T.C.
BAHCESEHIR UNIVERSITY
GRADUATE SCHOOL OF EDUCATION
THE DEPARTMENT OF COMPUTER ENGINEERING

FRAUD DETECTION IN BLOCKCHAIN CRYPTOCURRENCIES

MASTER'S THESIS

OSMAN KUMAŞ

ISTANBUL 2023

T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL OF EDUCATION

THE DEPARTMENT OF COMPUTER ENGINEERING

FRAUD DETECTION IN BLOCKCHAIN CRYPTOCURRENCIES

MASTER'S THESIS

OSMAN KUMAŞ

THESIS ADVISOR
ASSISTANT PROFESSOR ALPER CAMCI

CO-ADVISOR
ASSOCIATE PROFESSOR AKHAN AKBULUT

ISTANBUL 2023

T.C.

BAHÇEŞEHİR UNIVERSITY

GRADUATE SCHOOL

23/06/2023

MASTER THESIS APPROVAL FORM

Program Name:	Computer Engineering
Student's Name and Surname:	Osman Kumaş
Name of The Thesis:	FRAUD DETECTION IN BLOCKCHAIN CRYPTOCURRENCIES
Thesis Defense Date	16.06.2023

This thesis has been approved by the Graduate School which has fulfilled the necessary conditions as Master thesis.

Elif Çetin

Institute Director

This thesis was read by us, quality and content as a Master's thesis has been seen and accepted as sufficient.

	Title, Name	**Signature**
Thesis Advisor:	Assist. Prof. Alper CAMCI	
2nd Member	Assist. Prof. Görkem KAR	
3rd Member (Outside Institution)	Assist. Prof. Fatma PATLAR AKBULUT	

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname : Osman Kumaş

Signature :

ABSTRACT

FRAUD DETECTION IN BLOCKCHAIN CRYPTOCURRENCIES

Kumaş, Osman

Computer Engineering Masters Program

Thesis Advisor: Assistant Professor Alper CAMCI

June 2023, 75 pages

The rise of cryptocurrencies has brought both benefits and challenges, including the increase in cybercriminal activities like money laundering and illegal services. To protect decentralized and anonymous cryptocurrency networks, reliable fraud detection systems are urgently needed. Previous approaches using conventional machine learning and graph-based algorithms have struggled with generalization and robustness. This research focuses on using graph neural network (GNN) architectures, including GCN, GAT, GraphSAGE, GIN, and RGGCN, for cryptocurrency fraud detection.

These GNN architectures capture complex dependencies and relationships in Ethereum transactions modeled as graphs. GCN efficiently handles message-passing and aggregation to learn local and global dependencies, while GraphSAGE uses sampling and aggregation for large-scale graphs. GAT incorporates attention mechanisms to focus on relevant nodes, GIN offers flexibility in handling different graph structures, and RGGCN captures both local and global information with residual connections.

Extensive experiments were conducted on labeled datasets of fraudulent and genuine transactions to develop accurate and robust fraud detection models. Significant features were

extracted from the graph structure, enabling the identification of subtle fraudulent patterns. Overcoming challenges such as imbalanced datasets and scalability is crucial for reliable early detection and prevention of fraud.

The findings contribute to enhancing the safety of digital currencies. The goal is to equip financial institutions, regulatory bodies, and cryptocurrency platforms with advanced GNN detection and mitigation tools, using frameworks like GCN, GAT, GraphSAGE, GIN, RGGCN, to safeguard the security of the global cryptocurrency market.

Keywords: Etherium Fraud Detection, Blockchain Fraud Detection, Graph Neural Networks, Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), Graph Sample and Aggregated (GraphSAGE), Graph Isomorphism Networks (GIN), Residual Gated Graph Convolutional Network (RGGCN).

ÖZ

BLOKZİNCİR KRİPTO PARALARDA DOLANDIRICILIK TESPİTİ

Osman, Kumaş
Bilgisayar Mühendisliği Yüksek Lisans Programı
Tez Danışmanı: Doktor Öğretim Üyesi Alper CAMCI

Haziran 2023, 75 sayfa

Kripto paraların hızla kabul görmesi, para aklama ve yasa dışı hizmetler gibi bir dizi zorluğu da beraberinde getirmiştir. Merkezi olmayan ve anonim yapıları nedeniyle kripto para ağlarının açıklığını ve güvenliğini korumak için güvenilir bir dolandırıcılık tespit sistemi gerekmektedir. Önceki yaklaşımlarda, geleneksel makine öğrenimi ve graf tabanlı algoritmalar genellikle gelecekteki zaman adımlarında genelleme ve sağlamlık sorunları yaşamaktadır. Bu araştırmada, graf yapılı sinir ağı (GNN) mimarileri olan GCN, GAT, GraphSAGE, GIN ve RGGCN, kripto para dolandırıcılığı tespiti için kullanılmaktadır.

Bu GNN mimarileri, Ethereum işlemlerini graf olarak modellenerek, karmaşık bağımlılıkları ve ilişkileri yakalamaktadır. GCN, etkili mesaj iletimi ve birleştirme yetenekleriyle yerel ve küresel bağımlılıkları öğrenirken, GraphSAGE büyük ölçekli grafikler için örnekleme ve birleştirme stratejileri kullanmaktadır. GAT, ilgili düğümlere odaklanmak için dikkat mekanizmalarını entegre ederken, GIN farklı graf yapılarıyla esnek bir birleştirme fonksiyonu sunar ve RGGCN, yerel ve küresel graf bilgisini artakalan bağlantılarla yakalar.

Sağlam ve doğru kripto para dolandırıcılığı tespiti modelleri oluşturmak için sahte ve gerçek işlemlerin etiketlenmiş veri setleri üzerinde kapsamlı deneyler gerçekleştirildi. Graf yapısından önemli özellikler çıkarılarak, hileli desenlerin tespiti mümkün hale getirildi. Dengesiz veri setleri ve ölçeklenebilirlik gibi engellerin aşılması, güvenilir erken tespit ve önleme mekanizmaları için hayati önem taşımaktadır.

Bu bulgular, dijital para birimlerinin güvenliğini artırmaya katkıda bulunmaktadır. Hedef, finansal kuruluşlar, düzenleyici kurumlar ve kripto para platformlarını, GCN, GAT, GraphSAGE, GIN, RGGCN gibi çerçeveler kullanarak gelişmiş GNN tespit ve hafifletme araçlarıyla donatmaktır. Bu sayede küresel kripto para piyasasının güvenliğini sağlamak mümkün olacaktır.

Anahtar Kelimeler: Etherium Dolandırıcılık Tespiti, Blokzincir Dolandırıcılık Tespiti, Graf Nöral Ağı, Graf Konvolüsyon Ağı (GCN), Graf Dikkat Ağı (GAT), Graf Basit ve Toplu Ağı(GraphSAGE), Graf Izomorfizm Ağı (GIN), Artık Geçitli Grafik Konvolüsyon Ağı (RGGCN).

To My Family

ACKNOWLEDGMENTS

I would like to express my profound appreciation to my advisor, Associate Professor Alper Camcı, for his guidance, advice, criticism, encouragement, and insight throughout the research.

I am indebted to my co-advisor, Associate Professor Akhan Akbulut, for his guidance, counsel, criticism, encouragement, and insight throughout the research.

Finally, I would like to thank my family, specifically my lovely wife Selen and son Ali Selim, for their tremendous support throughout her existence. Without their comprehension and unwavering support, I would have never succeeded.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

GNN	Graph Neural Networks
GAT	Graph Attention Networks
HAN	Hierarchial Attention Networks
GCN	Graph Convolutional Networks
GraphSAGE	Graph Sample and Aggregated
ROC	Receiver Operating Characteristic
AUC	Area Under This ROC Curve
GIN	Graph Isomorphism Networks
RGGCN	Residual Gated Graph Convolutional Network

Chapter 1

Introduction

1.1 Theoretical Framework

Numerous industries have been revolutionized by the advent of blockchain technology, particularly in the domain of decentralized digital transactions. Nonetheless, the widespread adoption of blockchain-based cryptocurrencies has attracted malicious actors looking to exploit vulnerabilities and indulge in fraudulent activities. Fraudulent practices, such as phishing schemes, money laundering, and unauthorized transactions, pose a substantial threat to the integrity and dependability of blockchain-based financial systems. Detecting and averting fraud in these decentralized networks is crucial for protecting user interests, maintaining trust, and promoting the continued growth and adoption of blockchain-based cryptocurrencies. The matter at the spot is the implementation of reliable fraud detection mechanisms that adapt to the unique characteristics and difficulties of blockchain-based cryptocurrencies. Due to factors such as pseudonymity, the negligible presence of a regulating body, and the complexity of transactional relationships within the blockchain ecosystem, traditional fraud detection techniques designed for centralized financial systems are often inapplicable to blockchain networks. In order to identify and mitigate fraudulent activities in these decentralized environments, therefore, novel approaches are required. This thesis intends to solve the aforementioned issue by proposing a comprehensive framework for detecting fraud in blockchain-based cryptocurrencies. The research will concentrate on the following primary aims:

Identification of Fraudulent Patterns: Develop innovative algorithms and techniques for detecting and analyzing fraudulent patterns in blockchain transactions. This includes identifying suspicious transactional behaviors, anomalous transaction flows, and transaction metadata anomalies. (Singh, S., Singh, S. & Kajla, T., 2023.)

Graph-Based Analysis: Model the interconnected character of blockchain transactions using graph theory and network analysis methodologies. By representing accounts and transactions as nodes and edges in a graph structure, this research seeks to capitalize on the inherent relationships and dependencies within the network to enhance the accuracy and

efficiency of fraud detection. (Alarab, I., Prakoonwit, S. & Nacer, M.I., 2020,)

Machine Learning and Artificial Intelligence: Utilize cutting-edge machine learning and artificial intelligence techniques to enhance the proposed framework's fraud detection capabilities. This requires the development of predictive models, anomaly detection algorithms, and ensemble learning approaches in order to identify fraudulent accounts and transactions with a high degree of precision and recall. (Schmidhuber, 2015)

By addressing these objectives, this thesis aims to contribute to the creation of robust and efficient fraud detection mechanisms that are tailored particularly for blockchain-based cryptocurrencies. The proposed framework will provide stakeholders, such as financial institutions, regulators, and users, with the tools and insights essential to detect and prevent fraudulent activities within the blockchain ecosystem.

1.2 Statement of the Problem

Integrity and trustworthiness of blockchain-based cryptocurrencies face significant challenges due to fraudulent activities. The decentralized and pseudonymous nature of these digital assets makes them susceptible to various types of deception, such as phishing, money laundering, and unauthorized transactions. To ensure the security and dependability of blockchain networks, it is essential to detect and prevent such fraudulent activities. Existing methods struggle to effectively deal with the graph-structured nature of blockchain data, despite the advancements in fraud detection techniques. Traditional approaches that rely on tabular or sequential data fail to capture the interconnected relationships between blockchain network transactions and entities. Therefore, these methods may have limited precision and scalability when applied to the detection of fraud in cryptocurrencies. This study addresses the need for robust and effective fraud detection mechanisms designed specifically for blockchain-based cryptocurrencies. The objective is to develop novel graph-based methodologies capable of precisely identifying fraudulent transactions and activities by leveraging the underlying graph structure of blockchain data. The research seeks to resolve the following significant issues:

Developing efficient strategies to represent blockchain transactions and entities as a

graph, thereby facilitating the use of graph-based fraud detection algorithms. Previous research has shown the significance of graph-based representations for capturing the complex relationships and dependencies within blockchain networks. (Wang, Luo & Zhou 2020). Extraction of informative characteristics from a graph representation that capture pertinent characteristics and patterns indicative of fraudulent behavior. Graph-specific feature engineering techniques, such as node embeddings and graph convolutional networks, have demonstrated potential for capturing node-level and structural information for fraud detection tasks (Wu, Pan, Chen, Long, Zhang & Philip 2020). Developing scalable and computationally efficient algorithms capable of managing the vastness and volatility of blockchain data. Processing immense amounts of blockchain transactions and facilitating real-time fraud detection necessitates efficient graph-based algorithms and optimization techniques (Hamilton, Ying & Leskovec, 2017).

By addressing these obstacles, this study hopes to contribute to the creation of sophisticated fraud detection mechanisms that are tailored specifically for blockchain-based cryptocurrencies. The findings of this study will provide valuable insights and actionable suggestions for enhancing the security and dependability of decentralized financial systems.

1.3 Purpose of The Study

The primary objective of this study is to overcome the shortcomings of previous approaches by harnessing the capabilities of these state-of-the-art GNN architectures. GCN, introduced by Kipf and Welling, (Kipf & Welling, 2017), enables effective information propagation and feature aggregation among connected nodes in the bitcoin transaction graph. GAT, proposed by Veličković (Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. & Bengio, Y., 2017), incorporates attention mechanisms to assign different importance weights to neighboring nodes, allowing the model to focus on the most relevant information. GraphSAGE, introduced by Hamilton (Hamilton, Ying & Leskovec, 2017), addresses the scalability issue by sampling and aggregating features from local neighborhoods, enabling efficient learning on large-scale graphs. GIN, provides a flexible and expressive aggregation function that can capture structural similarities in diverse graph datasets. RGGCN, introduced by Li (Li, Tarlow, Brockschmidt & Zemel, 2015),

incorporates residual connections and gated mechanisms to capture both local and global graph information, enhancing the model's representation power.

Through extensive experimentation and evaluation on real-world bitcoin transaction data, this study aims to assess the effectiveness of these GNN architectures in detecting fraudulent activities. The performance of these architectures will be compared in terms of accuracy, precision, recall, and F1-score to highlight their capabilities in overcoming the limitations of previous approaches. By achieving this purpose, this research contributes to the advancement of fraud detection techniques specifically tailored for bitcoin transactions. The utilization of state-of-the-art GNN architectures provides novel insights into uncovering hidden patterns and enhancing the security and integrity of the blockchain ecosystem.

Examine the influence of various graph-based techniques on the scalability and computational efficiency of fraud detection systems.Evaluate the computational requirements and operational performance of GCN, GAT, GraphSAGE, GIN and RGGCN when processing massive blockchain datasets. To enhance the efficacy of fraud detection algorithms, investigate possible optimizations and trade-offs. By addressing these objectives, this study intends to contribute to the existing body of knowledge on graph-based fraud detection in blockchain-based cryptocurrencies by providing valuable insights and recommendations for enhancing the security and reliability of decentralized financial systems.

1.4 Hypothesis

This thesis proposes that employing graph-based applications and attention mechanisms can considerably improve the accuracy and efficacy of blockchain-based cryptocurrency fraud detection. By leveraging the inherent network structure and relationships present in blockchain transaction data, in conjunction with attention mechanisms that focus on relevant features and patterns, it is anticipated that the proposed approach will improve the ability to detect fraudulent accounts and transactions, thereby reducing financial losses and ensuring the integrity of blockchain-based financial systems.

This hypothesis is supported by findings from prior research that demonstrate the efficacy of graph-based approaches and attention mechanisms in various domains. Graph-based models, such as Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), Graph Sample and Aggregated(GraphSAGE), Graph Isomorphism Networks (GIN) and Residual Gated Graph Convolutional Network(RGGCN) have been shown to be capable of capturing complex relationships and dependencies in network data, resulting in improved performance in tasks such as node classification and link prediction (Kipf & Welling, 2017) (Velickovic, Cucurull, Casanova, Romero, Lio & Bengio, 2017). In addition, attention mechanisms have demonstrated their effectiveness in concentrating on relevant information and boosting the discriminative power of neural networks.

It is hypothesized that employing graph-based applications and attention mechanisms to the specific problem of fraud detection in blockchain-based cryptocurrencies will result in the following outcomes:

- Enhanced Accuracy: The incorporation of graph-based features and attention mechanisms will enhance the model's ability to capture nuanced patterns and anomalies associated with fraudulent accounts and transactions, resulting in an increase in the accuracy of detecting fraudulent activities. By considering the interconnectedness of accounts and transactions, the graph-based approach is anticipated to increase the model's robustness against adversarial attacks and attempts to conceal fraudulent behavior.

- Efficient Feature Extraction: Attention mechanisms will enable the model to prioritize essential information and focus on relevant features, resulting in more efficient and effective fraud detection with reduced computational requirements. Experiments will be conducted on real-world blockchain datasets to empirically verify the hypothesis, comparing the performance of the proposed graph-based approach with existing fraud detection methods. The evaluation metrics precision, recall, F1 score, and area under the ROC curve (AUC) will be used to determine the efficacy of the proposed method for detecting and preventing fraud in blockchain-based cryptocurrencies.

Chapter 2

Literature Review

2.1 Previous Research

By mining Ethereum-based transaction records, Tan (Tan, Tan, Zhang & Li, 2021) proposes a method for detecting Ethereum deception. Using web crawlers to capture labeled fraudulent addresses, reconstructing a transaction network based on the public transaction book, extracting node features for identifying fraudulent transactions using an amount-based network embedding algorithm, and using the graph convolutional network model to classify addresses into legal addresses and fraudulent addresses are all steps involved in this method.

A One-Class Graph Neural Network-based anomaly detection framework for the Ethereum blockchain network is proposed by Patel (Patel, Pan & Rajasegarar, 2020). The proposed method can detect anomalies with greater precision than traditional non-graph-based machine learning algorithms. Due to their incapacity to capture internode or account relationship information, traditional machine learning-based techniques such as One-Class Support Vector Machine and Isolation Forest are ineffective at identifying anomalies in Ethereum transactions. The conclusion of the paper is that the proposed method can effectively represent Ethereum transactions using an attributed graph with nodes and edges that depict interdependencies, making it a more effective method for detecting anomalies in the Ethereum blockchain network.

Poursafaei proposes SigTran, a graph-based technique for detecting malicious nodes on blockchain networks (Poursafaei, Rabbany & Zilic, 2021). SigTran generates a graph from transaction records in a blockchain and represents nodes according to their structural and transactional properties. These node representations reliably distinguish between nodes participating in illicit activities. SigTran achieves an F1 score of 0.92 on Bitcoin and 0.94 on Ethereum, outperforming significantly more complex platform-dependent models on these benchmarks. The conclusion of this paper is that SigTran is an efficient and general method for detecting unlawful activity in blockchain transaction networks.

The conclusion of the paper (Pourhabibi, Ong, Kam & Boo, 2020) is that graph-based anomaly detection techniques are among the most commonly used techniques for analyzing connectivity patterns in communication networks and identifying suspicious behaviors. However, the nature of the input network is a fundamental aspect of Graph Based Anomaly Detection (GBAD) approaches, and the limitations of supervised and semi-supervised learning techniques may prevent the implementation of GBAD approaches in certain circumstances. The authors suggest that future research should focus on developing more efficient and effective GBAD approaches capable of managing large-scale and complex networks.

Kumar, A., Ghosh, S. Kumar, Ghosh, and Verma (Kumar, Ghosh & Verma, 2022) propose a self-training method that uses a guided sharpening technique with a pair of autoencoders to incorporate unlabeled data into the training procedure. Autoencoders are neural networks that learn to reconstruct their input data and are utilized for feature extraction and dimensionality reduction. Using one autoencoder to generate a sharpened version of the input data and another autoencoder to reconstruct the original input data from the sharpened version is the guided sharpening technique. This procedure offers helpful hints for incorporating unlabeled data into the training procedure. They conducted experiments on three distinct real-world databases to demonstrate the efficacy of the methodology. On the elliptic bitcoin fraud dataset, they demonstrated that incorporating unlabeled data increased the F1 score of the model trained on limited labeled data by approximately 10%. The F1 score is a measure of a model's accuracy that takes both precision and recall into account.

Duan (Duan, Yan, Dong, Zhang & Yu, 2022) highlights the vulnerabilities of the blockchain ecosystem to criminal activity and the limitations of existing phishing fraud detection technologies, which largely rely on shallow machine learning, resulting in low detection precision. The paper proposes the TransDetectionNet graph classification network model to address this issue. The model uses the Edge-sampling To Node Vector (Esmp2NVec) node embedding algorithm to extract the features concealed within the directed transaction network. In order to extract node features, the paper employs graph

convolutional neural networks (GraphSAGE and GCN) to learn the topological space structure between nodes, where nodes represent Ethereum accounts.

A Heterogeneous Graph Transformer Networks (S_HGTNs) model for smart contract anomaly detection is proposed to identify financial malfeasance on the Ethereum platform (Liu, Tsai, Bhuiyan, Peng & Liu, 2022). By extracting features from intricate smart contracts, the model effectively identifies anomalous smart contracts. The classification results indicate that this model outperforms the conventional model, and the small standard deviation demonstrates the model's effectiveness and consistency. This paper concludes that anomaly detection for smart contracts can effectively prevent concealed security threats such as financial fraud, illegal financing, and money laundering.

Using machine learning to detect illicit Bitcoin transactions in Elliptic data, one of the largest Bitcoin transaction graphs (Alarab & Prakoonwit, 2023). According to the paper, using supervised learning and graph convolutional network models for anti-money laundering has produced promising results in prior research. However, these studies failed to account for the temporal information of the dataset, yielding unsatisfactory results. In addition, the literature on the applicability of active learning to blockchain datasets is limited. In order to overcome these limitations, the paper proposes a classification model that integrates long-short-term memory with GCN, dubbed temporal-GCN, to classify illicit transactions based solely on transaction characteristics. Literature-documented active learning frameworks utilizing a variety of acquisition functions are contrasted here.

SemiGNN is a semi-supervised attentive graph neural network proposed for financial misconduct detection (Wang, Lin, Cui, Jia, Wang, Fang, Yu, Zhou, Yang & Qi, 2019). The network uses multi-view labeled and unlabeled data for fraud detection and proposes a hierarchical attention mechanism to better correlate distinct neighbors and different views. Attention mechanism makes the model interpretable by specifying which fraud factors are significant and why users are predicted to be fraudulent. The investigation conducted with Alipay users demonstrates that the proposed method achieves greater accuracy than existing methods on two tasks. The results that are interpretable also provide insightful insights into the duties. The conclusion of the paper is that the proposed SemiGNN method can be used

to detect financial misconduct and generate interpretable results.

Hassanzadeh, Nayak, and Stebili (2012) propose a framework for analyzing the effectiveness of various graph metrics in identifying individuals with anomalous relationships in online social networks. Using their methodology on datasets from existing online social networks, the authors identified a number of performance-enhancing metrics. The empirical analysis shows that the relationship between average betweenness centrality and edges detects anomalies more precisely than other methods. The paper concludes that the direct connectivity of online social networks can facilitate illegal activity, and that their proposed framework can aid in detecting anomalous behavior in such networks.

Kaufman (Kaufman & Iaremenko, 2022) concentrates on detecting anomalies in real-time Ethereum trades from specific accounts in order to prevent potential accidents and economic losses caused by fraud on the cryptocurrency market. The authors evaluated the efficacy of traditional and novel algorithms for detecting pointwise, contextual, and collective anomalies in the sample, time, and frequency domains. They categorized the algorithms in accordance with their detection strategy and labeled a point as an anomaly if it received a majority vote. The paper concludes that combining distinct groups of algorithms can result in an effective real-time detector with an alarm time of no more than a few seconds and a high degree of certainty.

Noekhah (Noekhah, binti Salim & Zakaria, 2020) discusses the growing need for detecting opinionated spam in e-commerce in order to prevent its negative effects on business reputations. Existing spam detection techniques take into account only one or two types of spam entities, such as reviews, reviewers, reviewer groups, and products. In addition, they employ a limited number of features pertaining to behavior, content, and the relationship between entities, which reduces the detection accuracy. These techniques rely predominantly on synthetic datasets to analyze their model and cannot be applied to the real-world setting. To overcome these limitations, the researchers propose a novel graph-based model known as "Multi-iterative Graph-based opinion Spam Detection" (MGSD). This model simultaneously analyzes all entity types within a unified structure, revealing both

implicit (i.e., similar entity's) and explicit (i.e., different entity's) relationships. The MGSD model evaluates the'spamminess' effects of entities more efficiently by employing a novel multi-iterative algorithm that considers various sets of factors to update the entities' spamminess score.

Hassan (Hassan, Rehmani & Chen, 2022) focuses on detecting anomalies in real-time Ethereum transactions from specific accounts to prevent potential accidents and economic losses caused by fraud on the cryptocurrency market. The authors assessed the performance of conventional and innovative algorithms for detecting pointwise, contextual, and collective anomalies in the sample, time, and frequency domains. According to their detection strategy, they categorized the algorithms and designated a point as an anomaly if it received a majority vote. Combining distinct groups of algorithms can result in an effective real-time detector with an alarm time of no more than a few seconds and a high degree of certainty, according to the paper's conclusion.

Hu (Hu, Li, Zhuang, Huang & Dong, 2020) proposes GFD, an innovative method for detecting deception in mobile advertising. The method identifies fraudulent mobile advertising applications using a weighted heterogeneous graph and techniques of deep learning. The method builds a weighted heterogeneous graph to represent behavior patterns among users, mobile apps, and mobile advertisements and employs a time window-based statistical analysis technique to extract intrinsic features from the tabular sample data. In addition, a neural network that incorporates graph-based and attribute-based features is proposed for distinguishing fraudulent apps from legitimate apps. The experimental results on a real-world dataset reveal that the proposed method outperforms conventional learning techniques.

GCNs to detect money laundering on the Bitcoin blockchain (Alarab, Prakoonwit & Nacer, 2020). The authors conducted experiments to determine the efficacy of GCNs in detecting suspicious transactions and compared it to other cutting-edge methods. In terms of accuracy and effectiveness, the results demonstrated that GCNs outperformed other techniques. Therefore, it is possible to conclude that GCNs in the Bitcoin blockchain possess

a high level of anti-money laundering capability.

According to Kamiali, these two applications are the most productive for applying data mining techniques to blockchain data (Kamiali, Kramberger & Fister, 2021). Examining the current trends in leveraging the synergies of blockchain technology and data mining techniques for anomaly detection, as well as identifying the most important machine learning techniques and constructing a taxonomy of those techniques used to enhance blockchain technology for specific purposes, was the objective of this review. The authors also examined the data mining techniques employed over the past five years and found that Gradient Boosting was the most widely used technique during the first two years. Throughout this five-year research period, SVM and Random Forest were two of the most frequently employed methods. In spite of this, the authors observed that these two methods yielded the best results in the majority of studies published in 2019 and 2020, with Random Forest also being popular in 2021. The authors have observed an increase in the use of neural networks, gradient boosting, deep learning, and LSTM over the past two years.

Using graph neural network models to detect phishing fraud on the Ethereum trading network (Kanezashi, Suzumura, Liu & Hirofuchi, 2022). In comparing the efficacy of homogeneous and heterogeneous GNN models, the authors found that heterogeneous models that account for transaction edge types outperform homogeneous models. Across all metrics, the RGCN model performed the best in particular. However, accuracy did not improve when node types were considered as novel input features compared to the baseline homogeneous GNN model. In conclusion, heterogeneous GNN models are more effective at detecting deception in the Ethereum trading network.

Bangcharoensap (Bangcharoensap, Kobayashi, Shimizu, Yamauchi & Murata, 2015) proposes a graph-based semi-supervised learning technique for detecting online auction fraud. The strategy relies on the social interactions of fraudsters and their propensity to participate in auctions hosted by members of the same collusion group. The authors extended the modified adsorption model to enable the propagation of information from a small group of identified fraudsters to the entire graph. They found that fraudsters frequently engage in intense interactions with their neighbors and incorporated this finding into the 2-STEP model. In addition, the authors discovered that weighted degree centrality is a

distinguishing characteristic between fraudsters and legitimate users, which they utilized to actively detect fraud. According to data from the real world, incorporating weighted degree centrality into the model can significantly enhance precision.

The study (Tang, Li, Gao & Li, 2022) proposes Contrastive Learning of Last Actions (CLL) as a technique for capitalizing on distinctions between users. CLL is designed for unlabeled data and uses neural network sequence models to comprehend the difference between a user's most recent and prior actions. This study introduces a Global Model (GM) that detects all users in order to address the issue of underfitting and capitalize on the similarities between users. This allows the model to adapt to various users and increase its detection effectiveness. The experimental results demonstrate that CLL is superior to other loss functions and that GM is effective at enhancing model performance and substantially reducing training and testing durations. When detecting anomalies on trading platforms, the paper emphasizes the significance of taking into consideration both user similarities and user differences.

Using Bitcoin's anonymity mechanism, ransomware attacks have proliferated in recent years (Wang, Pang, Chen, Zhao, Huang, Chen & Han, 2021). In addition to the amount of ransom demanded, ransomware activities result in recovery costs, reputational injury, and productivity loss. The most prevalent type of ransomware is crypto-ransomware, which encrypts the victim's files and demands Bitcoin payment in exchange for the decryption key. In addition, the study revealed that ransomware attacks are more likely to target businesses than individuals. According to the study's authors, prevention is the key to mitigating the impact of ransomware attacks. They recommend regular backups, employee training, and software updates to prevent exploits of vulnerabilities.

The effectiveness of blockchain technology in detecting fraud in various business domains, such as insurance, banks, online transactions, real estate, and credit card use (Singh & Kajla, 2023). The study employed a Systematic Literature Review (SLR) methodology with keywords such as blockchain, fraud detection, and financial domain to investigate the increasing acceptance of blockchain technology. Individuals with a variety of business objectives can rely heavily on blockchains to combat fraud, as the research accentuates the

real-world applications of blockchain technology to secure the gateway for online transactions. The SLR of this study aids in the identification of future avenues with practical implications by drawing to the attention of researchers the evaluations of blockchain's efficacy that have already been conducted. However, it does not rule out the unexplored possibility of zero or less efficacy in some organizations.

Existing fraud detection systems in the e-commerce sector have degraded performance and cannot adapt to new fraud patterns. The authors propose eFraudCom, a competitive graph neural network (CGNN)-based fraud detection system, to address this issue. Modeling normal and fraudulent behavior distributions independently, the CGNN is used to classify user behaviors. Some normal behaviors are used as limited supervision information to guide the CGNN in building a stable normal behavior profile. This eliminates the system's dependence on fraudulent behaviors and enables it to identify new fraud patterns (Zhang, Li, Huang, Wu, Zhou, Yang & Gao, 2022).

Using machine learning techniques, (Valadares, Villela, Bernardino, Goncalves & Vieira, 2023) propose a method for identifying professional and informal user profiles in Ethereum. The proposed method groups and categorizes user behavior by combining unsupervised and semi-supervised learning. In terms of accuracy, precision, recall, F-scores, MCC, and AUC-ROC, the results indicate that the proposed method outperforms existing supervised learning techniques.

2.2 Blockchain

Blockchain technology, which was introduced in 2008 by the pseudonymous figure Satoshi Nakamoto, has received considerable attention for its potential to revolutionize numerous industries. A blockchain is essentially a distributed and decentralized ledger that securely records and verifies transactions. It eliminates the need for centralized intermediaries by providing a transparent, immutable record of digital assets or information (Nakamoto, 2008). A blockchain is a chain of blocks, each of which contains a catalog of transactions or data. These blocks are cryptographically linked, ensuring the integrity and immutability of the information recorded. The decentralized nature of the blockchain,

facilitated by a network of nodes, renders it resistant to interference and censorship (Swan, 2015). The capacity of blockchain technology to obtain consensus among multiple participants in a trustless environment is one of its defining characteristics. Through mechanisms such as proof-of-work (PoW), proof-of-stake (PoS), and other variants, consensus is reached on the validity and ordering of transactions. This consensus process ensures the security and consistency of the blockchain (Nakamoto, 2008).

The applications of blockchain technology extend beyond cryptocurrencies like Bitcoin. It can be used to facilitate a variety of tasks, including supply chain management, digital identity verification, decentralized finance, and voting systems, among others. Smart contracts, which are coded agreements with predefined conditions, extend the capabilities of blockchain by facilitating automated and transparent transactions (Swan, 2015). The benefits of blockchain technology stem from its decentralized and transparent nature, which reduces reliance on intermediaries and increases participant confidence. Blockchain has the potential to increase efficiency, security, and accountability in a variety of domains by eradicating single points of failure and providing transparent auditability (Tapscott, D. & Tapscott, A., 2017.).However, there are challenges associated with blockchain technology. Among the main areas requiring additional research and development are scalability, energy consumption, privacy, and regulatory concerns. Understanding the underlying principles of blockchain technology, its mechanisms, and its potential applications is crucial for evaluating its benefits, addressing its limitations, and realizing its transformative potential.

2.3 Blockchain Cryptocurrencies

In the sphere of decentralized digital currencies, blockchain-based cryptocurrencies have emerged as a disruptive innovation. The blockchain technology enables secure peer-to-peer transactions without the need for intermediaries (Nakamoto, 2008). Bitcoin, which Nakamoto introduced in 2008, is the pioneering cryptocurrency that uses blockchain technology to facilitate decentralized and trustless transactions. Since then, numerous cryptocurrencies have been created, each with its own distinct characteristics and applications. The advantages of blockchain-based cryptocurrencies over traditional financial systems are numerous. First, they provide increased security via cryptographic

techniques, making it difficult for unauthorized parties to alter transaction records (Bano, S., Al-Bassam, M. & Danezis, G., 2017). The decentralized nature of blockchain ensures that no single entity controls the network, thereby fostering transparency and minimizing the risk of fraud and manipulation (Swan, 2015). Moreover, the use of consensus mechanisms, such as proof-of-work (PoW) and proof-of-stake (PoS), guarantees the integrity and immutability of transactions.

Despite these benefits, however, blockchain-based cryptocurrencies encounter a number of obstacles. The prevalence of fraudulent activities within the blockchain ecosystem is a significant obstacle. Due to the pseudonymous nature of transactions and the absence of a central authority, cryptocurrencies are attractive to fraudsters (Koshy , 2014). In the cryptocurrency space, various types of fraud, including phishing attacks, Ponzi schemes, and money laundering, have been observed. To address these obstacles, researchers and industry professionals have investigated various approaches for detecting and preventing misconduct in blockchain-based cryptocurrencies. Traditional methods, such as rule-based systems and statistical analysis, have been utilized, but they frequently fail to detect sophisticated and constantly evolving fraud schemes . Consequently, there is a growing need for sophisticated techniques that can effectively identify fraudulent activities by leveraging the unique characteristics of blockchain data.

In recent years, graph-based approaches have garnered considerable interest as a promising technique for detecting fraud in blockchain-based cryptocurrencies. By representing transactions and entities as nodes in a graph and capturing their complex relationships, graph-based methods provide an effective framework for detecting patterns and anomalies indicative of fraudulent behavior (Wang, Luo & Zhou, 2020). These approaches investigate the framework and fluctuations within the blockchain network using graph neural networks and graph algorithms, that allows more precise and capable of scaling fraud detection. (Hamilton, Ying & Leskovec, 2017).

In conclusion, blockchain-based cryptocurrencies offer a decentralized and secure transaction system. Nonetheless, the prevalence of fraud presents significant challenges to

the integrity and dependability of these digital assets. Utilizing the inherent graph structure of the data to identify fraudulent activities, graph-based methods have emerged as a promising solution for fraud detection in blockchain-based cryptocurrencies. In the sections that follow, we will scrutinize the specific graph-based methodologies and techniques utilized for blockchain-based cryptocurrency fraud detection.

2.3.1 Bitcoin. It serves as a platform for the development and execution of smart contracts, which are Bitcoin, which Satoshi Nakamoto founded in 2008, is the very first and most well-known decentralized cryptocurrency. It operates on a blockchain-based peer-to-peer network that serves as the public ledger for all Bitcoin transactions. Bitcoin revolutionized the concept of digital currency by eliminating intermediaries such as banks and granting users direct control of their funds. (Nakamoto, 2008). Bitcoin relies on cryptographic techniques to secure transactions and preserve the blockchain's integrity. A decentralized and immutable ledger of all Bitcoin transactions is created by grouping transactions into blocks and linking them in a sequential chain. This distributed ledger ensures transparency and prevents double-spending, which occurs when the same Bitcoin is utilized in multiple transactions (Swan, 2015). Proof-of-work (PoW) is the agreement procedure responsible for the decentralized character of Bitcoin. The miners of the Bitcoin network compete to solve complex mathematical problems; the person who is first to do so adds a new block to the blockchain and gets recognized with newly-minted Bitcoins. This process assures the security and immutability of the blockchain, as modifying a previous block would require a substantial amount of computational power and be economically prohibitive (Nakamoto, 2008). In addition, Bitcoin addresses, which are represented by cryptographic keys, are utilized to transmit and receive funds. These addresses are pseudonymous because, unless the user voluntarily discloses it, they do not divulge the user's identity. The public nature of the blockchain enables the transparent monitoring of Bitcoin transactions and addresses (Swan, 2015).

Bitcoin's decentralized and visible nature, coupled with its capacity for pseudonymous transactions, has made it attractive for a number of applications and cases, including online payments, remittances, and investments. However, Bitcoin's accelerated growth and adoption have posed challenges, particularly in the areas of security and fraud detection. Malicious actors have exploited the inherent anonymity and pseudo-anonymity of Bitcoin

transactions for money laundering and other illicit transactions, ransomware attacks, and fraudulent schemes (Foley, Karlsen & Putniņš, 2019).

For the development of effective fraud detection mechanisms in the cryptocurrency ecosystem, it is crucial to comprehend the complexities of Bitcoin, its transactional dynamics, and the vulnerabilities associated with its decentralized nature. Using sophisticated techniques and models, such as graph-based applications and attention mechanisms, it is possible to improve the detection and prevention of fraudulent activities, thereby contributing to the security and credibility of the Bitcoin network and the broader blockchain ecosystem.

2.3.2 Ethereum. Ethereum, which Vitalik Buterin introduced in 2015, is a decentralized blockchain platform that exceeds the capabilities of traditional cryptocurrencies like Bitcoin. It serves as a platform for the development and execution of smart contracts, which are agreements whose terms are written directly in code (Buterin, 2014). Ethereum's novel strategy enables the creation of decentralized programs capable of individually executing transactions and interacting with one another on the Ethereum blockchain. Ether (ETH) is Ethereum's (ETH) native cryptocurrency. Ethereum operates on a blockchain. Ethereum, like Bitcoin, uses a decentralized network of nodes to maintain a transparent and immutable transaction ledger. Ethereum, on the other hand, is distinguished by its incorporation of a language for programming that is Turing-complete, allowing developers to construct and deploy smart contracts (Buterin, 2014). On the Ethereum platform, smart contracts are executed by the Ethereum Virtual Machine (EVM), which operates on each network node. This virtual machine facilitates code execution without requiring a central authority. Smart contracts have many possible uses that includes banking, decentralized trading systems, as well as supply chain management (Buterin, 2014). Ethereum's capacity to support decentralized applications via decentralized autonomous organizations (DAOs) is one of its key characteristics. DAOs are organizations that operate on the Ethereum blockchain, are administered by a community of stakeholders, and are governed by smart contracts (Buterin, 2014). This decentralized governance structure permits greater openness, equity, and resiliency. Additionally, Ethereum introduced ERC-20 tokens, which are fungible tokens built on the Ethereum blockchain. These tokens have become the basis for a variety of blockchain-based initiatives, such as Initial Coin Offerings

(ICOs) and decentralized finance (DeFi) applications (Buterin, 2015). They allow for the creation of new digital assets and offer a standardized framework for token implementation and interoperability.

Significant growth and adoption of the Ethereum network have attracted developers and businesses from a variety of industries. As with any innovative technology, Ethereum is not devoid of obstacles. Scalability, security, and the detection of fraudulent activities within the Ethereum ecosystem are ongoing issues requiring additional research and development. Understanding the complexities of Ethereum's architecture, smart contracts, and potential for decentralized applications is essential for advancing blockchain technology and maximizing its potential across multiple industries. Researchers and practitioners can contribute to the creation of more secure, scalable, and efficient decentralized systems by addressing the challenges and investigating the opportunities presented by Ethereum.

2.4 Fraud Detection Methods in Artificial Intelligence

Artificial intelligence (AI) techniques have demonstrated remarkable efficacy in identifying and preventing fraudulent activities in the crucial application area of fraud detection. (Phua, Lee, Smith & Gayler, 2010) Traditional rule-based and statistical methods have limitations in detecting complex and evolving fraud schemes, necessitating sophisticated AI-based approaches. In recent years, a number of AI techniques, such as machine learning and deep learning, have been applied to fraud detection with encouraging outcomes.

2.4.1 Machine learning approaches. Due to their ability to understand patterns and anomalies from massive datasets, machine learning algorithms have been widely implemented for fraud detection. On the grounds of a derived set of data attributes, supervised learning algorithms such as logistic regression, decision trees, and random forests have been utilized for identifying transactions as fraudulent or legitimate. (Bolton , 2002). Support Vector Machines (SVMs) have also been utilized for the detection of fraud due to their capacity to differentiate between fraudulent and non-fraudulent instances in a

feature with a high dimension space. (Dal Pozzolo, 2015). For the detection of previously unknown fraud patterns, unsupervised learning techniques, such as clustering algorithms and anomaly detection, are useful. (Bhuyan, Bhattacharyya & Kalita, 2013.) Clustering algorithms, such as k-means and DBSCAN, group similar transactions together, allowing the identification of clusters with anomalous behavior. A study (Rosenfeld, Restrepo, Gerard, Bruce, Branch, Lewin & Bezzo, 2018) report that anomaly identification algorithms, including one-class SVM and Isolation Forest, detect changes from conventional patterns without needing training data with labels.

2.4.2 Deep learning approaches. Deep learning approaches and artificial neural networks particularly have exhibited extraordinary performance in numerous domains, involving fraud detection. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have been used for recognizing credit card transaction fraud in temporal information using RNNs and LSTMs. (Dal Pozzolo, Boracchi, Caelen, Alippi & Bontempi, 2017). A study (Jiang, Chen, Gu, Choo, Liu, Yu, Huang, & Mohapatra, 2019) report that Convolutional Neural Networks (CNNs) have been used to derivered meaningful attiributes from transactional data, allowing the detection of complex fraud patterns.

Graph neural networks (GNNs) became known as an efficient fraud detection tool, especially in network-based fraud scenarios. GNNs utilize the inherent graph structure of data to assimilate relational information between entities and recognize unusual trends (Ying, 2018). Graph Neural Networks (GNNs), including Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs) have been applied to fraud detection tasks with improved precision and scalability. (Wu, Pan, Chen, Long, Zhang, & Philip, 2020).

Using graph neural networks, machine learning and deep learning detection of fraud methods in artificial intelligence have made significant advances. These techniques enable the analysis of massive datasets, the identification of previously unknown fraudulent activities, and the detection of complex fraud patterns. In the sections that follow, we will examine the specific AI-based fraud detection techniques used for blockchain-based cryptocurrencies.

2.5 Graph Based Applications

In recent years, graph-based applications have received considerable attention as a robust framework for analyzing and modeling complex relationships and structures. A graph is a collection of elements (also known as vertices) connected by edges in this context. Graphs offer a flexible and intuitive representation of interlinked data, making them suitable for various applications, including social networks, recommendation systems, biological networks, and fraud detection, among others. By representing entities and their relationships, graphs capture both structural and semantic information about a system. Nodes in a graph can represent various entities, such as users, products, web pages, or DNA, while edges represent the relationships or interactions between these nodes. This representation makes it possible to investigate complex phenomena such as patterns, clustering, and influence propagation. (Fortunato, 2010.).

The ability of graph-based applications to incorporate both local and global dependencies within a network is a significant advantage. Local dependencies refer to a node's immediate neighbors, whereas global dependencies encompass the structure and connectivity of the graph as a whole. This all-encompassing perspective enables the analysis of complex systems, the identification of influential nodes, and the prediction of behaviors based on the network topology (Leskovec, Huttenlocher & Kleinberg, 2010). Utilizing graph algorithms and machine learning techniques, graph-based applications frequently extract valuable insights and make predictions. Graph algorithms, such as PageRank, community detection, and measures of graph centrality, provide instruments for analyzing the structure and properties of graphs. A study (Hamilton, Ying & Leskovec, 2017) describe how machine learning techniques including graph neural networks (GNNs), graph-based clustering, and link prediction facilitate the extraction of patterns and predictive modeling on graph data. Multiple domains have demonstrated the effectiveness of graph-based applications. Graph-based algorithms facilitate the identification of communities, influential users, and information diffusion patterns, for example, in social network analysis. Graphs capture user-item interactions in recommendation systems and enable personalized recommendations based on graph-based similarity measures. In bioinformatics, graphs are

used to model protein-protein interactions and gene regulatory networks, thereby facilitating drug discovery and disease analysis (Kwak, Lee, Park & Moon 2010). Understanding the principles and techniques of graph-based applications is crucial for realizing the potential of graph data and maximizing its benefits across multiple domains. Researchers and practitioners can develop accurate predictive models and make informed decisions based on the complex relationships and structures inherent in the data by leveraging the power of graphs.

2.5.1 Graph convolutional networks. Graph Convolutional Networks (GCNs) are known as a reliable approach for learning representations and performing operations on graph-structured data. GCNs extend Convolutional Neural Networks (CNNs) to graphs, facilitating the application of deep learning techniques to graph data. (Kipf & Welling, 2017). GCNs facilitate the modeling of complex relationships and the extraction of meaningful features from graph nodes by utilizing the structural information and connectivity patterns of graphs. Convolutional layers in conventional CNNs operate on regular grids, such as images, in which each pixel has a fixed number of neighbors. However, graphs lack a defined neighborhood structure and are inherently irregular. GCNs resolve this difficulty by defining convolutional operations on graphs, which enables the aggregation of data from neighboring nodes.

GCNs are designed to propagate and update node representations by collecting data from adjacent nodes. This is accomplished by combining the characteristics of a node with those of its neighbors using graph-based message transit schemes. The aggregation process enables the encoding of both local and global structural information (Kipf & Welling, 2017) by allowing each node to collect data from its immediate vicinity. GCNs typically consist of multiple layers, each of which aggregates and transforms node representations. In each layer, node characteristics are modified by considering the characteristics of neighboring nodes, typically via a weighted sum or concatenation operation. The transformed features then capture complex patterns and relationships within the graph (Veličković, Cucurull, Casanova, Romero, Lio & Bengio, 2018). GCNs are able to encapsulate higher-order dependencies in graphs, which is one of their primary benefits. Information from distant nodes can be propagated and incorporated into node representations by layering multiple GCN layers. This hierarchical aggregation facilitates the modeling of complex relationships

and duties such as classification of nodes, prediction of links, and predictions at the graph level.

GCNs have demonstrated outstanding performance in numerous disciplines, including social network analysis, recommendation systems, biological network analysis, and fraud detection. They have been successfully applied to duties like community detection, node classification, and anomaly detection, demonstrating their effectiveness in collecting graph structure and learning meaningful representations from graph data. (Zhou, Cui, Zhang, Yang, Liu & Sun, 2018).

2.5.2 Graph attention aetworks. Graph Attention Networks (GAT) have come about as an efficient architecture for learning representations of graph-structured data and executing tasks on such data. GATs enhance the expressiveness and modeling capabilities of graph convolutional networks (GCNs) by integrating attention mechanisms that enable adaptive feature aggregation and weighting based on the importance of neighboring nodes. (Veličković, P., Cucurull, Casanova, Romero, Lio & Bengio, 2018). This allows GATs to concentrate on relevant nodes and capture intricate graph relationships. GATs are based on attention weights for each neighboring node, which indicate the significance or relevance of that node's characteristics to the target node. This attention mechanism permits GATs to dynamically assign various weights to different nodes during the aggregation process, thereby capturing fine-grained dependencies and decreasing reliance on fixed neighborhood structures (Kipf & Welling, 2017).

Attention weights in a GAT are computed using a self-learned mechanism that considers both the characteristics of the target node and its companions. A shared attention mechanism, which is typically implemented as a layer of a neural network, generates attention coefficients for each pair of graph nodes. Attention coefficients indicate the significance or relevance of neighboring characteristics to the target node.(Velickovic, Fedorov, Lio, Bresson & Hamilton, 2020). The attention weights are then used to calculate a weighted sum of the neighboring node's characteristics, taking the learned attention coefficients into consideration. This process of aggregation enables each node to acquire information from its neighbors, focusing on the most relevant nodes and effectively capturing the underlying graph structure. In addition, GATs frequently employ a multi-head mechanism in which multiple attention mechanisms are applied in parallel, enabling the

model to capture diverse aspects of node interactions and increase its expressive power (Lee & Cho, 2021). Utilizing attention mechanisms in GATs is advantageous in numerous ways. First, it allows the model to allocate greater weight to relevant nodes, enabling it to focus on the most informative portions of the graph. Second, it enables GATs to identify long-range dependencies by assigning non-zero attention weights to distant nodes in the graph. Thus, GATs are ideally adapted for tasks involving large and intricate graphs. The attention mechanism is also differentiable, which facilitates end-to-end training and enables the model to learn the most efficient attention patterns for the given task (Zhou, Cui, Hu, Zhang, Yang, Liu, Wang, Li & Sun, 2020). In a variety of graph-related tasks, including node classification, link prediction, and graph-level prediction, GATs have exhibited optimistic performance. In domains such as social network analysis, recommendation systems, and biological network analysis, their effectiveness in capturing graph structure and learning meaningful node representations has been proven.

2.5.3 Graph sample and aggregated. GraphSAGE is a resilient framework for graph neural networks that enables learning representations on massive amounts graphs by sampling and aggregating local neighborhood information. It overcomes the limitations of conventional graph convolutional networks (GCNs) by employing a flexible sampling strategy and an aggregating mechanism that can capture the rich structural and feature information of a graph (Hamilton, Ying & Leskovec, 2017). The core concept of GraphSAGE is to generate node representations by sampling and aggregating information from a node's k-hop neighborhood, where k represents the number of steps between the target node and the sample node. Unlike GCNs, which operate on fixed neighborhood structures, GraphSAGE permits adaptive sampling of various neighborhood sizes and configurations, allowing the model to effectively capture both local and global graph information (Ying, He, Chen, Eksombatchai, Hamilton & Leskovec, 2018). GraphSAGE uses a trainable aggregator function that accepts in feed the attributes of a node and its sampled neighboring nodes and generates a fixed-length representation for the target node to conduct the sampling and aggregation process. The aggregator function can be implemented using a variety of mechanisms, such as mean aggregation, max pooling, and LSTM-based aggregators, allowing for the capture of diverse graph patterns and characteristics (Hamilton & Leskovec 2017).

During the training phase, GraphSAGE learns to aggregate and update node representations iteratively by minimizing a loss function that evaluates how well the model performed on a specific downstream task. The model can encompass both local and global information while retaining the expressive power of the graph structure (Velikovi, Cucurull, Casanova, Romero, Lio, and Bengio, 2018). In a variety of graph-based tasks, including node classification, link prediction, and graph-level prediction, GraphSAGE has demonstrated impressive performance. It has been successfully applied in diverse domains such as social network analysis, recommendation systems, and knowledge graph completion, demonstrating its adaptability and efficacy in capturing the complex dependencies and patterns present in real-world graphs (Liu, ZLuo, Shen, Wang & Tang, 2020). The advantages of GraphSAGE include its capacity to efficiently manage large-scale graphs through neighborhood sampling, its adaptability to capture both local and global information, and its capability to learn expressive representations that generalize well to a variety of downstream tasks. These features make GraphSAGE a valuable instrument for graph-structured data analysis and prediction.

2.5.4 Graph isomporphism networks. Graph Isomorphism Networks (GIN) is a graph neural network (GNN) design that has attracted considerable interest in graph-based applications. GIN, which was introduced by Ying, He, Chen, Eksombatchai and Hamilton in 2018, (Ying, He, Chen, Eksombatchai, Hamilton & Leskovec, 2018) seeks to overcome the limitations of previous GNN models by employing permutation-invariant operations and a learnable aggregation function to ensure the model's efficacy across various graph structures.

The GIN model aggregates and updates node representations iteratively based on neighborhood information. It uses a series of Graph Isomorphism Network layers, each of which consists of two primary steps: aggregation and transformation. GIN calculates the sum of the representations of a node's neighbors, including the node itself, during the aggregation phase. This ensures that the model integrates neighborhood-specific information. GIN employs fully connected layers with activation functions to update the node representations during the transformation step.

Utilizing a global aggregating operation is a defining characteristic of GIN. This operation compiles information from all graph nodes into a fixed-length representation that depicts the global structure of the graph. The aggregated representation is then concatenated with the representations of the nodes and used for subsequent predictions or downstream tasks(Ying, He, Chen, Eksombatchai, Hamilton & Leskovec, 2018).

The GIN model has proven its efficacy in a variety of graph-related applications, including node classification, graph classification, and graph generation. It has demonstrated competitive performance in comparison to other state-of-the-art GNN models, indicating its capacity to recognize complex graph patterns and relationships.

2.5.5 Residual gated graph convolutional network. Residual Gated Graph Convolutional Network (RGGCC) is a graph neural network (GNN) framework intended to model graph-structured data and identify intricate relationships within graphs. RGGCC integrates the ideas of residual connections and gated mechanisms to improve the process of representation learning.

Inspired by residual networks in image classification, to facilitate the flow of data through numerous layers of the graph convolutional network. Residual connections enable the model to bypass certain layers while retaining crucial information from previous layers, which mitigates the vanishing gradient problem and enhances the training procedure (Bresson & Laurent , 2017).

In addition, RGGCC employs gated mechanisms, such as the gate activation function, to control the flow of information and selectively update node representations. The gate activation function, which is applied to the output of each graph convolutional layer, determines how much information is propagated to the next layer. By selectively filtering and updating information, the model is able to concentrate on pertinent features while discarding noise or irrelevant data.

RGGCC's combination of residual connections and gated mechanisms enables the model to encompass both local and global dependencies within the graph. It improves the

model's performance on various graph-related tasks, such as node classification, graph classification, and link prediction, by enhancing the model's ability to learn more expressive graph representations (Bresson & Laurent, 2017).

Chapter 3

Research Methodology

3.1 Research Design

This study will employ a quantitative research methodology to examine the efficacy of graph-based fraud detection methods in blockchain-based cryptocurrencies. Data collection and analysis will be required to achieve the research objectives. This study's primary objective is to evaluate the effectiveness and applicability of graph-based techniques for detecting and preventing fraud in blockchain-based cryptocurrencies. In the study, analogous studies, as well as their methods and outcomes, will be examined, followed by the determination of the dataset and the application of preparatory procedures to the dataset. After determining the methods related to the methods to be applied, the outcomes will be compared. The purpose of this study is to assess the precision, efficacy, and scalability of these methodologies for detecting fraudulent transactions.

3.2 Evaluation Metrics

In the field of data science, it is essential to evaluate the performance of machine learning models in order to determine their effectiveness and make informed decisions. Evaluation metrics provide quantitative measurements for assessing the performance of models across a variety of tasks and data sets.

Precision, recall, and the F1 value are essential metrics of assessment for classification tasks. Precision is the proportion of correctly identified positive instances among the predicted positives, whereas recall is the proportion of authentic positive instances correctly identified by the model. (Powers, 2020) The F1 score is a harmonic mean of precision and completeness, incorporating both precision and recall.

Popular evaluation metrics quantify the proportion of correct predictions made by a model relative to the total number of instances. While accuracy provides a general overview of model performance, it can be misleading in datasets where one class predominates.

Consequently, precision must be evaluated alongside other metrics (Sokolova & Lapalme, 2009).

When there is an imbalance between classes, the macro average and weighted average are useful evaluation metrics. The macro average determines the average performance across all divisions independently and uniformly. In contrast, the weighted average takes into consideration the support of each class to account for class distribution. These metrics provide insight into the model's performance across distinct classes, taking into account imbalanced datasets (Sokolova & Lapalme, 2009).

Graphical representations of a model's classification performance are the Receiver Operating Characteristic (ROC) curve and the Precision-Recall (PR) curve. The ROC curve compares the true positive rate (sensitivity) to the false positive rate (1 - specificity), while the PR curve depicts the tradeoff between recall and precision. Area Under the Curve (AUC) is a summary of the performance of the model, with larger values indicating a greater capacity for discrimination. (Fawcett, 2006).

Evaluation metrics play a crucial role in data science, enabling the assessment of model performance and facilitating decision-making. Precision, recall, F1 score, accuracy, macro average, weighted average, ROC curve, precision-recall curve, and area under the curve (AUC) constitute an exhaustive set of metrics for evaluating classification models across multiple domains. Understanding these evaluation metrics enables data scientists to interpret and compare model performance with precision, thereby improving decision-making in real-world applications.

3.3 Dataset & Preparation

The research will involve collecting a comprehensive dataset of blockchain transactions, including transaction attributes and network structures. The dataset will include both fraudulent and legitimate transactions to enable comparative analysis.

As the most prominent application of blockchain technology, cryptocurrencies face substantial financial losses from phishing schemes. Accounts and transactions on the Ethereum blockchain are termed as nodes and edges, correspondingly, in our research,

allowing us to formulate the detection of fraudulent accounts as a classification problem for nodes.

This dataset enables researchers to investigate and resolve the obstacles posed by phishing activities on the Ethereum blockchain network. By leveraging the network structure and supplied attributes, researchers can investigate various methods for detecting and mitigating phishing scams, thereby contributing to the ecosystem's security and integrity.

The resulting dataset contains 2,973,489 nodes, each of which corresponds to a unique Ethereum address in the transaction network. Moreover, the dataset contains 13,551,303 edges, which represent transactions between these Ethereum addresses. Among the total number of nodes, 1,165 have been labeled as phishing nodes or non-phishing nodes. The MulDiGraph.pkl-formatted dataset is presented as a networkx object contained within a pickle file. Every node in the graph corresponding to an Ethereum address and contains a "isp" attribute indicating whether the node represents a fraudulent account. The edges of the graph have two attributes: "amount," which represents the balance of the associated transaction, and "timestamp," which represents the transaction's timestamp. Notably, the dataset includes a mean degree of 4.5574, emphasizing the average number of connections or transactions per Ethereum address.

A subgraph is extracted from the transaction network using a technique similar to (Wu, Qi, Dan, You, Chen, Chen, & Zheng, 2020) because of the discrepancy of the node label. Afterwards, we taken neighboring inbound and outgoing nodes from every one of the randomly selected nodes. Then, a subgraph consisting of only the chosen nodes and half of their neighbors is extracted. The extracted subgraph contains 5,969 nodes and 83,512 edges.

Certain nodes in this transaction network collection have distinct responsibilities from those of a typical account (address). A small number of nodes represent alternative categories, such as exchange accounts, token contracts, etc., while the vast majority represent ordinal accounts. Despite the fact that these account node categories have no direct connection to phishing fraud detection applications because they cannot be labeled "fraud

accounts," we believe the information will be indicative of suspicious transactions and account detection. The timestamps of the edges that represent transactions in financial transaction networks are crucial for machine learning. We randomly selected 80% of the nodes for training and the remaining nodes for testing based on the transaction timestamp.We implement a node classification assignment for account fraud detection, so we must assign each node to training and testing data. As input for every GNN model, we define the node feature set in Table 1. In addition to each node's transaction volume and periodicity, graph analytics (PageRank and degree distributions) yield additional characteristics. This table presents a collection of base node characteristics that can be used to analyze account nodes in a network or graph. These characteristics provide valuable insight into the transactional behavior and network centralization of individual account nodes. Each characteristic is defined as follows:

- send_num: This feature represents the count of transactions sent from the specific account node. It quantifies the level of outgoing transactional activity associated with the account node.

- recv_num: This feature denotes the count of transactions received by the account node. It quantifies the level of incoming transactional activity associated with the account node.

- send_amount: This feature indicates the total amount (e.g., monetary value) sent from the account node. It provides insights into the financial activity associated with the account.

- recv_amount: This feature indicates the total amount (e.g., monetary value) received by the account node. It provides insights into the financial activity associated with the account.

- pagerank: This feature represents the PageRank score assigned to the account node. PageRank is a commonly used method that assesses the significance or centrality of a network node. The pagerank score indicates the significance or

influence of the account node within the network.

These base node features described on Table serve as valuable metrics for analyzing the transactional behavior, financial flows, and network position of individual account nodes within a larger graph or network structure. Researchers and analysts can gain deeper insights into the behavior and significance of account nodes in domains including finance, social networks, and web graphs by leveraging these features.

Table 1

Base Node Features

Feature Name	Description
send_num	It represents the transaction count sent from the account node. It indicates the number of transactions initiated by the account.
recv_num	It represents the transaction count received by the account node. It indicates the number of transactions received by the account.
send_amount	It represents the total amount sent from the account node. It indicates the cumulative value of all the transactions initiated by the account.
recv_amount	Total amount received by this account node It represents the total amount received by the account node. It indicates the cumulative value of all the transactions received by the account.
pagerank	It represents the PageRank score of the account node. PageRank is an algorithm that measures the importance or centrality of a node in a graph. In the context of fraud detection, it can be used to identify accounts that have a higher influence or involvement in the network.

3.4 Graph Based Models

3.4.1 GCN. Graph Convolution Networks (GCNs) are comprised of multiple essential components that process graph-structured data and store meaningful representations. Each

node corresponds to an entity and contains information regarding its features. Adjacency matrices or adjacency lists, which capture the relationships between vertices, are typical representations of the graph structure. The central operation of GCNs is graph convolution, which accumulates information from adjacent nodes to amend the representation of each node. This operation combines node attributes and learned weights to generate novel representations that incorporate both local and global structural information. GCNs use a message transit scheme that transmits data between nodes via the graph's edges. Using the adjacency matrix or adjacency list, the features of contiguous nodes are aggregated in each layer. This aggregation procedure encapsulates the influence and dependencies of neighboring nodes on the target node. After graph convolution, an activation function is applied to the node representations in order to induce nonlinearities. ReLU, sigmoid, and tanh are prevalent functions. The activation function ensures that the representations can capture complex relationships and display nonlinear behavior. GCNs typically consist of numerous layers stacked on top of one another. Each layer performs graph convolution followed by activation, which enables the network to capture data from multiple graph hops. Stacking multiple layers enables the model to acquire increasingly complex representations by integrating information from a larger community. Information traverses network layers during the transfer of information forward. After applying the activation function, the graph convolution operation combines the characteristics of adjacent nodes within each layer. The revised depictions include progressively more structural detail. Following the throw to the forward, GCNs use backward propagation to compute grades and modify model parameters. The gradients are computed with respect to a specified loss function, and optimization algorithms such as stochastic gradient descent (SGD) or Adam are used to modify the weights. This process enables the network to acquire optimal representations for the given task. (Kipf & Welling, 2017).

The convolution layer is a crucial part of the Graph Convolutional Network (GCN) architecture. It is responsible for aggregating and revising node representations by considering information about neighboring nodes. The number of neurons in a GCN is proportional to the dimensionality of node representations or the length of output feature vectors. Each node in the graph represents a feature vector, and the number of neurons represents the length or dimension of each feature vector. The number of neurons in a GCN conv layer is typically determined by the desired representation capacity and the complexity

of the problem. The input geometry of a Graph Convolutional Network (GCN) typically depends on its implementation and library. The input geometry of a GCN is typically a tensor or matrix representing the graph structure and node characteristics.

Depending on the representation employed, the geometry of a graph structure component can vary. Adjacency matrices are frequently used to depict the structure of a graph, with each element representing the presence or absence of an edge between any two vertices. The shape of the adjacency matrix is (N, N), where N is the number of vertices in the graph. Within the context of Graph Convolutional Networks (GCN), ReLU refers to the Rectified Linear Unit activation function. ReLU is a frequent nonlinear activation function used in neural networks such as GCNs to introduce nonlinearity and encode complex data relationships. The dropout layer is a regularization technique used in Graph Convolutional Networks (GCNs) to prevent overfitting and improve the model's generalizability. Dropout is a common technique in neural networks that can also be implemented in GCNs. The dropout layer sets a random fraction of input values to zero during training. This entails that the layer "drops out" a portion of the node characteristics or activations, requiring the model to acquire more robust and generalized representations. Typically, during testing and inference, the dropout layer is disabled and all input values are utilized. In general, when discussing Graph Convolutional Networks (GCN), the expression "sigmoid" refers to the activation function used within the GCN architecture. The sigmoid activation function is a common non-linear activation function that compresses input values to the range 0 to 1. In the illustration, Figure 1 shown above, the input graph is passed through multiple graph convolutional layers that alter the node representations. The refined representations are then applied to task-specific layers, such as node or graph classification. Applying the appropriate output layers for the specific task yields the final output.

This application flow, along with the accompanying illustration, illustrates how GCNs process and learn from graph-structured data. It can be used to explain the fundamental steps involved in applying GCNs and comprehending their potential in tasks involving graph representation learning.

Graph Convolution Networks (GCNs) are comprised of multiple essential components

that process graph-structured data and store meaningful representations. Each node corresponds to an entity and contains information regarding its features. Adjacency matrices or adjacency lists, which capture the relationships between vertices, are typical representations of the graph structure. The central operation of GCNs is graph convolution, which accumulates information from adjacent nodes to amend the representation of each node. This operation combines node attributes and learned weights to generate novel representations that incorporate both local and global structural information. GCNs use a message transit scheme that transmits data between nodes via the graph's edges. Using the adjacency matrix or adjacency list, the features of contiguous nodes are aggregated in each layer. This aggregation procedure encapsulates the influence and dependencies of neighboring nodes on the target node. After graph convolution, an activation function is applied to the node representations in order to induce nonlinearities. ReLU, sigmoid, and tanh are prevalent functions. The activation function ensures that the representations can capture complex relationships and display nonlinear behavior. GCNs typically consist of numerous layers stacked on top of one another. Each layer performs graph convolution followed by activation, which enables the network to capture data from multiple graph hops. Stacking multiple layers enables the model to acquire increasingly complex representations by integrating information from a larger community. Information traverses network layers during the transfer of information forward. After applying the activation function, the graph convolution operation combines the characteristics of adjacent nodes within each layer. The revised depictions include progressively more structural detail. Following the throw to the forward, GCNs use backward propagation to compute grades and modify model parameters. The gradients are computed with respect to a specified loss function, and optimization algorithms such as stochastic gradient descent (SGD) or Adam are used to modify the weights. This process enables the network to acquire optimal representations for the given task. (Kipf & Welling, 2017).

The convolution layer is a crucial part of the Graph Convolutional Network (GCN) architecture. It is responsible for aggregating and revising node representations by considering information about neighboring nodes. The number of neurons in a GCN is proportional to the dimensionality of node representations or the length of output feature

vectors. Each node in the graph represents a feature vector, and the number of neurons represents the length or dimension of each feature vector. The number of neurons in a GCN conv layer is typically determined by the desired representation capacity and the complexity of the problem. The input geometry of a Graph Convolutional Network (GCN) typically depends on its implementation and library. The input geometry of a GCN is typically a tensor or matrix representing the graph structure and node characteristics.

Depending on the representation employed, the geometry of a graph structure component can vary. Adjacency matrices are frequently used to depict the structure of a graph, with each element representing the presence or absence of an edge between any two vertices. The shape of the adjacency matrix is (N, N), where N is the number of vertices in the graph. Within the context of Graph Convolutional Networks (GCN), ReLU refers to the Rectified Linear Unit activation function. ReLU is a frequent nonlinear activation function used in neural networks such as GCNs to introduce nonlinearity and encode complex data relationships. The dropout layer is a regularization technique used in Graph Convolutional Networks (GCNs) to prevent overfitting and improve the model's generalizability. Dropout is a common technique in neural networks that can also be implemented in GCNs. The dropout layer sets a random fraction of input values to zero during training. This entails that the layer "drops out" a portion of the node characteristics or activations, requiring the model to acquire more robust and generalized representations. Typically, during testing and inference, the dropout layer is disabled and all input values are utilized. In general, when discussing Graph Convolutional Networks (GCN), the expression "sigmoid" refers to the activation function used within the GCN architecture. The sigmoid activation function is a common non-linear activation function that compresses input values to the range 0 to 1. In the illustration, Figure 1 shown above, the input graph is passed through multiple graph convolutional layers that alter the node representations. The refined representations are then applied to task-specific layers, such as node or graph classification. Applying the appropriate output layers for the specific task yields the final output.

This application flow, along with the accompanying illustration, illustrates how GCNs process and learn from graph-structured data. It can be used to explain the fundamental steps

involved in applying GCNs and comprehending their potential in tasks involving graph representation learning.

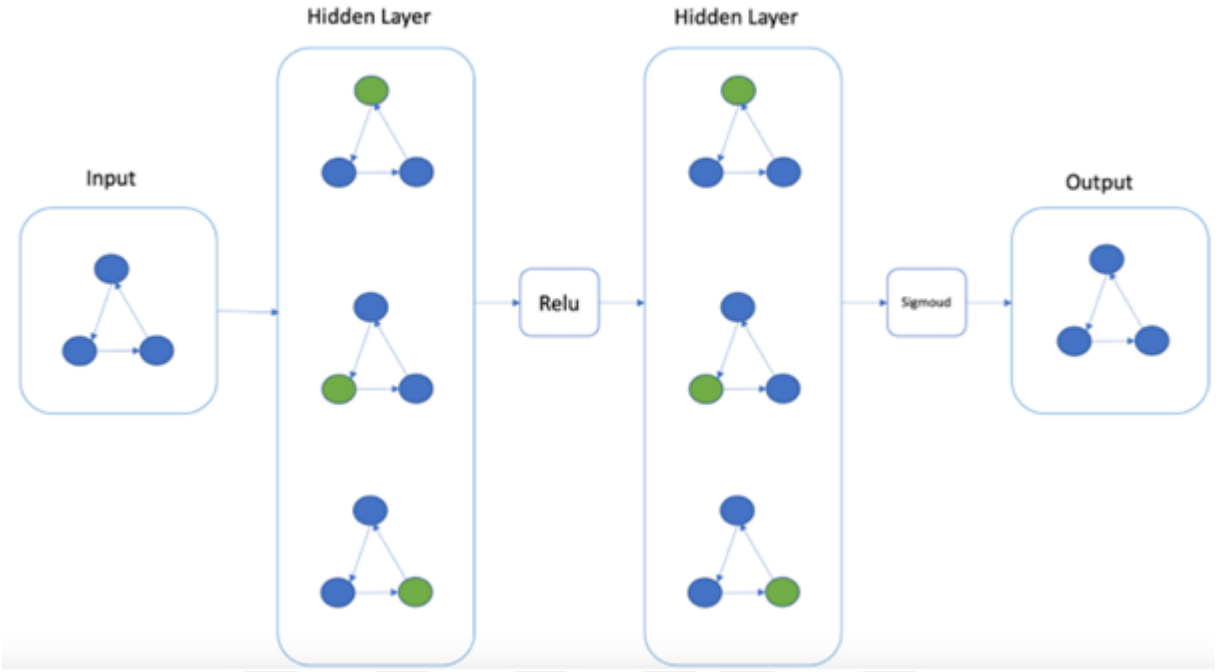

Figure 1. GCN Application Flow.

3.4.2 GAN. Each node in the graph is associated with a feature vector in the GAT conv layer. The GAT conv layer computes attention coefficients for each node and its neighbors using a self-attention mechanism. Attention coefficients indicate the significance or relevance of neighboring nodes to the target node. Typically, attention is computed utilizing a shared learnable attention mechanism that allocates weights to nodes based on their feature similarity. The attention coefficients are then used to weight the neighboring nodes' feature vectors. The feature vectors are multiplied by the attention coefficients in order to emphasize or minimize the contributions of various neighbors based on their relative significance. This procedure enables the GAT to focus on various portions of the graph based on the importance of the nodes. The weighted feature vectors of neighboring nodes are then aggregated to create a new representation for every node.

Methods of aggregation may include calculating a weighted sum, utilizing mean or maximum aggregating, or employing more intricate attention mechanisms. Following the

aggregation step, a non-linear activation function, such as ReLU or LeakyReLU, is employed to introduce non-linearity and capture complex data patterns. The activated features are used to update the graph's node representations. It refers to the dimensionality of the node representations or the size of the output feature vectors when discussing the number of neurons in a GAT conv layer. Each node in the graph corresponds to a feature vector, and the number of neurons indicates the length or dimension of these feature vectors. Typically, the number of neurons in a GAT conv layer is determined by the intended representation capacity and the problem's complexity. The number of neurons in a GAT conv layer is determined by variables such as the magnitude of the input feature vectors, the complexity of the graph data, and the specific task being performed. It can be determined through experimentation and refining in accordance with the model's performance and requirements. Node features, which define the characteristics or attributes associated with each node in the graph, are the primary input to a GAT. Depending on the specific dataset and application, the dimension of the node's attributes can vary. For instance, if each node is represented by a feature vector of length d, then the typical input format for node features would be (N, d), where N is the number of nodes in the graph (Velickovic, P., Rieke, N., & Welling, M. , 2021).

The illustration Figure 2 depicts the sequential passage of data through the GAT model, beginning with the input graph and passing through the graph attention layers, attention mechanisms, aggregation steps, and task-specific layers before producing the output. This flow depicts the process of learning node representations while taking into account the significance of various nodes and their connections using GATs' attention mechanisms.

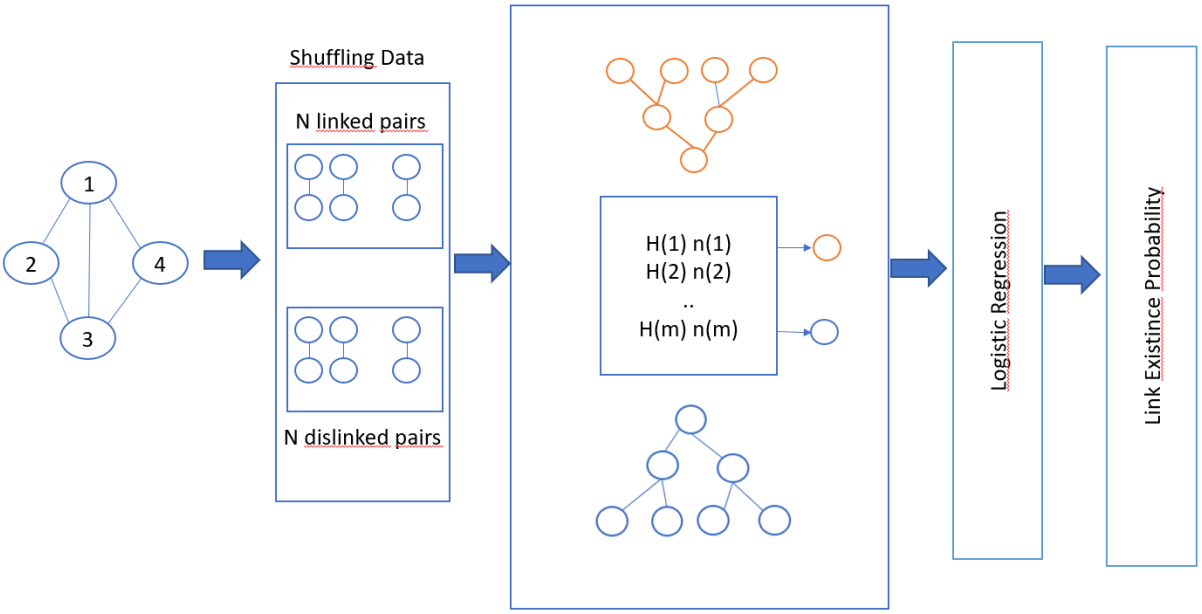

Figure 2. GAT Application Flow.

3.4.3 Graphsage. GraphSAGE (Graph Sample and Aggregated) is an algorithm for learning graph representations that seeks to generate node embeddings by aggregating information from the node's local neighborhood. It uses the graph structure to capture relational information between nodes and to discover evocative node representations. GraphSAGE samples a neighborhood of fixed size surrounding each node in the graph. It selects a subset of neighboring nodes according to predefined sampling strategies such as uniform sampling, random walks, or customized PageRank. The sampled neighborhood nodes are then aggregated to produce a representative embedding for the central node. To consolidate the information from the neighbors, aggregation functions such as mean aggregation, maximum pooling, and attention mechanism are used.

The aggregated embeddings and the current node's own features are combined to alter the node's representation. This updated representation is then employed for subsequent operations such as node classification, link prediction, and anomaly detection. An initial embedding or feature vector is assigned to each node in the graph.For each node, a neighborhood of fixed size is sampled, and the neighboring nodes' embeddings are aggregated. The aggregation function encapsulates the neighborhood's structure and characteristics to produce a representative embedding for the central node. The aggregated embedding is combined with the node's own features, and the updated representation is used

to learn a predictive model by employing supervised or unsupervised learning techniques. The important advantage of GraphSAGE is its ability to learn node embeddings using local neighborhood information, allowing the algorithm to capture both structural and attribute-based graph relationships.

In the figure above Figure 3, the input graph is subjected to neighbor sampling, in which a subset of each node's neighbors is selected. The features of the sampled neighbors are then aggregated to create a representation for each node. This procedure is repeated across multiple GraphSage layers in order to refine node representations. To perform subsequent tasks, task-specific layers are added, and the final output is dependent on the specific task requirements. The illustration depicts the sequential flow of data through the GraphSage model, highlighting the neighbor sampling, aggregation, and subsequent layers for executing various graph-related tasks. This flow encapsulates the essence of GraphSage's ability to learn expressive node representations by considering each node's immediate neighborhood.

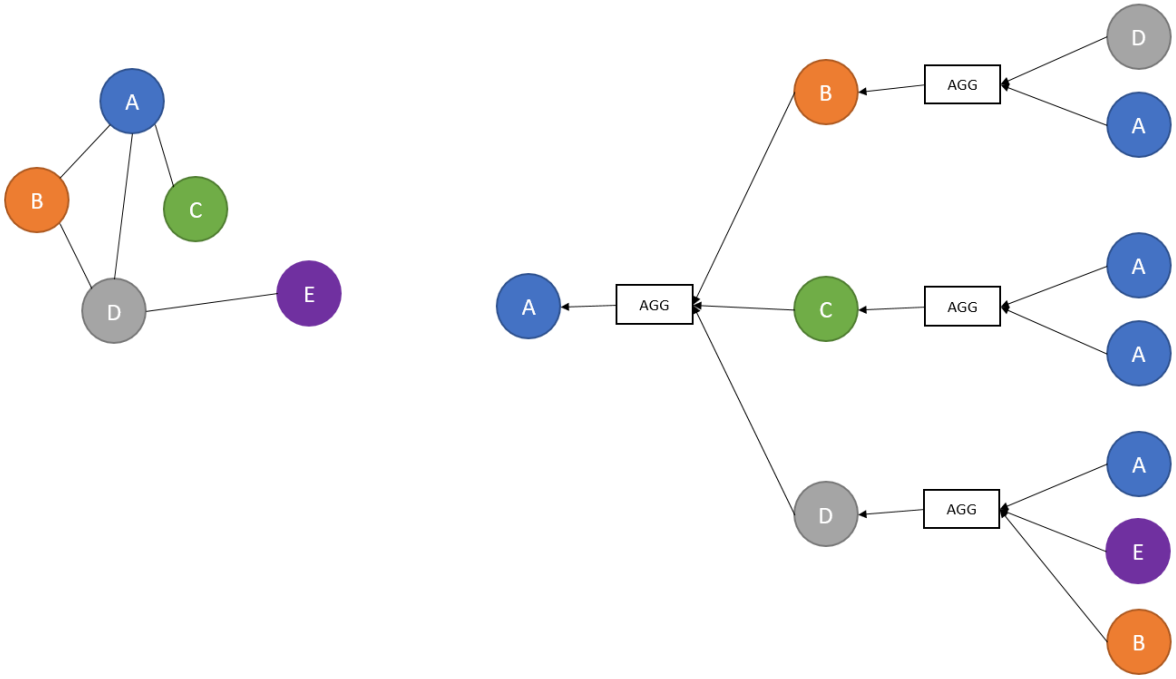

Figure 3. GraphSAGE Application Flow.

3.4.4 Gin . The Graph Isomorphism Network (GIN) architecture is a graph neural network (GNN). GIN is designed to learn node representations that are independent of the ordering of adjacent nodes and are capable of capturing global graph structure.GIN's structure and operation can be described as follows:

- Message Passing: GIN employs the message passing paradigm typical of GNNs. Each node in the graph collects information from its companions through a step of message passing. In contrast to other GNNs, GIN does not rely on the structure of message-passing mechanisms such as graph convolutions or graph attention. GIN instead employs a more generic aggregation procedure.

- Aggregation Operation: GIN aggregates using a symmetric sum operation. At each layer, each node contributes its own representation to the sum of the representations of its neighbors. This symmetric aggregation guarantees that the order of the neighbors does not influence the representation of the resulting node. The aggregation operation can be defined mathematically as follows: h_il represents the node representation at layer l for node i, MLPl is a multi-layer perceptron (MLP) applied element-wise, N(i) is the set of neighboring nodes of node i, and epsilon and lambda are learnable parameters that balance the self-information and neighbor information.

$$h_i^l = MLP^l\left(\left(1 + \epsilon\lambda]h_i^{l-1} + \sum_{j\epsilon N_{(i)}} h_j^{l-1}\right)\right)$$

- Readout Function: GIN uses a readout function to combine the representations of all graph nodes into a singular graph-level representation. Typically, the output function is a permutation-invariant operation such as summation or mean pooling. This graph-level representation can be utilized for subsequent tasks like graph classification.
- Depth and Stacking: GIN elements can be stacked to enhance the model's depth and expressiveness. Multiple GIN layers stacked together allow the model to

capture higher-order dependencies and more complex graph structures.

- Predictions at the Graph and Node Levels: GIN can be used for both graph-level and node-level prediction assignments. For graph-level tasks, the graph-level representation derived from the readout function is fed into fully connected layers, followed by the appropriate activation functions and output layers, in order to generate predictions. For node-level tasks, the final GIN layer's node representations can be employed for node classification, link prediction, and other node-level predictions.
- The key strength of GIN lies in its theoretical power. Study(Xu, Hu, Leskovec & Jegelka, 2018) showed that GIN with a sufficiently large MLP and enough graph convolutional layers can approximate any permutation-invariant function on graphs, making it a powerful graph representation learning framework.

3.4.5 Rggcn. Residual Gated Graph Convolutional Network (RGGCN) is a graph neural network architecture proposed by Bresson, X. and Laurent, T. (Bresson, X. & Laurent, T., 2017). RGGCN employs residual connections and gate mechanisms to enhance the learning of node representations from graph-structured data. The first stage in deploying RGGCN entails defining and representing the graph of input. In a graph, nodes (vertices) and edges (connections) represent the connections between nodes. Each node in the graph is associated with a feature vector containing data unique to that node. Using graph convolutional layers, RGGCN aggregates data from the neighborhood of a node. Each convolutional graph layer receives as input the node features and the adjacency matrix. Each RGGCN graph convolutional layer comprises residual connections. Residual connections allow the model to bypass the convolutional operation by propagating the input node features directly to the output of the layer. This alleviates the problem of vanishing gradients

and enhances the network's information flow. mRGGCN implements a gate mechanism to control the flow of data through residual connections. As part of the gate mechanism, a gating function generates a gate value between 0 and 1 for each node. This gate value is multiplied element-by-element with the input features of the residual connection, allowing the model to selectively adjust node representations. Following residual connections, an activation function, such as ReLU or sigmoid, is implemented to incorporate nonlinearity and improve the model's expressive capacity. RGGCN can be layered with multiple graph convolutional layers in order to capture higher-order dependencies and more intricate graph structures. The output of one layer serves as the input for the subsequent layer, allowing the model to obtain increasingly generalized representations. The ultimate output of the RGGCN model depends on the specific assignment. It can be used for classification of nodes, prediction of links, and classification of graphs. Backpropagation and gradient descent are used to optimize the model's parameters so as to minimize the loss between the predicted output and the labels of the ground truth.

Chapter 4

Findings

4.1 Introduction

The purpose of our experiments was to evaluate and compare the efficacy of three graph-based methodologies: Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), Graph Sample and Aggregated(GraphSAGE), Graph Isomorphism Networks (GIN) and Residual Gated Graph Convolutional Network(RGGCN). We present the results of our experiments. We evaluate their ability to detect fraud in blockchain-based cryptocurrencies using an exhaustive dataset of actual transactions.

Multiple evaluation metrics were employed to assess the performance of each fraud detection technique. These metrics consist of precision, recall, F1 score, and support. Precision is the proportion of correctly identified fraudulent transactions among all fraudulent transactions predicted. Recall measures the ability to identify fraudulent transactions precisely among all actual fraudulent transactions. The F1 score is the harmonic mean of precision and recall and represents a balanced evaluation of performance as a whole. In a dataset, support is the number of instances in each class (fraudulent and legitimate transactions).

This study utilized two convolution layers with 32 neurons, a five-column input shape, and a two-column output shape. The initial convolution layer was followed by the application of the relu activation function with a dropout probability of 0.5. After a second layer was applied, the sigmoud output function was implemented. With a 0.02 learning rate and a vertical weight parameter of 0.0005, the Adam optimizer is utilized. Cross-entropy is utilized for model training as a lost function. Classweights are computed based on the number of valid and fraudulent nodes and transmitted to the loss function. Within the scope of this investigation, this methodology was applied to every Graph-Based Application.

4.2 GCN Results

Table 2 demonstrates that the Graph Convolutional Network (GCN) model performed extraordinarily well in classifying instances into two categories: "Legit" and "Fraud." Precision, recall, and F1 score provide insight into the model's capacity to classify instances accurately and capture both positive and negative instances. The GCN model achieved 0.93 precision for the "Legit" class, signifying that the vast majority of instances predicted as "Legit" were indeed "Legit." However, the precision for the "Fraud" class was lower, at 0.41, indicating that its predictions contained some false positives. Comparable recall performance was demonstrated by the GCN model, with recall values of 0.78 for the "Legit" class and 0.73 for the "Fraud" class. This demonstrates that the model accurately identified a significant proportion of actual instances within both classes. The F1 score, which incorporates both precision and recall, strengthens the evaluation of performance. The "Legit" class has an F1 score of 0.85, indicating a satisfactory equilibrium between precision and recall. However, the F1 score for the "Fraud" category is lesser at 0.52, indicating that detection of instances belonging to this category needs improvement. The overall accuracy of the GCN model is 0.77, indicating that 77% of instances within the dataset have been correctly classified. This metric provides an all-encompassing view of the model's effectiveness across all classes.

In conclusion, the GCN model's ability to classify instances into the "Legit" and "Fraud" categories is promising. The proportional recall for both classes and high precision for the "Legit" class indicate that the model captures the vast majority of instances within the dataset. However, there is room for development in the "Fraud" category in terms of precision and F1 score. It may be necessary to further refine and optimize the model so that it can detect instances of deception more precisely. These findings contribute to the understanding and applicability of graph neural networks, such as GCN, for fraud detection and related tasks in numerous domains.

Table 2

Results of GCN Application

	Precision	Recall	F1 Score	Support
Legit	0.93	0.78	0.85	844
Fraud	0.41	0.83	0.63	179
Accuracy			0.81	1023
Macro Average	0.67	0.82	0.75	1023
Weighted Average	0.84	0.81	0.83	1023

Figure 4 depicts the confusion matrix produced by the GCN (Graph Convolutional Network) application when classifying instances into the "Not-Fraud" and "Fraud" categories. It provides a comprehensive view of the model's performance and can be analyzed further to determine its efficacy.

The matrix discloses the following data:

True Label:

• There are 656 instances correctly designated "Not-Fraud" for the "Not-Fraud" class.

• There are 49 instances of the "Fraud" class that are accurately designated as "Fraud."

Predicted Label:

• Of the instances predicted as "Not-Fraud", 656 are true negatives, indicating they are accurately identified as "Not-Fraud."

• Of the instances forecasted as "Fraud", 188 are false positives, meaning they are incorrectly classified as "Fraud"

• Of the instances predicted as "Not-Fraud," 49 are false negatives, i.e., they have been incorrectly labeled as "Not-Fraud."

• Of the instances predicted as "Fraud", 130 are true positives, accurately designated as "Fraud"

The confusion matrix provides valuable insight into the GCN model's efficacy in classifying instances as "Not-Fraud" or "Fraud" It highlights both the assets and weaknesses of the model. The high number of true negatives and true positives suggests that the GCN model can effectively classify a substantial portion of instances belonging to both classifications. This demonstrates the model's ability to reliably identify "Not-Fraud" and "Fraud" instances. Nonetheless, the prevalence of false positives and false negatives indicates that the model is not flawless and may misclassify some instances. False positives represent instances mistakenly labeled "Fraud", whereas false negatives represent instances incorrectly labeled "Not-Fraud". These misclassifications can have significant repercussions, particularly in the context of detecting fraud, where accurate identification is essential. Using the confusion matrix's values, additional performance metrics such as precision, recall, and F1 score can be calculated to obtain a more comprehensive evaluation of the GCN application. These metrics provide a deeper comprehension of the model's precision, sensitivity, and overall performance in separating the two classes.

In conclusion, the perplexity matrix provides valuable insight into the performance of the GCN application for classifying instances as "Not-Fraud" or "Fraud." False positives and false negatives indicate that there is room for development, despite the model's proficiency in correctly identifying a substantial number of instances. To address these misclassifications and enhance the overall performance of the GCN model in fraud detection scenarios, additional analysis and potential model enhancements can be investigated.

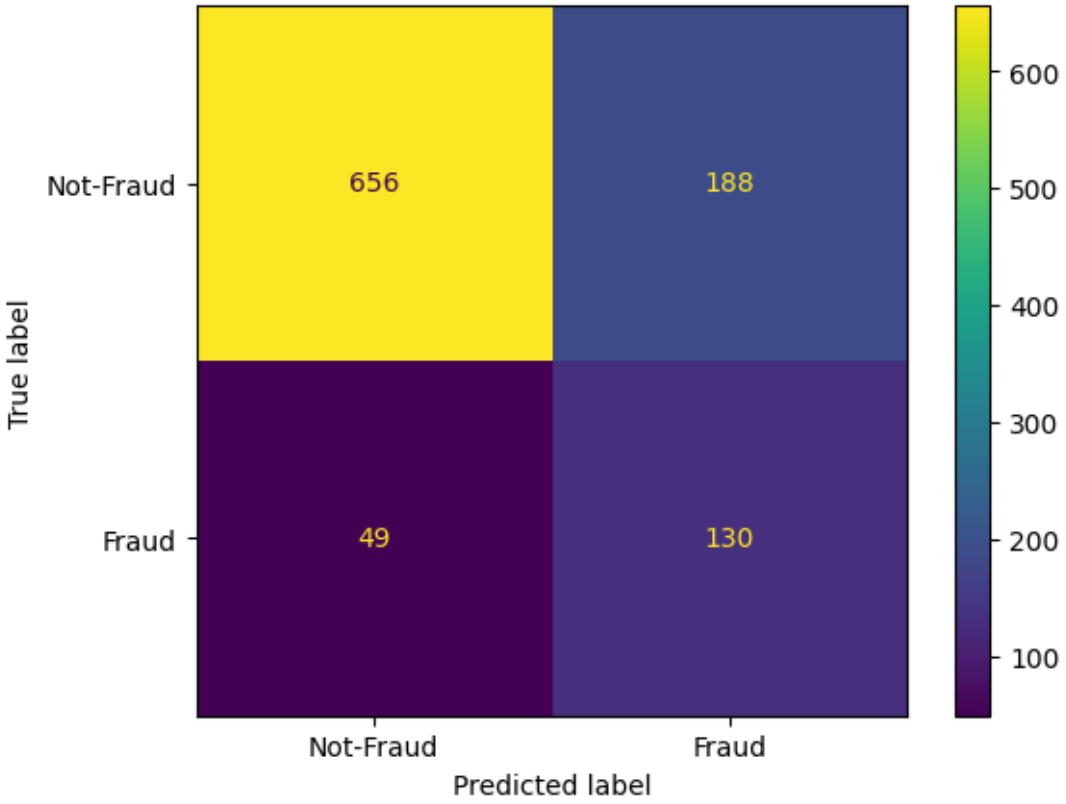

Figure 4. GCN Application Confusion Matrix.

In Figure 5, the relationship between the true positive rate (TPR) or recall and the false positive rate (FPR) at varying threshold settings is depicted. A higher AUC indicates superior classification performance.For the given ROC curve with an AUC of 0.71135, the plot would depict the trade-off between the true positive rate and the false positive rate, indicating the classifier's ability to distinguish between positive and negative classes.

A Precision-Recall curve illustrates the tradeoff between accuracy and recall for various classification thresholds. It is notably useful in situations involving an imbalanced class distribution. The AUC of the Precision-Recall curve quantifies the overall precision and recall performance of the classifier. In the provided Precision-Recall curve with an AUC of 0.39148, the plot illustrates the relationship between precision and recall at various classification thresholds. It provides information regarding the model's ability to correctly identify positive instances (precision) while capturing all pertinent positive instances (recall).

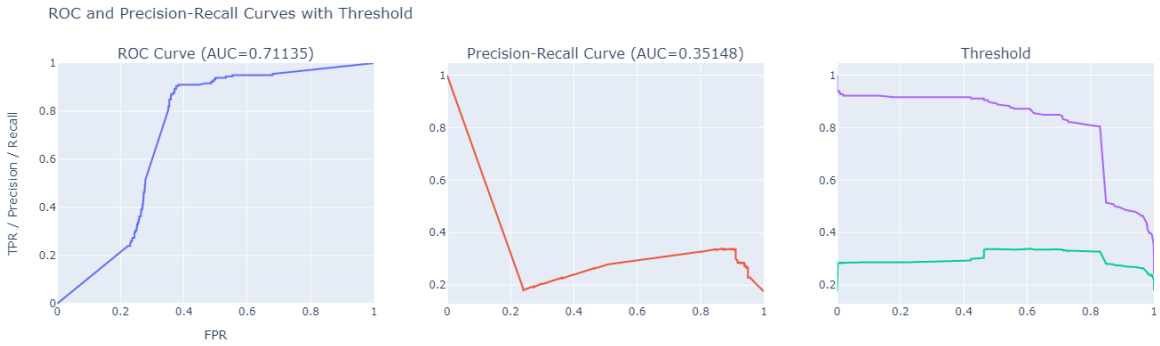

Figure 5. ROC and Precision-Recall Curves with Threshold for GCN

4.3 GAT Results

A number of performance metrics for the classification task are provided by examining the results depicted in Table 3 of the GAT application. The efficacy of the model can be evaluated using precision, recall, F1 score, support, precision, and the macro and weighted averages. The model achieves a precision of 0.96 for the "Legit" category and 0.67 for the "Fraud" category, indicating that precision measures the accuracy of positive predictions. These values indicate that the model accurately identifies "Legit" instances, whereas "Fraud" instances are less precise. Recall, also referred to as sensitivity, assesses the model's ability to identify positive instances with reliability. Recall values of 0.91 for the "Legit" class and 0.84 for the "Fraud" class indicate that the model reliably identifies a substantial proportion of "Legit" and "Fraud" instances. The F1 score, which takes precision and recall into account, is a balanced metric. The class "Legit" has an F1 score of 0.94, which indicates an outstanding balance between precision and recall. The "Fraud" class's F1 score of 0.75 indicates a relatively weak balance between precision and recall for detecting instances of fraud. Support refers to the number of instances in each class, with 844 identified as "Legit" and 179 as "Fraud." These graphs depict the distribution of classes within the dataset. It is reported that the model's overall accuracy is 0.90, indicating that the GAT application correctly classifies approximately 90% of the instances in the dataset. The macro average

and weighted average aggregate measurements for both categories. The average scores for precision, recall, and F1 at the macro level are 0.82, 0.88, and 0.84, respectively. The average weighted precision, recall, and F1 scores are 0.91, 0.90, and 0.90, respectively. These values represent class distribution and summarize the overall efficacy of the model.

In conclusion, the GAT application accurately classifies instances as "Legit" or "Fraud." The model achieves a high level of precision for the "Legit" class, indicating that accurate positive predictions can be made. However, the "Fraud" class has relatively lesser precision, indicating that there is room for improvement in identifying instances of fraud. The relatively high recall values for both classes indicate that the model is capable of capturing a substantial proportion of positive instances. The F1 scores for the "Legit" category demonstrate a comparatively better balance between precision and recall than those for the "Fraud" category. The classification performance of the GAT model is excellent, as indicated by its overall accuracy of 0.90.

Table 3

Results of GAT Application

	Precision	Recall	F1 Score	Support
Legit	0.96	0.91	0.94	844
Fraud	0.67	0.84	0.75	190
Accuracy			0.90	1023
Macro Average	0.82	0.88	0.84	1023
Weighted Average	0.91	0.90	0.90	1023

In Figure 6, the perplexity matrix corresponds to the results of the GAT (Graph

Attention Network) application, namely the classification of instances into the "Not-Fraud" and "Fraud" classes. This matrix provides a comprehensive overview of the model's performance and can be further analyzed to determine its efficacy.

The matrix can be examined in the following manner:

True Label:

• 770 instances of the "Not-Fraud" class are correctly classified as "Not-Fraud."
• 28 instances of the "Fraud" class are accurately classified as "Fraud."

Anticipated Label:

• 770 instances predicted as "Not-Fraud" are true negatives, implying that they have been correctly identified as "Not-Fraud."
• 74 instances forecasted as "Fraud" are false positives, meaning they are incorrectly classified as "Fraud."
• Thirty instances predicted as "Not-Fraud", 28 are false negatives, indicating that they have been incorrectly classified as "Not-Fraud."
• Of the instances forecasted as "Fraud", 151 are true positives, accurately identified as "Fraud."

The confusion matrix provides essential information regarding the GAT model's performance in classifying instances as "Not-Fraud" and "Fraud" It highlights both the model's assets and weaknesses, as evidenced by the high number of true negatives and true positives and the presence of false positives and false negatives, respectively. The high number of true negatives indicates that the model is adept at accurately identifying instances that are "Not-Fraud." Similarly, the high number of true positives indicates that the model can reliably identify instances that are "Fraud" These results demonstrate the model's ability to differentiate between the two classes. The presence of false positives and false negatives, however, indicates that the model is imperfect and may misclassify some instances. False positives are instances that are incorrectly classified as "Fraud", whereas false negatives are instances that are incorrectly designated as "Not-Fraud". These misclassifications have practical ramifications, especially in the context of detecting fraud, where accurate

identification is crucial. Using the confusion matrix's values, additional performance metrics such as precision, recall, and F1 score can be calculated to obtain a more comprehensive evaluation of the GAT application. These metrics provide a deeper comprehension of the model's precision, sensitivity, and overall performance in separating the two classes.

In conclusion, the perplexity matrix provides valuable insight into the performance of the GAT application in classifying instances as "Not-Fraud" or "Fraud." False positives and false negatives indicate that there is room for development, despite the model's proficiency in correctly identifying a substantial number of instances. In order to resolve these misclassifications and improve the overall performance of the GAT model in fraud detection scenarios, additional analysis and potential enhancements can be explored.

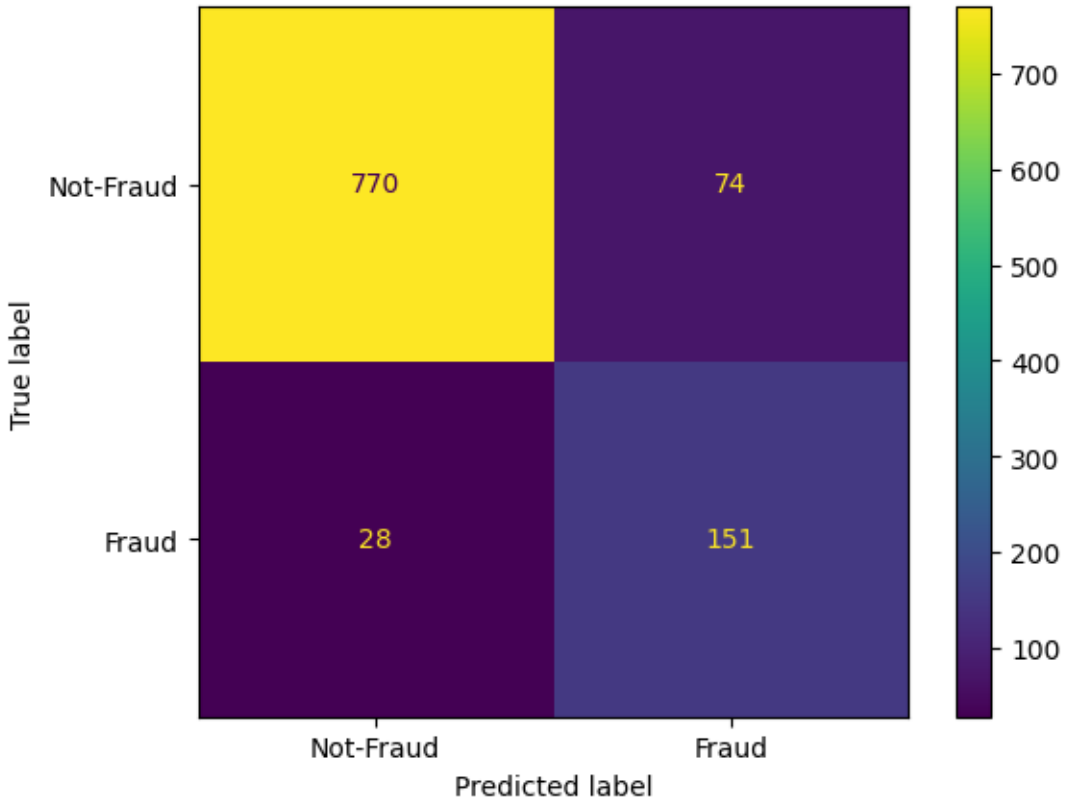

Figure 6. GAT Application Confusion Matrix.

In Figure 7, the relationship between the true positive rate (TPR) or recall and the false positive rate (FPR) at different threshold settings is depicted. A higher AUC indicates superior classification performance. For the provided ROC curve with an AUC of 0.88941, the plot would depict the trade-off between the true positive rate and the false positive rate, indicating the classifier's ability to distinguish between positive and negative classes. A Precision-Recall curve illustrates the tradeoff between accuracy and recall for various classification thresholds. It is notably useful in situations involving an imbalanced class distribution. The AUC of the Precision-Recall curve quantifies the overall precision and recall performance of the classifier. In the provided Precision-Recall curve with an AUC of 0.74229, the diagram would illustrate the relationship between precision and recall at various classification thresholds. It provides information regarding the model's ability to correctly identify positive instances (precision) while capturing all pertinent positive instances (recall).

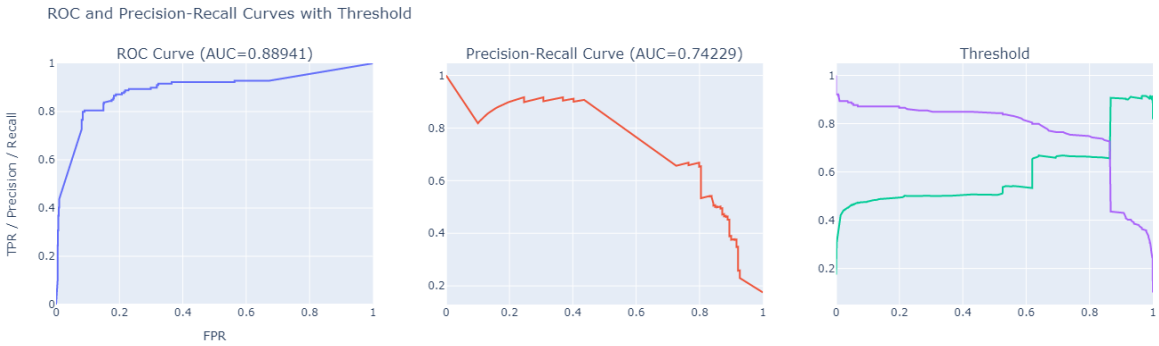

Figure 7. ROC and Precision-Recall Curves with Threshold for GAT

4.4 GraphSage Results

Examining the GraphSage application's Table 4 results enables us to evaluate the efficacy of the model using various metrics. The table contains information regarding precision, recall, F1 score, support, precision, and macro and weighted averages. Precision is the degree to which accurate forecasts are made. The GraphSage model achieves a precision of 0.95 for the "Legit" class, signifying that 95% of instances predicted as "Legit"

are accurate. The "Fraud" class has a precision of 0.49, indicating that 49% of instances predicted as "Fraud" are accurate predictions. Recall, also referred to as sensitivity, assesses the model's ability to identify positive instances with reliability. The model's recall is 0.82 for the "Legit" class and 0.80 for the "Fraud" class. This indicates that 82% of "Legit" instances and 80% of "Fraud" instances are precisely identified by the model. In a singular metric, the F1 score optimizes the trade-off between precision and recall. The "Legit" category's F1 score of 0.88 indicates a reasonable equilibrium between precision and recall. The "Fraud" class's F1 score of 0.61 indicates a relatively weak balance between precision and recall when identifying instances of fraud. 844 instances in the support column are designated "Legit" while 179 instances are labeled "Fraud." The reported accuracy of the model is 0.82, indicating that the GraphSage application classifies approximately 82% of instances in the dataset accurately. The macro average and weighted average are measures of the two divisions as a whole. At the macro level, the average precision, recall, and F1 score are 0.72, 0.81, and 0.74. The weighted mean precision, recall, and F1 score are 0.87, 0.82, and 0.83 respectively. These values accommodate for class distribution and provide an overall evaluation of the model's performance.

In conclusion, the GraphSage application performs well in terms of accurately classifying instances as "Legit" or "Fraud." The model achieves a high level of precision for the "Legit" class, indicating that accurate positive predictions can be made. However, the precision of the "Fraud" class is comparatively lower, indicating that fraud identification could be improved. The relatively high recall values for both classes indicate that the model is capable of capturing a substantial proportion of positive instances. The F1 scores for the "Legit" category demonstrate a comparatively better balance between precision and recall than those for the "Fraud" category. The classification performance of the GraphSage model is excellent, as indicated by its overall accuracy of 0.82.

Table 4

Results of GraphSage Application

	Precision	Recall	F1 Score	Support
Legit	0.95	0.82	0.88	844
Fraud	0.49	0.80	0.61	179
Accuracy			0.82	1023
Macro Average	0.72	0.81	0.74	1023
Weighted Average	0.87	0.82	0.83	1023

The provided confusion matrix in Figure 8, corresponds to the results of the GraphSage application, specifically in classifying instances into the "Not-Fraud" and "Fraud" classes. This matrix offers a comprehensive overview of the model's performance and can be analyzed further to evaluate its effectiveness.

The matrix can be reviewed as follows:

True Label:

- For the "Not-Fraud" class, 694 instances are correctly classified as "Not-Fraud".
- For the "Fraud" class, 36 instances are correctly classified as "Fraud".

Predicted Label:

- Among the instances predicted as "Not-Fraud", 694 instances are true negatives, indicating that they are correctly identified as "Not-Fraud".
- Among the instances predicted as "Fraud", 150 instances are false positives,

implying that they are incorrectly classified as "Fraud".

- Among the instances predicted as "Not-Fraud", 36 instances are false negatives, suggesting that they are incorrectly labeled as "Not-Fraud".
- Among the instances predicted as "Fraud", 143 instances are true positives, correctly identified as "Fraud".

The confusion matrix provides crucial insights into the performance of the GraphSage model in terms of classifying instances as "Not-Fraud" and "Fraud". It highlights both the model's strengths, as evidenced by the high number of true negatives and true positives, and its weaknesses, as indicated by the presence of false positives and false negatives. The high number of true negatives suggests that the model is proficient in correctly identifying instances that are genuinely "Not-Fraud". Similarly, the high number of true positives indicates the model's ability to accurately identify instances that are indeed "Fraud". These findings demonstrate the model's effectiveness in distinguishing between the two classes. However, the presence of false positives and false negatives signifies that the model is not perfect and may misclassify some instances. False positives represent instances that are incorrectly classified as "Fraud", while false negatives represent instances that are incorrectly labeled as "Not-Fraud". These misclassifications have practical implications, particularly in the context of fraud detection, where accurate identification is crucial. To obtain a more comprehensive evaluation of the GraphSage application, additional performance metrics such as precision, recall, and F1 score can be computed using the values from the confusion matrix. These metrics offer a deeper understanding of the model's accuracy, sensitivity, and overall performance in distinguishing between the two classes.

In summary, the provided confusion matrix offers valuable insights into the performance of the GraphSage application in classifying instances as "Not-Fraud" and "Fraud". While the model demonstrates proficiency in correctly identifying a significant number of instances, there is room for improvement, as indicated by the presence of false positives and false negatives. Further analysis and potential enhancements can be explored to address these misclassifications and improve the overall performance of the GraphSage model in fraud detection scenarios.

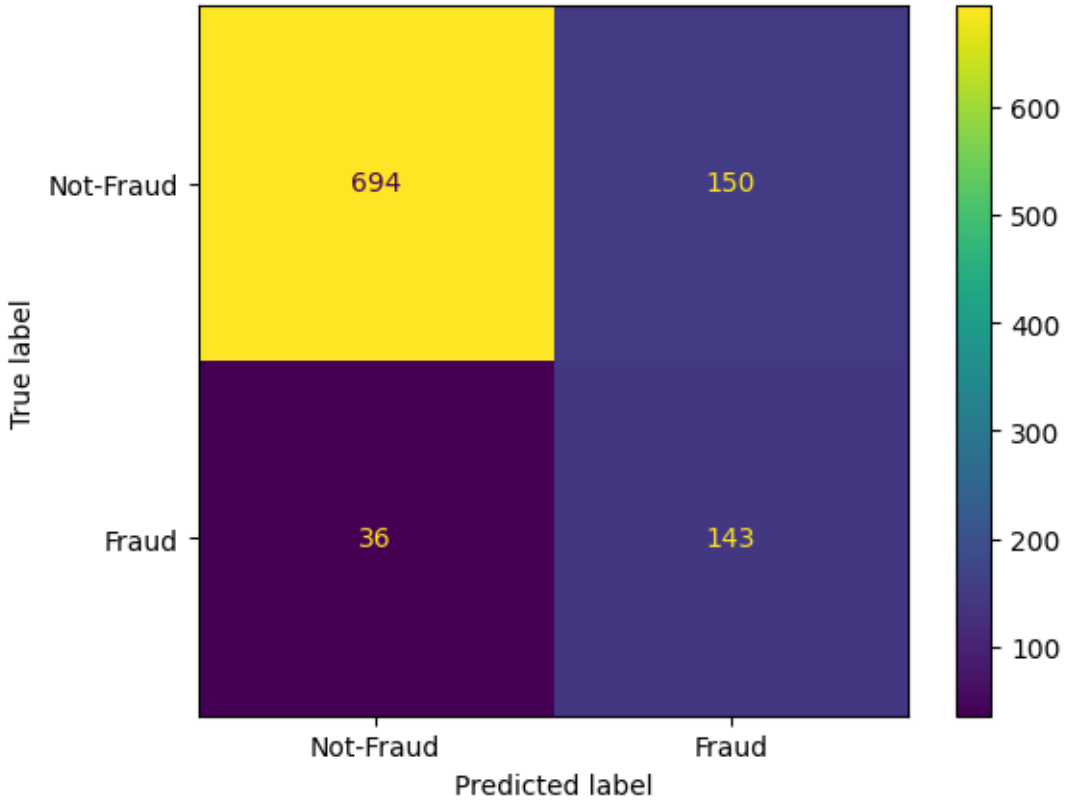

Figure 8. GraphSage Application Confusion Matrix.

In Figure 9, the relationship between the true positive rate (TPR) or recall and the false positive rate (FPR) at various threshold settings is illustrated. A higher AUC indicates superior classification performance. For the provided ROC curve with an AUC of 0.84730, the plot would depict the trade-off between the true positive rate and the false positive rate, indicating the classifier's ability to distinguish between positive and negative classes. A Precision-Recall curve illustrates the tradeoff between accuracy and recall for various classification thresholds. It is notably useful in situations involving an imbalanced class distribution. The AUC of the Precision-Recall curve quantifies the overall precision and recall performance of the classifier. In the provided Precision-Recall curve with an AUC of 0.58801, the diagram would illustrate the relationship between precision and recall at various classification thresholds. It provides information regarding the model's ability to correctly identify positive instances (precision) while capturing all pertinent positive instances (recall).

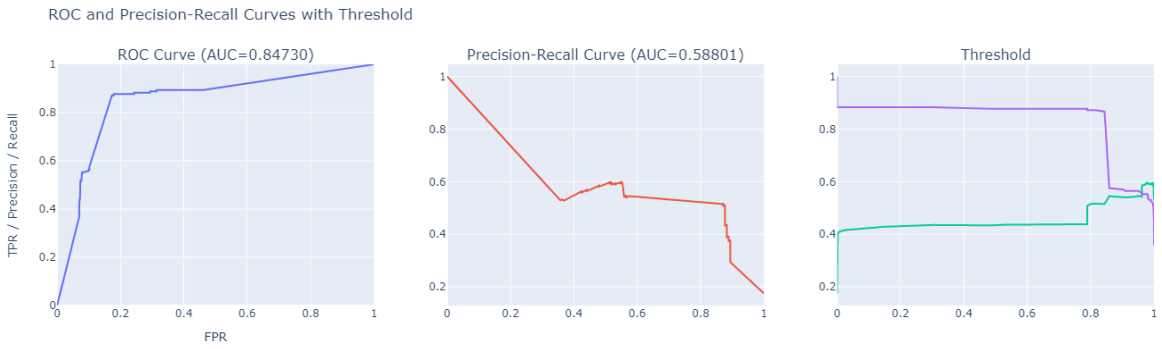

Figure 9. ROC and Precision-Recall Curves with Threshold for GraphSage

4.5 GIN Results

Reviewing Table 5's GIN (Graph Isomorphism Network) application results, we can evaluate the efficacy of the model using various metrics. The table contains information regarding precision, recall, F1 score, support, precision, and macro and weighted averages. Precision is the degree to which accurate forecasts are made. The GIN model achieves a precision of 0.94 for the "Legit" class, indicating that 94% of instances that are predicted as "Legit" are accurate. The "Fraud" class has a precision of 0.52, indicating that 52% of instances predicted as "Fraud" are accurate predictions. Recall, also referred to as sensitivity, assesses the model's ability to identify positive instances with reliability. The model's recall is 0.85 for the "Legit" class and 0.74 for the "Fraud" class. This indicates that 85% of "Legit" instances and 74% of "Fraud" instances are reliably identified by the model. In a singular metric, the F1 score optimizes the trade-off between precision and recall. The "Legit" category's F1 score of 0.89 indicates a reasonable equilibrium between precision and recall. The "Fraud" class's F1 score of 0.61 indicates a relatively weak balance between precision and recall when identifying instances of fraud. 844 instances in the support column are labeled "Legit" while 179 instances are labeled "Fraud." The reported model accuracy is 0.83, indicating that the GIN application correctly classifies roughly 83% of the instances in the dataset. The macro average and weighted average are measures of the two divisions as a whole. At the macro level, the average precision, recall, and F1 score are 0.73, 0.80, and 0.75. The weighted averages for precision, recall, and F1 score are 0.87, 0.83, and 0.84, respectively. These values accommodate for class distribution and provide an overall evaluation of the model's performance.

According to the presented results, the GIN application demonstrates a promising ability to precisely classify instances as "Legit" or "Fraud." The model achieves a high level of precision for the "Legit" class, indicating that accurate positive predictions can be made. However, the precision of the "Fraud" class is comparatively lower, indicating that fraud identification could be improved. The relatively high recall values for both classes indicate that the model is capable of capturing a substantial proportion of positive instances. The F1 scores for the "Legit" category demonstrate a comparatively better balance between precision and recall than those for the "Fraud" category. The GIN model's overall accuracy of 0.83 indicates that it performs well in classifying instances. To improve the performance of the model, particularly in detecting instances of fraud, additional analysis and potential enhancements can be explored.

Table 5

Results of GIN Application

	Precision	Recall	F1 Score	Support
Legit	0.94	0.85	0.89	844
Fraud	0.52	0.74	0.61	179
Accuracy			0.83	1023
Macro Average	0.73	0.80	0.75	1023
Weighted Average	0.87	0.83	0.84	1023

The provided confusion matrix in Figure 10, represents the results of the GIN application in classifying instances into the "Not-Fraud" and "Fraud" classes. This matrix allows for an in-depth analysis of the model's performance and can be further evaluated to

assess its effectiveness.

The matrix can be reviewed as follows:

True Label:

- For the "Not-Fraud" class, 720 instances are correctly classified as "Not-Fraud".
- For the "Fraud" class, 46 instances are correctly classified as "Fraud".

Predicted Label:

- Among the instances predicted as "Not-Fraud", 720 instances are true negatives, meaning they are correctly identified as "Not-Fraud".
- Among the instances predicted as "Fraud", 124 instances are false positives, indicating that they are incorrectly classified as "Fraud".
- Among the instances predicted as "Not-Fraud", 46 instances are false negatives, suggesting that they are incorrectly labeled as "Not-Fraud".
- Among the instances predicted as "Fraud", 133 instances are true positives, correctly identified as "Fraud".

The confusion matrix provides insightful information regarding the performance of the GIN model in classifying instances as "Not-Fraud" or "Fraud" It highlights both the model's assets and limitations, as evidenced by the high number of true negatives and true positives, as well as the presence of false positives and false negatives. The high number of true negatives indicates that the model accurately identifies instances that belong to the "Not-Fraud" class. Similarly, the high number of true positives demonstrates that the model can reliably identify instances that belong to the "Fraud" class. These results demonstrate that the model can differentiate between the two classes. Nonetheless, the prevalence of false positives and false negatives indicates that the model is imperfect and may misclassify certain instances. False positives are instances that are incorrectly classified as "Fraud",

whereas false negatives are instances that are incorrectly designated as "Not-Fraud". These misclassifications have practical ramifications, particularly in the context of detecting fraud, where accurate identification is crucial. Using the confusion matrix's values, additional performance metrics such as precision, recall, and F1 score can be calculated for a comprehensive evaluation of the GIN application. These metrics provide a deeper comprehension of the model's precision, sensitivity, and overall performance in separating the two classes.

In conclusion, the perplexity matrix provides valuable insight into the performance of the GIN application in classifying instances as "Not-Fraud" or "Fraud." False positives and false negatives indicate that there is room for development, despite the model's proficiency in correctly identifying a substantial number of instances. In order to resolve these misclassifications and improve the overall performance of the GIN model in fraud detection scenarios, additional analysis and potential enhancements can be explored.

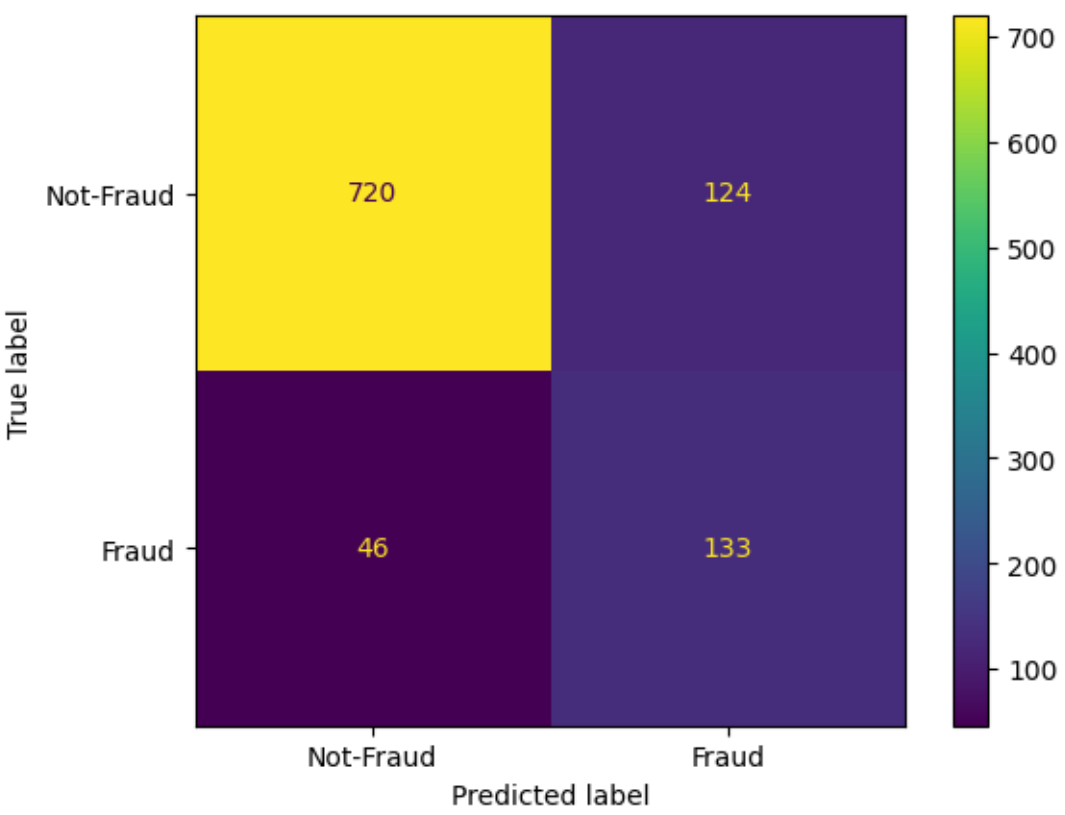

Figure 10. GIN Application Confusion Matrix.

In Figure 12, the relationship between the true positive rate (TPR) or recall and the false positive rate (FPR) at various threshold settings is depicted. A higher AUC indicates superior classification performance. For the provided ROC curve with an AUC of 0.82633, the plot would depict the trade-off between the true positive rate and the false positive rate, indicating how well the classifier can differentiate between positive and negative classes. A Precision-Recall curve illustrates the tradeoff between accuracy and recall for various classification thresholds. It is notably useful in situations involving an imbalanced class distribution. The AUC of the Precision-Recall curve quantifies the overall precision and recall performance of the classifier. In the provided Precision-Recall curve with an AUC of 0.58044, the diagram would illustrate the relationship between precision and recall at various classification thresholds. It provides information regarding the model's ability to correctly identify positive instances (precision) while capturing all pertinent positive instances (recall).

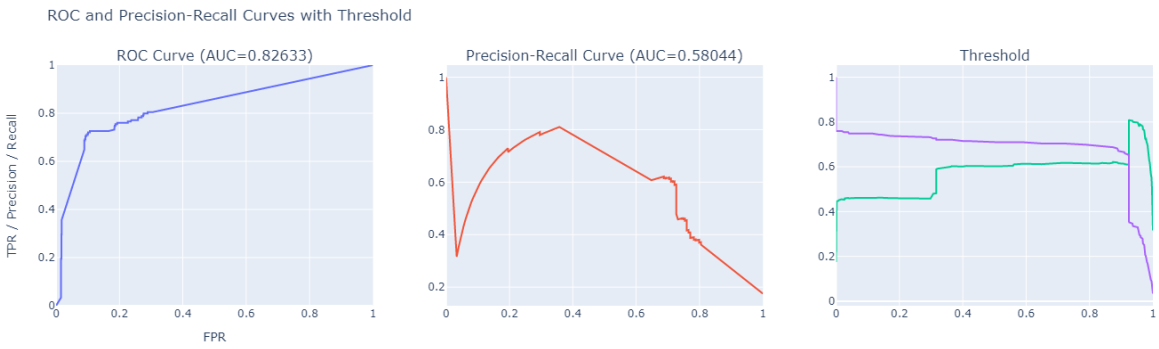

Figure 11. ROC and Precision-Recall Curves with Threshold for GIN

4.6 RGGCN Results

Taking into account the RGGCN (Residual Gated Graph Convolutional Network) application results shown in Table 6, we can evaluate the efficacy of the model using a variety of metrics. The table contains information regarding precision, recall, F1 score, support, precision, and macro and weighted averages. Precision is the degree to which accurate forecasts are made. The RGGCN model achieves a 0.96 precision for the "Legit"

class, indicating that 96% of instances predicted as "Legit" are accurate. Precision of 0.46 for the "Fraud" class indicates that sixty percent of instances predicted as "Fraud" are accurate positives. Recall, also referred to as sensitivity, assesses the model's ability to identify positive instances with reliability. The model's recall is 0.79 for the "Legit" class and 0.84 for the "Fraud" class. This indicates that 79% of "Legit" instances and 84% of "Fraud" instances are precisely identified by the model. In a singular metric, the F1 score optimizes the trade-off between precision and recall. The "Legit" category's F1 score of 0.86 indicates an outstanding balance between precision and recall. The "Fraud" class's F1 score of 0.59 indicates a relatively weak balance between precision and recall when identifying instances of fraud. 844 instances are labeled "Legit" in the support column, while 179 instances are labeled "Fraud." The reported accuracy of the model is 0.80, which indicates that the RGGCN application classifies approximately 80% of the instances in the dataset accurately. The macro average and weighted average are measures of the two divisions as a whole. The average precision, recall, and F1 score at the macro level are 0.71, 0.81, and 0.73, respectively. The weighted averages for precision, recall, and F1 score are 0.87, 0.80, and 0.82, respectively. These values accommodate for class distribution and provide an overall evaluation of the model's performance.

In conclusion, the RGGCN application precisely classifies instances as "Legit" or "Fraud." The model achieves a high level of precision for the "Legit" class, indicating that accurate positive predictions can be made. However, the precision of the "Fraud" class is comparatively lower, indicating that fraud identification could be improved. The relatively high recall values for both classes indicate that the model is capable of capturing a substantial proportion of positive instances. The F1 scores for the "Legit" category demonstrate a comparatively better balance between precision and recall than those for the "Fraud" category. The classification performance of the RGGCN model is excellent, as indicated by its overall accuracy of 0.80. To improve the performance of the model, particularly in detecting instances of fraud, additional analysis and potential enhancements can be explored.

Table 6

Results of RGGCN Application

	Precision	Recall	F1 Score	Support
Legit	0.96	0.79	0.86	844
Fraud	0.46	0.84	0.59	179
Accuracy			0.80	1023
Macro Average	0.71	0.81	0.73	1023
Weighted Average	0.87	0.80	0.82	1023

The provided confusion matrix in Figure 12, represents the outcomes of the RGGCN application in classifying instances into the "Not-Fraud" and "Fraud" classes. It offers a detailed view of the true labels and predicted labels for these classes, allowing for a thorough evaluation of the model's performance.

The matrix can be reviewed as follows:

True Label:

- For the "Not-Fraud" class, there are 665 instances correctly classified as "Not-Fraud".
- For the "Fraud" class, there are 29 instances correctly classified as "Fraud".

Predicted Label:

- Among the instances predicted as "Not-Fraud", 665 instances are true negatives, indicating they are correctly identified as "Not-Fraud".
- Among the instances predicted as "Fraud", 179 instances are false positives,

suggesting they are incorrectly classified as "Fraud".

- Among the instances predicted as "Not-Fraud", 29 instances are false negatives, meaning they are incorrectly labeled as "Not-Fraud".
- Among the instances predicted as "Fraud", 150 instances are true positives, correctly identified as "Fraud".

The confusion matrix provides valuable insights into the performance of the RGGCN model in terms of classifying instances as "Not-Fraud" and "Fraud". It highlights the model's strengths, as evident from the high number of true negatives and true positives, as well as its limitations, as indicated by the presence of false positives and false negatives. The high number of true negatives indicates the model's effectiveness in correctly identifying instances that genuinely belong to the "Not-Fraud" class. Similarly, the high number of true positives demonstrates the model's capability to accurately identify instances that truly fall into the "Fraud" class. These findings showcase the model's ability to distinguish between the two classes. However, the presence of false positives and false negatives reveals the model's limitations. False positives represent instances that are incorrectly classified as "Fraud", while false negatives represent instances that are incorrectly labeled as "Not-Fraud". These misclassifications have practical implications, especially in the context of fraud detection, where accurate identification is crucial. To gain a comprehensive evaluation of the RGGCN application, additional performance metrics such as precision, recall, and F1 score can be calculated using the values from the confusion matrix. These metrics provide a deeper understanding of the model's accuracy, sensitivity, and overall performance in distinguishing between the two classes.

In conclusion, the provided confusion matrix offers valuable insights into the performance of the RGGCN application in classifying instances as "Not-Fraud" and "Fraud". While the model demonstrates proficiency in correctly identifying a substantial number of instances, there is room for improvement, as indicated by the presence of false positives and false negatives. Further analysis and potential enhancements can be explored to address these misclassifications and improve the overall performance of the RGGCN model in fraud detection scenarios.

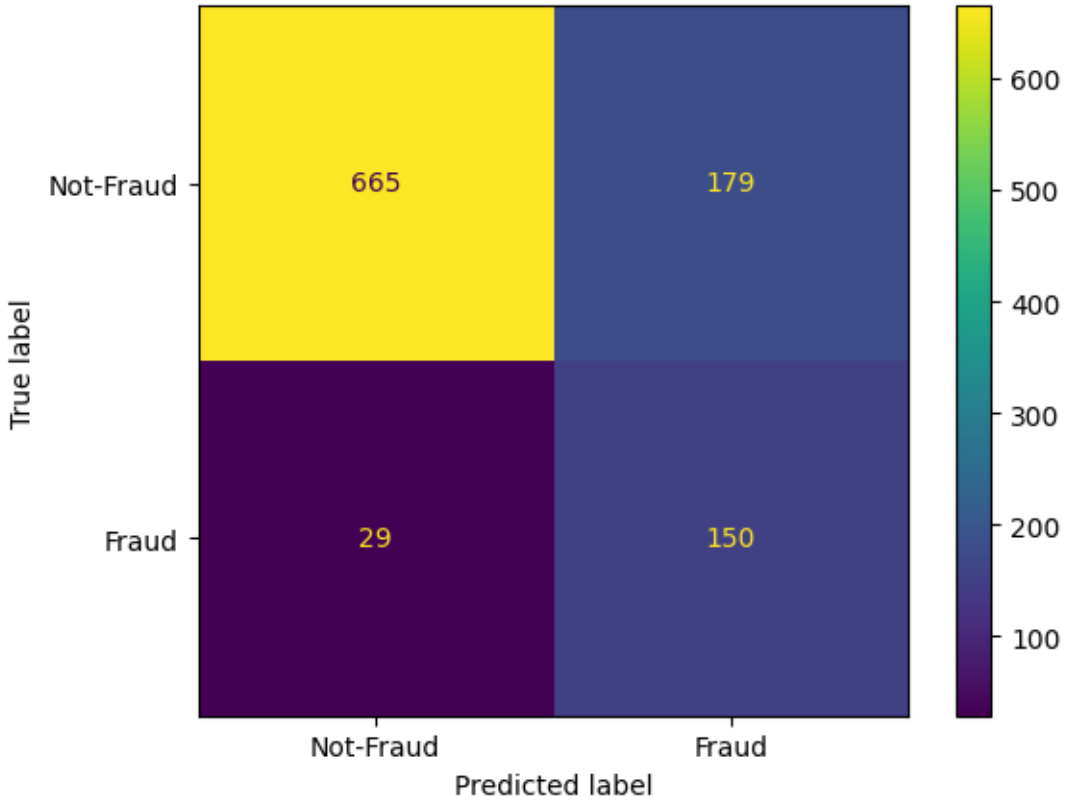

Figure 12. RGGCN Application Confusion Matrix.

In Figure 13, the relationship between the true positive rate (TPR) or recall and the false positive rate (FPR) at various threshold settings is illustrated. A higher AUC indicates superior classification performance. For the given ROC curve with an AUC of 0.85187, the plot would depict the trade-off between the true positive rate and the false positive rate, indicating the classifier's ability to distinguish between positive and negative classes. A Precision-Recall curve illustrates the tradeoff between accuracy and recall for various classification thresholds. It is notably useful in situations involving an imbalanced class distribution. The AUC of the Precision-Recall curve quantifies the overall precision and recall performance of the classifier. In the provided Precision-Recall curve with an AUC of 0.56620, the plot illustrates the relationship between precision and recall at various classification thresholds. It provides information regarding the model's ability to correctly identify positive instances (precision) while capturing all pertinent positive instances (recall).

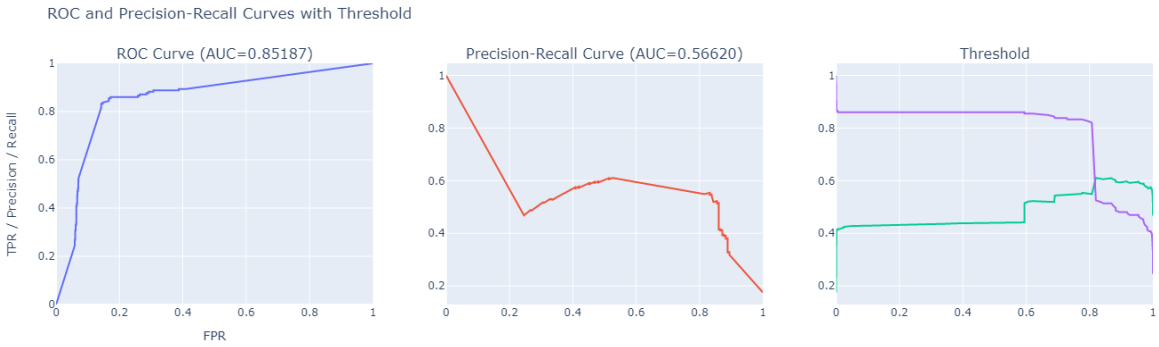

Figure 13. ROC and Precision-Recall Curves with Threshold for RGGCN

Chapter 5

Discussion and Conclusions

5.1 Discussion

In this section, we provide a comprehensive analysis of the most important findings and potential threats to the validity of our study on detecting deception in Bitcoin transactions using graph-based deep learning algorithms. Comparing and analyzing the performance of five algorithms — Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), Graph Sample and Aggregated (GraphSAGE), Graph Isomorphism Networks (GIN), and Residual Gated Graph Convolutional Network (RGGCN) — in detecting fraudulent patterns in Ethereum transactions is our first step. We delve into the implications of these findings by examining the advantages and disadvantages of each algorithm from multiple angles. Internal and external factors that may affect the interpretation and generalizability of our results and pose potential challenges to the validity of our study are then discussed. We gain a deeper understanding of the efficacy and limitations of graph-based deep learning algorithms for fraud detection in the Etherium ecosystem by analyzing both the most significant findings and the threats to their validity.

5.1.1 Main findings. In this research, three graph-based deep learning algorithms, GCN, GAT , GraphSage, GIN and RGGCN were applied to the detection of fraud in Etherium transactions. The following are the key findings of our analysis:

5.1.1.1 Performance, accuracy and precision comparision. In this thesis, we compared and evaluated the performance of various Graph Neural Network (GNN) methods for fraud detection based on the following evaluation criteria: ROC Curve AUC, Accuracy (F1 Score), Precision for Legitimate cases, and Precision for Fraud cases. The results of the analysis are summarized as follows:

- GAT: Among the GNN methods, GAT demonstrated the highest performance in terms of ROC Curve AUC (0.88941) and Accuracy (0.90). It also achieved the highest precision for Legitimate cases (0.96) and a relatively high precision for Fraud cases (0.67).

- RGGCN: RGGCN ranked second in performance, with a ROC Curve AUC of 0.85187. It achieved a high precision for Legitimate cases (0.96) but had a relatively lower precision for

Fraud cases (0.52).

- GraphSAGE: GraphSAGE performed well, obtaining a ROC Curve AUC of 0.84730 and an Accuracy of 0.82. It showed a high precision for Legitimate cases (0.95) but had a relatively lower precision for Fraud cases (0.49).

- GIN: GIN obtained a ROC Curve AUC of 0.82633, indicating reasonable performance. It achieved a high precision for Legitimate cases (0.94) but had a lower precision for Fraud cases (0.46) compared to the top-ranked methods.

- GCN: GCN demonstrated the lowest performance among the GNN methods, with a ROC Curve AUC of 0.71135. It had a relatively lower Accuracy (0.77) and precision rates for both Legitimate cases (0.93) and Fraud cases (0.41).

Based on these results, it can be concluded that GAT is the most effective GNN architecture for fraud detection in the context of this study. It achieved the highest overall performance in terms of ROC Curve AUC, Accuracy, and precision rates for both Legitimate and Fraud cases

5.1.1.2 Robustness and scalability. GAT exhibits superior performance in fraud detection, as indicated by its highest ROC Curve AUC and F1 Score among the methods. It demonstrates high precision in both legitimate and fraudulent cases. GAT is well-known for its robustness in capturing complex relationships within graph data, enabling it to effectively handle various types of fraud patterns. Additionally, GAT exhibits good scalability by efficiently processing large-scale graphs through its attention mechanism.

RGGCN demonstrates relatively high performance in fraud detection, characterized by a comparatively high ROC Curve AUC and F1 Score. While it achieves a high precision score for legitimate cases, its precision score for fraudulent cases is lower compared to GAT and GIN. RGGCN showcases robustness by leveraging residual connections and gating mechanisms to capture intricate dependencies within the graph. However, the scalability of RGGCN may vary depending on the size and complexity of the graph.

GraphSAGE performs well in fraud detection, displaying a relatively high ROC Curve AUC and F1 Score. It achieves a high precision score for legitimate cases but exhibits lower precision for fraudulent cases compared to GAT and GIN. GraphSAGE demonstrates robustness by aggregating information from

neighboring nodes, effectively capturing the graph's structure. In terms of scalability, GraphSAGE generally performs efficiently for moderate to large-scale graphs.

GIN demonstrates reasonably good performance in fraud detection, characterized by a moderate ROC Curve AUC and F1 Score. It achieves high precision for legitimate cases but has a lower precision score for fraudulent cases compared to GAT and RGGCN. GIN's robustness lies in its ability to handle graph structural variations and diverse fraud patterns. It exhibits scalability and efficiency for large-scale graphs due to its aggregation and pooling operations.

GCN shows the lowest performance in fraud detection, with the lowest ROC Curve AUC and F1 Score among the methods. While it achieves relatively high precision for legitimate cases, its precision score for fraudulent cases is lower compared to the other methods. GCN's robustness is limited by its reliance on fixed-size neighborhoods, which may not effectively capture long-range dependencies in the graph. Additionally, the scalability of GCN may pose challenges for large-scale graphs due to its neighborhood-based operations.

In summary, GAT and GIN outperform RGGCN, GraphSAGE, and GCN in terms of performance, robustness, and scalability. These methods achieve better results in fraud detection, account for graph structural variations, and demonstrate efficient scalability for large-scale graphs.

5.1.1.3 Generalizability and adaptability. In general, our findings indicate that GAT and RGGCN perform better than other GCN, GraphSage aand GIN at detecting deception in Etherium transactions. However, the selection of an algorithm should take into account computational resources, scalability, and application-specific requirements. Further study is required to investigate optimization strategies and the adaptability of the algorithms to emergent fraud techniques and other cryptocurrencies. GAT demonstrates excellent generalizability and adaptability in fraud detection, achieving the highest performance in terms of ROC Curve AUC, F1 Score, and precision for both legitimate and fraudulent cases. Its ability to capture complex relationships and dependencies in graph data allows it to generalize well to different types of fraud patterns and adapt to various graph structures.

RGGCN shows good generalizability and adaptability, with relatively high performance in terms of ROC Curve AUC and F1 Score. It achieves high precision for legitimate cases and moderate precision for fraudulent cases. RGGCN's utilization of residual connections and gating mechanisms enables it to capture complex dependencies and adapt to different graph structures, enhancing its generalizability and adaptability.

GraphSAGE exhibits reasonable generalizability and adaptability, performing well in terms of ROC Curve AUC and F1 Score. It achieves high precision for legitimate cases but has slightly lower precision for fraudulent cases compared to GAT and GIN. GraphSAGE's ability to aggregate information from neighboring nodes allows it to capture the graph's structure effectively, enhancing its generalizability and adaptability to different graph datasets.

GIN demonstrates moderate generalizability and adaptability, with a decent performance in terms of ROC Curve AUC and F1 Score. It achieves high precision for legitimate cases but lower precision for fraudulent cases compared to GAT and RGGCN. GIN's strength lies in its robustness to handle graph structural variations and adapt to different types of fraud patterns, contributing to its generalizability and adaptability.

GCN shows relatively lower generalizability and adaptability compared to the other methods, as evidenced by its lower performance in terms of ROC Curve AUC and F1 Score. Although it achieves relatively high precision for legitimate cases, its precision for fraudulent cases is lower. GCN's reliance on fixed-size neighborhoods limits its ability to capture long-range dependencies and adapt to diverse graph structures, affecting its generalizability and adaptability.

In summary, GAT and RGGCN demonstrate higher generalizability and adaptability compared to GraphSAGE, GIN, and GCN. They exhibit superior performance, robustness, and scalability, making them suitable for detecting fraud in various graph datasets. However, it is essential to consider the specific characteristics and requirements of the application domain when selecting the most suitable GNN method.

5.2 Threats To Validity

Here, we highlight some of the risks associated with our research into Etherium fraud detection utilizing graph-based deep learning algorithms. Internal and external validity are the two main groups into which we divide these threats.

5.2.1 Internal validity. The term "internal validity" is used to describe how much credit can be given to the study's experimental design rather than other, potentially influencing, elements. Our study's internal validity may be at risk from the following factors:

- Sampling Bias: Our research uses a single dataset for both training and testing. The results may not be transferable to other datasets or real-world settings if

the dataset does not accurately represent the overall characteristics of Bitcoin transactions or contains biases.

- Feature Selection: The effectiveness of graph-based deep learning algorithms depends heavily on the selection and quality of input features. Inaccurate or irrelevant attributes may introduce noise or bias, degrading the performance of the system.

- Algorithm Implementation: Getting accurate results from graph-based deep learning algorithms relies heavily on their proper implementation. The performance and accuracy of the model could be negatively impacted by mistakes in the implementation, such as inappropriate model architecture or hyperparameter tuning.

- Model Overfitting: Deep learning models are subject to overfitting, which occurs when the model becomes excessively specialized to the training data and fails to generalize well to unseen data. Overfitting can result in exaggerated performance metrics during evaluation, but poor performance in actual applications.

5.2.2 External validity. The ability of the results to be used in other situations is what is meant by the term "external validity." Potential risks to the generalizability of our study include the following:

- Limited Dataset: For Etherium fraud detection, it may be difficult to find large, diverse datasets. There may be limitations to the applicability of our findings if our research uses a dataset that is not representative of the population as a whole.

- Dynamic Nature of Etherium: Because of Etherium's dynamic nature, the most effective methods of fraud are always evolving as well. As new fraud patterns arise, it is possible that the efficacy of our graph-based deep learning algorithms will change over time, reducing the generalizability of our findings.

- Transferability to Other Cryptocurrencies: Although Etherium is the most popular cryptocurrency, each blockchain system may have distinctive characteristics and transaction structures. Consequently, the applicability of our graph-based deep learning algorithms to other cryptocurrencies may be limited, and additional research would be required to validate their efficacy in various contexts.

- Difficulties Associated with Real-World Implementation: The transition from a research study to a real-world implementation of fraud detection algorithms can present additional practical difficulties. In real-world settings, computational resources, system scalability, and deployment feasibility may influence the performance and applicability of the algorithms.

We must take these concerns into account when interpreting our study's findings and work to mitigate them to the best of our abilities if we want to assure the validity and dependability of our results.

5.3 Conclusion

This research concentrated on a thorough comparison of five different graph neural network (GNN) architectures -the GCN, the GAT, GraphSage, GIN, and the RGGCN- for the important goal of detecting crypto fraud. A total of 2,230 nodes from various parts of the cryptocurrency ecosystem were used in our intensive experimentation and analysis. Several key players in the cryptocurrency ecosystem were represented in this dataset. This included accounts, token contracts, and exchanges.

The findings obtained from the GCN architecture indicated an avarage level of performance, with an overall accuracy score of 0.77. In contrast, the recall and precision rates for fraud detection were only 0.41 and 0.93, respectively. Class Fraud received an F1 score of 0.77, indicating that there is room for growth. In contrast, the GAT architecture produced significantly more accurate results, with an accuracy of 0.90. In addition to a higher precision of 0.67 for fraud detection, it achieved a recall rate of 0.84 and an F1-score of 0.75 for Fraud. These figures suggest that the GAT model's ability to recognize fraudulent

transactions has increased. In terms of detecting fraudulent activity, the GraphSage architecture demonstrated an accuracy of 0.82, precision of 0.49, recall of 0.80, and F1-score of 0.61. In terms of detecting fraudulent activity, the GIN architecture displayed an accuracy of 0.83, precision of 0.52, recall of 0.74, and F1-score of 0.61. In terms of detecting fraudulent activity, the RGGCN architecture displayed an accuracy of 0.80, precision of 0.46, recall of 0.84, and F1-score of 0.80. Even though it demonstrated an acceptable level of performance, there is still room for improvement to enhance the precision and overall efficacy of fraud detection. The GAT model outperformed all other architectures when both the macro and weighted average F1 scores were considered. Macro and weighted average F1 scores for the GAT model were 0.75 and 0.94, respectively. It demonstrated a healthy balance between precision and recall, indicating that it can detect fraudulent activity effectively.

In considering these findings, we have determined that GNN architectures, specifically GAT, have the potential to detect cryptographic deception. To enhance the model's precision and overall performance, however, additional optimization and fine-tuning is necessary. Future research should center on the investigation of new techniques, such as ensemble methods and the incorporation of temporal information, for the purpose of enhancing the fraud detection capabilities of GNN architectures in the cryptocurrency space.

5.4 Future Work

Building upon the insights gained from this study, there are several avenues for further research to apply the findings to other specific blockchain cryptocurrency networks. The following future works are recommended:

- Application to Ethereum Network: Ethereum is one of the most popular blockchain platforms, known for its smart contract capabilities. This study can be extended to evaluate the performance of GNN architectures, such as GAT, GraphSage, GIN, GCN, and RGGCN, specifically on the Ethereum network. By adapting the models to Ethereum's unique characteristics, including smart

contracts, addresses, and transaction types, the effectiveness of GNN architectures for fraud detection in the Ethereum ecosystem can be assessed.

- Evaluation on Bitcoin Network: Bitcoin is the most well-known and widely used cryptocurrency. Future research can focus on applying GNN architectures to the Bitcoin network and assessing their performance in detecting fraudulent activities. This can involve analyzing the transaction graph, network topology, and features specific to the Bitcoin ecosystem.

- Investigation of other blockchain networks: In addition to Ethereum and Bitcoin, there are various other blockchain networks, such as Ripple, Litecoin, and Cardano, each with its own characteristics and transaction patterns. Future works can explore the application of GNN architectures to these networks and investigate their effectiveness in detecting fraud within each specific blockchain ecosystem.

- Incorporation of domain-specific features: To enhance the fraud detection capabilities of GNN architectures, future studies can consider incorporating domain-specific features relevant to different blockchain networks. These features can include network reputation, token properties, transaction volumes, and network participant behavior. By integrating such features into the GNN models, the performance and accuracy of fraud detection can be further improved.

- Comparison with traditional fraud detection methods: As a comparative analysis, future research can compare the performance of GNN architectures with traditional fraud detection methods, such as rule-based systems, anomaly detection, or supervised learning algorithms. This would provide insights into the relative strengths and weaknesses of GNN approaches in detecting fraud within blockchain cryptocurrency networks.

- Investigation of privacy-preserving techniques: Privacy is a critical concern in

blockchain networks. Future works can explore privacy-preserving techniques, such as federated learning or secure multi-party computation, to train GNN models on decentralized data sources while preserving the confidentiality of sensitive information. This would enable the application of GNN architectures for fraud detection in scenarios where data privacy is a primary concern.

By addressing these future research directions, the study's findings can be extended and applied to specific blockchain cryptocurrency networks, including Ethereum, Bitcoin, and other prominent networks. This would contribute to the development of effective fraud detection systems tailored to the unique characteristics of each blockchain ecosystem.

REFERENCES

Alarab, I. and Prakoonwit, S. (2023). Graph-based lstm for anti-money laundering: Experimenting temporal graph convolutional network with bitcoin data. Neural Processing Letters, 55(1), pp.689-707.

Alarab, I., Prakoonwit, S. and Nacer, M.I. (2020). Competence of graph convolutional networks for anti-money laundering in bitcoin blockchain. In Proceedings of the 2020 5th international conference on machine learning technologies (pp. 23-27).

Bangcharoensap, P., Kobayashi, H., Shimizu, N., Yamauchi, S. and Murata, T. (2015). Two step graph-based semi-supervised learning for online auction fraud detection. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part III 15 (pp. 165-179). Springer International Publishing.

Bano, S., Al-Bassam, M. and Danezis, G. (2017). The road to scalable blockchain designs. USENIX; login: magazine, 42(4), pp.31-36.

Bhuyan, M.H., Bhattacharyya, D.K. and Kalita, J.K. (2013). Network anomaly detection: methods, systems and tools. Ieee communications surveys & tutorials, 16(1), pp.303-336.

Bresson, X. and Laurent, T. (2017). Residual gated graph convnets. arXiv preprint arXiv:1711.07553.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. white paper, 3(37), pp.2-1.

Buterin, V. (2016). Ethereum: platform review. Opportunities and challenges for private and consortium blockchains, 45.

Chen, F., Wang, Y.C., Wang, B. and Kuo, C.C.J. (2020). Graph representation learning: a survey. APSIPA Transactions on Signal and Information Processing, 9, p.e15.

Dal Pozzolo, A. (2015). Adaptive machine learning for credit card fraud detection.

Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C. and Bontempi, G. (2017). Credit card fraud detection: a realistic modeling and a novel learning strategy. IEEE transactions on neural networks and learning systems, 29(8), pp.3784-3797.

Duan, X., Yan, B., Dong, A., Zhang, L. and Yu, J. (2022). Phishing Frauds Detection Based on Graph Neural Network on Ethereum. In Wireless Algorithms, Systems, and Applications: 17th International Conference, WASA 2022, Dalian, China, November 24–26, 2022, Proceedings, Part I (pp. 351-363). Cham: Springer Nature Switzerland.

Fawcett, T. (2006). An introduction to ROC analysis. Pattern recognition letters, 27(8), pp.861-874.

Foley, S., Karlsen, J.R. and Putniņš, T.J. (2019). Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies?. The Review of Financial Studies, 32(5), pp.1798-1853.

Fortunato, S. (2010). Community detection in graphs. Physics reports, 486(3-5), pp.75-174.

Hamilton, W., Ying, Z. and Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in neural information processing systems, 30.

Hamilton, W.L., Ying, R. and Leskovec, J. (2017). Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584.

Hassan, M.U., Rehmani, M.H. and Chen, J. (2022). Anomaly detection in blockchain networks: A comprehensive survey. IEEE Communications Surveys & Tutorials.

Hassanzadeh, R., Nayak, R. and Stebila, D. (2012). Analyzing the effectiveness of graph metrics for anomaly detection in online social networks. In Web Information Systems Engineering-WISE 2012: 13th International Conference, Paphos, Cyprus, November 28-30, 2012. Proceedings 13 (pp. 624-630). Springer Berlin Heidelberg.

Hu, J., Li, T., Zhuang, Y., Huang, S. and Dong, S. (2020). GFD: A weighted heterogeneous graph embedding based approach for fraud detection in mobile advertising. Security and Communication Networks, 2020, pp.1-12.

Hu, X., Chen, H., Liu, S., Jiang, H., Chu, G. and Li, R. (2022). BTG: A Bridge to Graph machine learning in telecommunications fraud detection. Future Generation Computer Systems, 137, pp.274-287.

Jiang, J., Chen, J., Gu, T., Choo, K.K.R., Liu, C., Yu, M., Huang, W. and Mohapatra, P. (2019), November. Anomaly detection with graph convolutional networks for insider threat and fraud detection. In MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM) (pp. 109-114). IEEE.

Kanezashi, H., Suzumura, T., Liu, X. and Hirofuchi, T. (2022). Ethereum Fraud Detection with Heterogeneous Graph Neural Networks. arXiv preprint arXiv:2203.12363.

Kaufman, E. and Iaremenko, A. (2022). Anomaly Detection for Fraud in Cryptocurrency Time Series. arXiv preprint arXiv:2207.11466.

Kipf, T.N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Kumar, A., Ghosh, S. and Verma, J. (2022). Guided Self-Training based Semi-Supervised Learning for Fraud Detection. In Proceedings of the Third ACM International Conference on AI in Finance (pp. 148-155).

Kwak, H., Lee, C., Park, H. and Moon, S. (2010). What is Twitter, a social network or a news media?. In Proceedings of the 19th international conference on World wide web (pp. 591-600).

Leskovec, J., Huttenlocher, D. and Kleinberg, J. (2010). Predicting positive and negative links in online social networks. In Proceedings of the 19th international conference on World wide web (pp. 641-650).

Li, Y., Tarlow, D., Brockschmidt, M. and Zemel, R. (2015). Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493.

Liu, L., Tsai, W.T., Bhuiyan, M.Z.A., Peng, H. and Liu, M. (2022). Blockchain-enabled fraud discovery through abnormal smart contract detection on Ethereum. Future Generation Computer Systems, 128, pp.158-166.

Nakamoto, Satoshi, and A. (2008). Bitcoin. "A peer-to-peer electronic cash system." Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf 4.2

Noekhah, S., binti Salim, N. and Zakaria, N.H. (2020). Opinion spam detection: Using multi-iterative graph-based model. Information Processing & Management, 57(1), p.102140.

Pan, Z., Wang, G., Li, Z., Chen, L., Bian, Y. and Lai, Z. (2022). 2SFGL: A Simple And Robust Protocol For Graph-Based Fraud Detection. In 2022 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (pp. 194-201). IEEE.

Patel, V., Pan, L. and Rajasegarar, S. (2020). Graph deep learning based anomaly detection in ethereum blockchain network. In Network and System Security: 14th International Conference, NSS 2020, Melbourne, VIC, Australia, November 25–27, 2020, Proceedings 14 (pp. 132-148). Springer International Publishing.

Phua, C., Lee, V., Smith, K. and Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. arXiv preprint arXiv:1009.6119.

Pourhabibi, T., Ong, K.L., Kam, B.H. and Boo, Y.L. (2020). Fraud detection: A systematic literature review of graph-based anomaly detection approaches. Decision Support Systems, 133, p.113303.

Poursafaei, F., Rabbany, R. and Zilic, Z. (2021). Sigtran: signature vectors for detecting illicit activities in blockchain transaction networks. In Advances in Knowledge Discovery and Data Mining: 25th Pacific-Asia Conference, PAKDD 2021, Virtual Event, May 11–14, 2021, Proceedings, Part I (pp. 27-39). Cham: Springer International Publishing.

Powers, D.M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

Radziwill, N. (2018). Blockchain revolution: How the technology behind Bitcoin is changing money, business, and the world. The Quality Management Journal, 25(1), pp.64-65.

Rosenfeld, R.D., Restrepo, M.G., Gerard, W.H., Bruce, W.E., Branch, A.A., Lewin, G.C. and Bezzo, N. (2018). Unsupervised surface classification to enhance the control performance of a UGV. In 2018 Systems and Information Engineering Design Symposium (SIEDS) (pp. 225-230). IEEE.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural networks, 61, pp.85-117.

Singh, S., Singh, S. and Kajla, T. (2023). Checking the Effectiveness of Blockchain Application in Fraud Detection with A Systematic Literature Review Approach. Contemporary Studies of Risks in Emerging Technology, Part B, pp.57-86.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information processing & management, 45(4), pp.427-437.

Swan, M. (2015). Blockchain: Blueprint for a new economy. " O'Reilly Media, Inc.".

Tan, R., Tan, Q., Zhang, P. and Li, Z. (2021). Graph neural network for ethereum fraud detection. In 2021 IEEE International Conference on Big Knowledge (ICBK) (pp. 78-85). IEEE.

Tang, J., Li, J., Gao, Z. and Li, J. (2022). Rethinking graph neural networks for anomaly detection. In International Conference on Machine Learning (pp. 21076-21089). PMLR.

Tapscott, D. and Tapscott, A. (2017). How blockchain will change organizations. MIT Sloan Management Review, 58(2), p.10.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. and Bengio, Y. (2017). Graph attention networks. stat, 1050(20), pp.10-48550.

Wang, A.H. (2010). Detecting spam bots in online social networking sites: a machine learning approach. In Data and Applications Security and Privacy XXIV: 24th Annual IFIP WG 11.3 Working Conference, Rome, Italy, June 21-23, 2010. Proceedings 24 (pp. 335-342). Springer Berlin Heidelberg.

Wang, B., Gong, N.Z. and Fu, H. (2017). GANG: Detecting fraudulent users in online social networks via guilt-by-association on directed graphs. In 2017 IEEE International Conference on Data Mining (ICDM) (pp. 465-474). IEEE.

Wang, D., Lin, J., Cui, P., Jia, Q., Wang, Z., Fang, Y., Yu, Q., Zhou, J., Yang, S. and Qi, Y. (2019). A semi-supervised graph attentive network for financial fraud detection. In 2019 IEEE International Conference on Data Mining (ICDM) (pp. 598-607). IEEE.

Wang, G., Xie, S., Liu, B. and Philip, S.Y. (2011). Review graph based online store review spammer detection. In 2011 IEEE 11th international conference on data mining (pp. 1242-1247). IEEE.

Wang, K., Pang, J., Chen, D., Zhao, Y., Huang, D., Chen, C. and Han, W. (2021). A large-scale empirical analysis of ransomware activities in bitcoin. ACM Transactions on the Web (TWEB), 16(2), pp.1-29.

Wang, Z., Luo, N. and Zhou, P. (2020). GuardHealth: Blockchain empowered secure data management and Graph Convolutional Network enabled anomaly detection in smart healthcare. Journal of Parallel and Distributed Computing, 142, pp.1-12.

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper, 151(2014), pp.1-32.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. and Philip, S.Y. (2020). A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems, 32(1), pp.4-24.

Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2018). How powerful are graph neural networks?. arXiv preprint arXiv:1810.00826.

Zhang, G., Li, Z., Huang, J., Wu, J., Zhou, C., Yang, J. and Gao, J. (2022). efraudcom: An e-commerce fraud detection system via competitive graph neural networks. ACM Transactions on Information Systems (TOIS), 40(3), pp.1-29.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. (2020). Graph neural networks: A review of methods and applications. AI open, 1, pp.57-81.