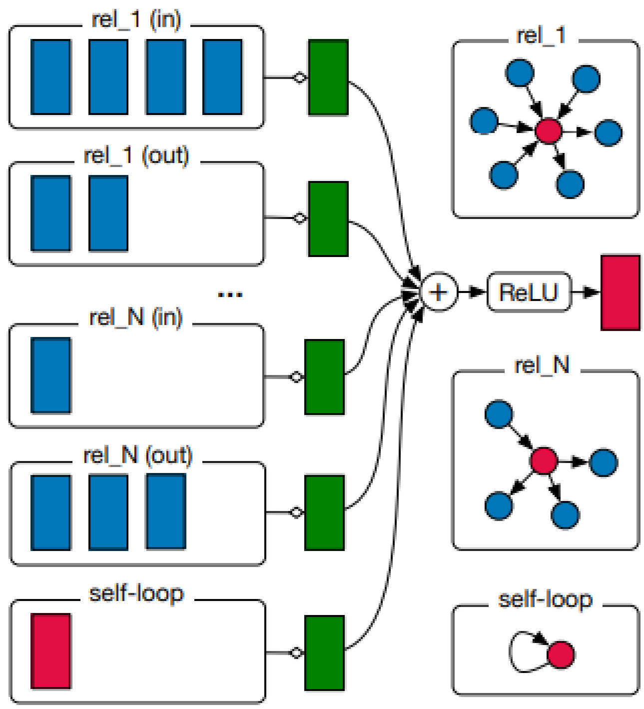

Figure 2: R-GCN model updating the representation of a target node [22].

Figure shows how the R-GCN model updates the representation of a target node (red) by incorporating information from its neighbours (dark blue) across different relation types. Neighbour representations are transformed using relation-specific weights for incoming and outgoing edges. The transformed neighbour messages from all relations are then accumulated through a normalized sum into a single vector (green). This relational summary goes through a non-linear activation to produce the updated node representation. This per-node update is parallelized across the graph with shared parameters [22].

5 Experimentation and results

5.1 Dataset Description

The dataset we have used to conduct our study is a synthetic dataset provided by IBM [7]. The dataset consists of 24 million unique transactions, involving 6,139 unique cards and 100,343 unique merchants. Among these transactions, 29,757 are labelled as fraudulent, accounting for 0.1% of the total transactions.

The highly imbalanced nature of the Tabformer dataset, with a significantly lower proportion of fraudulent transactions compared to legitimate ones, is a common characteristic of real-world fraud detection scenarios. This imbalance poses challenges for anomaly detection models, as they need to effectively identify the rare fraudulent instances while minimizing false positives.

The Tabformer dataset provides a realistic representation of financial transaction data, enabling researchers and practitioners to develop and evaluate anomaly detection models in a controlled environment. The dataset's large scale and imbalanced distribution make it suitable for benchmarking the performance of various anomaly detection techniques, including traditional machine learning models, deep learning architectures, and graph neural networks.

The records in the dataset are as shown below:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	User	Card	Year	Month	Day	Time	Amount	Use Chip	Merchant Name	Merchant City	Merchant State	Zip	MCC	Is Fraud?	Errors?
2	0	0	2019	4	1	4:08	$39.58	Chip Transaction	Kelly Auto Repair	Brandon	FL	33510	7538	No	
3	0	0	2019	4	2	4:12	$36.98	Chip Transaction	Kelly Auto Repair	Brandon	FL	33510	7538	No	
4	0	0	2019	4	4	4:20	$40.36	Chip Transaction	Kelly Auto Repair	Brandon	FL	33510	7538	No	
5	0	0	2019	4	4	10:54	$7.98	Chip Transaction	Walmart	Brandon	FL	33510	5311	No	
6	0	0	2019	4	5	4:46	$49.87	Swipe Transaction	Nissan Service	Brandon	FL	33510	7538	No	
7	0	0	2019	4	5	10:59	$11.27	Chip Transaction	Applebees	Brandon	FL	33511	5812	No	

Figure 3: Transaction records of the IBM Dataset

5.2 Dataset Preprocessing

The Tabformer dataset undergoes several preprocessing steps to ensure data quality and prepare it for the comparative study on anomaly detection models. The preprocessing pipeline is implemented using a custom function which takes the dataset as input along with various parameters to control the preprocessing steps.

Firstly, the function automatically detects columns with only two unique values and encodes them as binary (0 and 1). This step helps in representing binary features effectively. Next, the columns are converted to numeric data type, when possible, to ensure consistent data representation across the dataset. To handle categorical variables, the non-numeric columns are one-hot encoded, converting them into binary vectors. This encoding technique allows the models to process categorical data effectively. Missing values in the dataset are handled by replacing them with the mode (most frequent value) of each column, ensuring that no data points are lost due to missing information.

To reduce the dataset's size while maintaining the class distribution, stratified sampling is applied based on the 'Is Fraud?' column. This step helps in creating a representative sample of both fraudulent and non-fraudulent transactions. The 'Time' column, which contains datetime information, is treated separately, and appropriate datetime features are extracted from it. Data cleaning is performed on specific columns to ensure data consistency. The 'Amount' column undergoes cleaning to remove any special characters or inconsistencies. Additionally, columns that may not be relevant for anomaly detection, such as 'Card', 'User', and 'Merchant Name', are removed from the dataset. Certain columns, including 'Use Chip', 'Merchant City', 'Merchant State', 'Zip', 'MCC', and 'Errors?', are considered as categorical variables and are encoded accordingly. The 'Is Fraud?' column is designated as the target variable for the anomaly detection task.

It is worth noting that normalization and class balancing techniques are not applied in this preprocessing pipeline. The preprocessing steps are carefully designed to clean, transform, and prepare the Tabformer dataset for the subsequent comparative study on anomaly detection models.

5.3 Baselines

The following models have been compared to prove the effectiveness of Heterogenous RGCN.

i) Logistic Regression: A baseline linear classification model commonly used for its simplicity and interpretability, often applied as a benchmark in machine learning tasks.

ii) Random Forest: A versatile ensemble learning method that aggregates the predictions of multiple decision trees, known for its robustness and ability to handle high-dimensional data with complex interactions.

iii) LightGBM: A gradient boosting framework optimized for efficiency and speed, leveraging histogram-based algorithms and leaf-wise tree growth to achieve superior performance on large-scale datasets.

iv) XGBoost: Another gradient boosting library renowned for its scalability and accuracy, employing a regularized objective function and parallelized tree construction to achieve state-of-the-art results in various machine learning competitions.

v) CatBoost: A gradient boosting algorithm designed to handle categorical features efficiently, utilizing novel techniques such as ordered boosting and feature importance calculation to deliver competitive performance with minimal hyperparameter tuning.

vi) CNN: Convolutional Neural Networks, widely used for image classification tasks, leveraging convolutional layers to automatically extract hierarchical features from input data and achieve superior performance in visual recognition tasks.

vii) LSTM: Long Short-Term Memory networks, a type of recurrent neural network (RNN) equipped with memory cells capable of capturing long-range dependencies in sequential data, commonly applied in natural language processing and time series forecasting tasks.

viii) CNN-LSTM: A hybrid neural network architecture combining convolutional and recurrent layers, adept at processing both spatial and temporal information, often utilized in tasks involving sequential data with spatial dependencies, such as video analysis and sensor data processing.

5.4 Implementation

For our study, we have sampled 60% of the original data which is around 15 million due to memory constraints. We split the data into training, validation, and test data sets with the ratio 60:20:20. We have used Scikit-Learn (1.4.1), XGBoost (2.0.3), Pytorch (2.2.0) and DGL (0.9.1) for our study. For our deep learning models, we have used a learning rate of 0.0001 and Adam optimizer with number of epochs set to 300 and a batch size of 512.

For GNN's, we have used a learning rate of 0.01 with hidden dimensions 16 and a weight decay of 5e-4. Adam optimizer is used to optimize the parameters and the number of epochs is set to 1000. The loss function used in both cases is CrossEntropyLoss.

5.5 Evaluation Metrics

Comparing the various methods used and choosing the best model is an important part of developing an effective and reliable system for credit card fraud detection. Since credit card fraud datasets are highly imbalanced, it is important to choose the appropriate evaluation metrics. In this study, we have selected four metrics F1-Score, Precision, Recall and AUCPR as they focus on capturing the true positives predicted.

Before we get into the metrics let's look at the way predictions are classified:

True Positive (TP): Correctly predicted fraudulent transactions.

True Negative (TN): Correctly predicted legitimate transactions.

False Positive (FP): Incorrectly predicted fraudulent transactions.

False Negative (FN): Incorrectly predicted legitimate transactions.

Precision and Recall are obtained by using these values. F1-Score is defined as the harmonic mean of precision and recall.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{4}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{5}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (6)$$

The last metric considered is AUCPR. The AUCPR (Area Under the Precision-Recall Curve) metric is a key evaluation measure in machine learning, especially in scenarios with class imbalance or where correctly identifying positive instances is vital. It quantifies the precision-recall trade-off by computing the area under the precision-recall curve, offering a single value that summarizes a model's performance across various threshold settings. A higher AUCPR indicates better model performance, emphasizing the balance between precision and recall.

5.6 Results

To prove that our approach with Graph Neural Networks can perform better than classical and deep learning models we run a comparison study on using all the models and calculate the evaluation metrics. The results obtained are as follows:

Methods	Precision	Recall	F1-Score	AUCPR
Logistic Regression	0.63	0.13	0.22	0.18
Random Forest	0.97	0.51	0.67	0.76
LightGBM	0.71	0.50	0.58	0.44
CatBoost	**0.96**	0.63	0.76	**0.80**
XGBoost	0.95	0.63	0.76	**0.80**
CNN	0.85	0.45	0.59	0.65
LSTM	0.91	0.45	0.61	0.68
CNN-LSTM	0.82	0.56	0.67	0.78
GNN	0.94	**0.66**	**0.78**	**0.78**

Table 1: Results of different models and evaluation metrics on the dataset

Looking at the performances of our fraud detection model, the classical models Logistic Regression performs the worst among all. This might be because LR is biased more towards the majority class and is unable to handle data imbalance. The performance of XGBoost and CatBoost are almost similar to each other with an F1-Score of 0.76, and they perform the best among the machine learning models. This is in line with our research that XGBoost can handle imbalanced data well.

Coming to the Deep Learning models, CNN and LSTM have performed well compared to LR and LightGBM. Our ensemble method using CNN and LSTM has performed better than CNN and LSTM alone and is also in line with the F1 Score (0.67) of Random Forest model.

Lastly, Graph Neural Networks have outperformed the other models with an F1 Score of 0.78 and a recall rate of 0.66.

In summary GNN has achieved better results compared to classical and deep learning methods when it comes to solving the problem of credit card fraud detection.

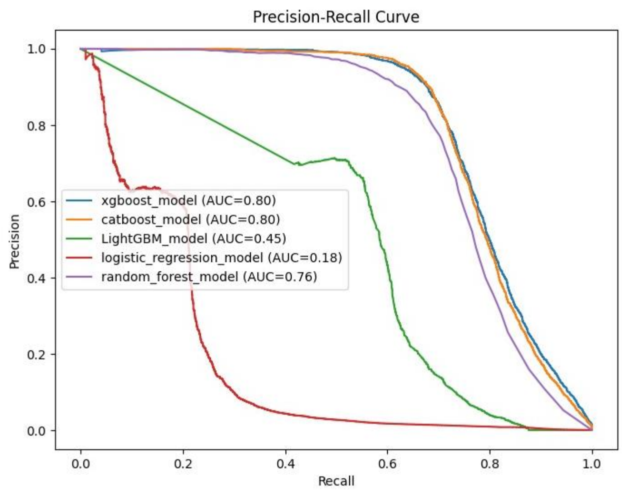

Figure 4: AUCPR curves for classical models

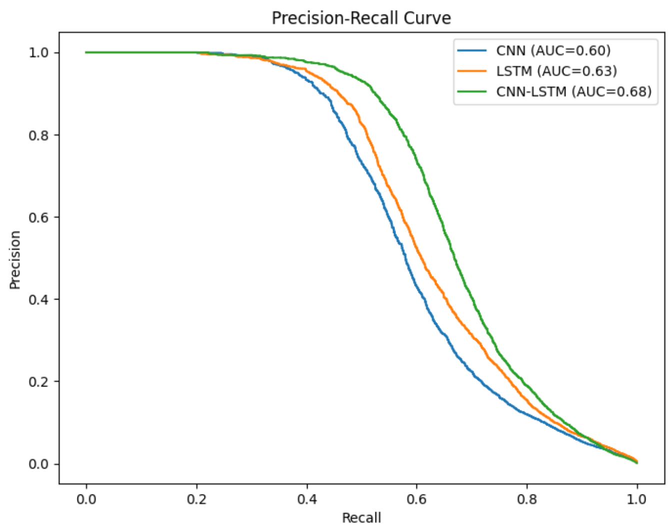

Figure 5: AUCPR curves for deep learning models

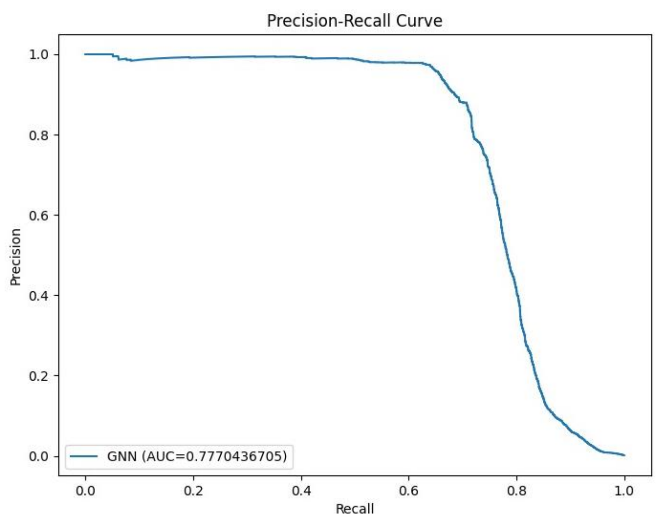

Figure 6: AUCPR curve for RGCN model

6 Conclusion and Future Work

In conclusion, this paper presents an approach to address the limitations of existing graph representations in transactional networks by introducing a heterogeneous graph framework. By structuring the graph with transaction nodes as the focal point and incorporating learnable parameters for card and merchant ID nodes, our model enhances the representation of transactional data and improves classification performance. Through experimentation, we demonstrate the effectiveness of Graph Neural Networks when compared to other methods, showcasing its potential for various applications in fraud detection, recommendation systems, and beyond.

For future work, incorporating sampling strategies and edge weights into our heterogeneous graph framework represents a promising direction offering opportunities to enhance scalability, efficiency, and predictive performance in transactional network analysis and related domains. Further, integrating attention mechanisms into our heterogenous graph to allow nodes to focus on relevant neighbours during message passing can enhance the ability to capture complex relationships.

7 Bibliography

1. Federal Trade Commission (9 February 2024) As Nationwide Fraud Losses Top $10 Billion in 2023, FTC Steps Up Efforts to Protect the Public. https://www.ftc.gov/news-events/news/press-releases/2024/02/nationwide-fraud-losses-top-10-billion-2023-ftc-steps-efforts-protect-public

2. Capital One What is credit card fraud & how do you report it? (6 February 2024) https://www.capitalone.com/learn-grow/privacy-security/what-to-do-about-credit-card-fraud/

3. J. O. Awoyemi, A. O. Adentumbi and S. A. Oluwadare, "Credit card fraud detection using Machine Learning Techniques: A Comparative Analysis", *Computing Networking and Informatics (ICCNI) 2017 International Conference on*, pp. 1-9.

4. R. Sailusha, V. Gnaneswar, R. Ramesh and G. R. Rao, "Credit Card Fraud Detection Using Machine Learning," *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2020, pp. 1264-1270, doi: 10.1109/ICICCS48265.2020.9121114.

4. D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic and A. Anderla, "Credit Card Fraud Detection - Machine Learning methods," *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, East Sarajevo, Bosnia and Herzegovina, 2019, pp. 1-5, doi: 10.1109/INFOTEH.2019.8717766.

5. Krishna Kumar Mohbey, Mohammad Zubair Khan, and Ajay Indian. 2022. Credit Card Fraud Prediction Using XGBoost: An Ensemble Learning Approach. Int. J. Inf. Retr. Res. 12, 2 (Aug 2022), 1–17. https://doi.org/10.4018/IJIRR.299940

6. IBM. (2021). TabFormer. GitHub. https://github.com/IBM/TabFormer

7. Hassan, Najadat & Altiti, Ola & Abu Aqouleh, Ayah & Younes, Mutaz. (2020). Credit Card Fraud Detection Based on Machine and Deep Learning. 204-208. 10.1109/ICICS49469.2020.239524.

8. Thanh Thi Nguyen, Hammad Tahir, Mohamed Abdelrazek, Ali Babar (2020) "Deep Learning Methods for Credit Card Fraud Detection," [Online]. Available: arXiv:2012.03754

9. Johannes Jurgovsky, Michael Granitzer, Konstantin Ziegler, Sylvie Calabretto, Pierre-Edouard Portier, Liyun He-Guelton, Olivier Caelen. Sequence classification for credit-card fraud detection https://doi.org/10.1016/j.eswa.2018.01.037

10. X. Liu, K. Yan, L. Burak Kara and Z. Nie, "CCFD-Net: A novel deep learning model for credit card fraud detection," *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)*, Las Vegas, NV, USA, 2021, pp. 9-16, doi: 10.1109/IRI51335.2021.00008.

11. G. L. Sahithi, V. Roshmi, Y. V. Sameera and G. Pradeepini, "Credit Card Fraud Detection using Ensemble Methods in Machine Learning," *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 2022, pp. 1237-1241, doi: 10.1109/ICOEI53556.2022.9776955.

12. Ishan Sohony, Rameshwar Pratap, and Ullas Nambiar. 2018. Ensemble learning for credit card fraud detection. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data (CODS-COMAD '18). Association for Computing Machinery, New York, NY, USA, 289–294. https://doi.org/10.1145/3152494.3156815

13. Anthony Taing (2022, January 22). Fraud detection with Graph Attention Networks - Stanford CS224W GraphML Tutorials - Medium. Medium. https://medium.com/stanford-cs224w/fraud-detection-with-gat-edac49bda1a0

14. G. Liu, J. Tang, Y. Tian and J. Wang, "Graph Neural Network for Credit Card Fraud Detection," *2021 International Conference on Cyber-Physical Social Intelligence (ICCSI)*, Beijing, China, 2021, pp. 1-6, doi: 10.1109/ICCSI53130.2021.9736204.

15. Y. Tian, G. Liu, J. Wang and M. Zhou, "ASA-GNN: Adaptive Sampling and Aggregation-Based Graph Neural Network for Transaction Fraud Detection," in *IEEE Transactions on Computational Social Systems*, doi: 10.1109/TCSS.2023.3335485.

16. Y. Liu, X. Ao, Z. Qin, J. Chi, J. Feng, H. Yang, and Q. He, "Pick and choose: a gnn-based imbalanced learning approach for fraud detection," in the Web Conference, 2021, pp. 3168–3177.

17. Sardana, A. (2023, December 5). Optimizing Fraud Detection in Financial Services with Graph Neural Networks and NVIDIA GPUs | NVIDIA Technical Blog. NVIDIA Technical Blog. https://developer.nvidia.com/blog/optimizing-fraud-detection-in-financial-services-with-graph-neural-networks-and-nvidia-gpus/

18. L. Gong and Q. Cheng, "Exploiting Edge Features for Graph Neural Networks," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 9203-9211, doi: 10.1109/CVPR.2019.00943.

19. Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. 2019. Heterogeneous Graph Neural Network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19). Association for Computing Machinery, New York, NY, USA, 793–803. https://doi.org/10.1145/3292500.3330961

20. M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in Proc. Eur. Semantic Web Conf., New York, NY, USA: Springer-Verlag, 2018, pp. 593–607.

21. 23.S. Fan, G. Liu, and J. Li, “A heterogeneous graph neural network with attribute enhancement and structure-aware attention,” IEEE Trans. Comput. Social Syst., early access, Jan. 30, 2023.

22. Math behind graph neural networks. (2022, March 19). Rishabh Anand. https://rish-16.github.io/posts/gnn-math/