

TEMPORAL GRAPH FOR FRAUD DETECTION AND ANALYTICS

LEONG TENG MAN

BACHELOR OF APPLIED SCIENCE IN DATA
ANALYTICS WITH HONOURS
UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : LEONG TENG MAN
Date of Birth : 11th November 2000
Title : TEMPORAL GRAPH FOR FRAUD DETECTION
AND ANALYTICS
Academic Session : SEM II 2022/2023

I declare that this thesis is classified as:

- ☐ CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)*
- ☐ RESTRICTED (Contains restricted information as specified by the organization where research was done)*
- ☒ OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)


I acknowledge that Universiti Malaysia Pahang reserves the following right:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

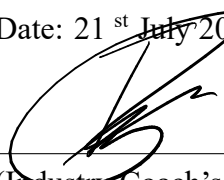
Certified by:


(Student's Signature)
001103140327

New IC/Passport Number
Date: 21st July 2023


(Academic Tutor's Signature)
DR. KU MUHAMMAD
NA'IM KU KHALIF

Name of Academic Tutor
Date: 21st July 2023


(Industry Coach's Signature)
EN. MOHD IZHAR FIR-
DAUS ISMAIL
Name of Industry Coach
Date: 21st July 2023

TEMPORAL GRAPH FOR FRAUD DETECTION AND ANALYTICS

LEONG TENG MAN

Data Science Project Report submitted in fulfilment of the requirements
for the award of the degree of
Bachelor of Applied Science in Data Analytics with Honours

Centre for Mathematical Sciences
UNIVERSITI MALAYSIA PAHANG

JULY 2023

SUPERVISOR'S DECLARATION

We hereby declare that we have checked this Data Science Project Report. In our opinion, this final report of Data Science Project is adequate in terms of scope and quality for the award of Bachelor of Applied Science in Data Analytics with Honours.



(Academic Tutor's Signature)

Full Name : Dr. Ku Muhammad Na'im Ku Khalif

Position : Senior Lecturer

Date : 21st July 2023



(Industry Coach's Signature)

Full Name : En. Mohd Izhar Firdaus Ismail

Position : Chief Solutions Architect

Date : 21st July 2023

STUDENT DECLARATION

I hereby declare that the work in this project report is based on my original work.

A handwritten signature in black ink, appearing to read 'leong', written over a horizontal line.

(Student's Signature)

Name : Leong Teng Man

ID number : SD20019

Date : 21st July 2023

ACKNOWLEDGEMENTS

I would like to express my appreciation to my academic supervisor, Dr. Ku Muhammad who is lenient, patient and understanding my difficulties in learning advanced theories required in completing this project.

I would like to also express my gratitude to my industry supervisor, Mr. Izhar who is lenient and giving me a lot of time and freedom to complete this project while allowing me to continue internship at the company Abyres Enterprise Technologies SDN. BHD. In addition to that, I would like to also thank the Abyres company which have indirectly sponsored me during the process of completing my project in form of internship allowance.

I am grateful to my parents for their understanding and support in maintaining my well-being and morality during the process of project completion.

Lastly, I do like to thank to many who indirectly help and inspire me. I do like to thank to the team from Standard University CS224W Machine Learning with Graphs. Without their developed Pytorch Geometric tools and examples, it would be impossible for me to complete this project. Thanks to many great educators and authors make their books, lecture notes and textbooks available. Their enthusiasm in texts inspire me to be intellectually curious.

Leong Teng Man
UMP Gambang
21 July 2023

ABSTRAK

Transaksi representasi yang berdasarkan masa sebagai graf menawarkan wawasan untuk pengesanan penipuan. Walau bagaimanapun, data yang terhad dan sulit memberi cabaran. Untuk mengatasi ini, projek bertujuan untuk menilai kualiti model generatif dengan menggunakan data yang dihasilkan dalam melatih model pengelas penipuan. Pengekod graf diri (*graph autoencoder*) digunakan untuk melengkapkan data untuk pengesanan penipuan. Eksperimen menggunakan dataset Elliptic, satu dataset transaksi Bitcoin dengan 49 graf komponen berhubung mewakili pengesanan pada selang dua minggu. Sebuah pengekod graf diri variasi khas (*variational graph autoencoder*, VGAE) dibinakan, tetapi menghadapi cabaran dengan fungsi kesalahan (*loss function*) yang ditentukan, menyebabkan penghasilan sisi yang berlebihan. Di antara tiga model rangkaian neural graf (*Graph Neural Networks*, GNN) yang diuji - rangkaian berlingkaran graf (*Graph Convolutional Networks*, GCN), rangkaian tumpuan graf (*Graph Attention Networks*, GAT) dan GATv2, didapati bahawa tiada satu pun dapat mengesan transaksi haram yang tidak dikesan secara efektif. Walaupun GAT dan GATv2 lebih unggul daripada GCN, model yang dilatih dengan data yang direkonstruksi berfungsi lebih buruk daripada model yang hanya menggunakan data asli. Fluktuasi metrik yang konsisten di kalangan model-model tanpa mengira data yang digunakan semasa latihan. Pemerhatian ini menunjukkan VGAE menangkap pola data asli. Kajian ini menyoroti cabaran dan mencadangkan hala tuju penyelidikan masa depan, termasuk autoencoder graf dengan pengaturan lawan, seni bina alternatif, fungsi kesalahan yang cekap, dan kaedah ensemble (*ensemble method*).

ABSTRACT

Representing time-based transactions as graphs offers insights for fraud detection. However, limited and confidential datasets pose challenges. To address this, the project aims to assess the quality of generative model by using the generated data in training the fraud classifier model. A generative approach using graph autoencoders is employed to augment data for fraud detection. Experiments utilize the Elliptic dataset, a Bitcoin transaction dataset with 49 connected component graphs representing confirmations at two-week intervals. A custom variational graph autoencoder (VGAE) is developed, encountering challenges with the defined loss function, resulting in excessive edge reconstruction. Among experimented three graph neural network (GNN) models—Graph Convolutional Networks (GCN), Graph Attention Networks (GAT) and GATv2, it is found that none effectively detect unseen illicit transactions. While GAT and GATv2 outperform GCN, models trained on reconstructed data perform worse than those on purely original data. It is observed the consistent metric fluctuations among models regardless of data used during training. This indicates the VGAE captures underlying patterns. This study highlights challenges and suggests future research directions, including adversarially regularized graph autoencoders, alternative architectures, efficient loss functions, and ensemble methods.

TABLE OF CONTENTS

DECLARATION	
TITLE PAGE	
ACKNOWLEDGEMENTS	i
ABSTRAK	ii
ABSTRACT	iii
TABLE OF CONTENTS	iv
LIST OF TABLES	vi
LIST OF FIGURES	vii
LIST OF SYMBOLS	viii
LIST OF ABBREVIATIONS	ix
CHAPTER 1 INTRODUCTION	1
1.1 Research Background	1
1.2 Problem Statement	2
1.3 Research Question	3
1.4 Research Objective	3
1.5 Research Scope	3
1.6 Significance of Study	4
CHAPTER 2 LITERATURE REVIEW	5
2.1 Introduction	5
2.2 Deep Learning	5
2.3 Data Analytics	6
2.4 Fraud	7
2.5 Temporal Graph	7
2.6 Fraud Detection and Graph Anomaly Detection (GAD)	7
2.6.1 Related Survey	8

2.6.2	Graph anomaly detection (GAD)	8
2.7	Deep Generative Model for Graph	10
2.8	Chapter Summary	10
CHAPTER 3 METHODOLOGY		11
3.1	Introduction	11
3.2	Dataset	11
3.3	Modelling	12
3.3.1	Representing the data as Temporal Graphs	12
3.3.2	Graph Neural Network (GNN)	12
3.3.3	Variational Graph Autoencoder (VGAE)	15
3.4	Model Evaluation	16
3.5	Experiment Setup	17
3.5.1	Fraud Classifier GNN Model	18
3.5.2	Graph Reconstruction – VGAE	18
3.5.3	Training	19
CHAPTER 4 DATA ANALYSIS, RESULTS AND DISCUSSION		21
4.1	Ellitpc Dataset Summary	21
4.2	VGAE	24
4.3	Model that use Original Dataset	25
4.4	Visualizing the Prediction	27
4.5	Model that use Reconstructed Data	27
4.6	Summary	33
CHAPTER 5 CONCLUSION AND RECOMMENDATIONS		34
5.1	Conclusion	34
5.2	Recommendation	34
REFERENCES		36

LIST OF TABLES

Table 2.1: List of Literature

9

LIST OF FIGURES

Figure 3.1:	Data Science Methodology	12
Figure 3.2:	Research Plan	13
Figure 3.3:	transaction graph starting from 232013274 transaction id node	14
Figure 3.4:	transaction graph starting from 300318525 transaction id node	14
Figure 3.5:	transaction graph starting from 232629023 transaction id node	14
Figure 3.6:	source: (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021). A single layer of a simple GNN	14
Figure 3.7:	source: (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021). A single layer of message-passing style GNN	15
Figure 3.8:	Custom VGAE model	19
Figure 4.1:	Count of transactions by classes	22
Figure 4.2:	Count of transaction nodes by class across the time steps	22
Figure 4.3:	Transaction Graph at time steps = 2	23
Figure 4.4:	Training History of VGAE	24
Figure 4.5:	Precision Recall Curve Plot	25
Figure 4.6:	Size ratio between reconstructed and original edge numbers	26
Figure 4.7:	Model Performance Comparison (Train Using Original Data)	26
Figure 4.8:	Model Comparison	28
Figure 4.9:	AUCROC: Model Comparison (Train Using Original Data)	29
Figure 4.10:	Transaction Graph labelled by GAT model at time steps = 12	29
Figure 4.11:	GAT trained on different dataset	30
Figure 4.12:	GATv2 trained on different dataset	31
Figure 4.13:	GCN trained on different dataset	32

LIST OF SYMBOLS

$G = \{V, E\}$	Graph is a set of vertices and edges
U	Universal or Global. It represent attributes related to a graph.
i, j, k	integer subscripts; index subscripts
n	natural number subscript

LIST OF ABBREVIATIONS

ML	Machine Learning
MLP	Multilayer Perceptron
DL	Deep Learning
DS	Data Science
DA	Data Analytics
GAD	Graph Anomaly Detection
GAN	Generative Adversarial Network
CNN	Convolutional Neural Network
GNN	Graph Neural Network
GCN	Graph Convolutional Neural Network
GAE	Graph Autoencoder
VGAE	Variational Graph Autoencoder
GAT	Graph Attention Network
LSTM	Long short-term memory
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
AUC	Area under the Curve
MSE	Mean Squared Error

CHAPTER 1

INTRODUCTION

1.1 Research Background

In mathematics, a graph is a set containing a set of nodes or vertices and a set of edges representing connection between nodes. There are many types of graph.

The exact representation of nodes and edges may vary. For example, depending the restriction on traversing in a graph, a graph can be either directed or undirected. An edge in directed graph is a tuple whose first component represent source node and second component represent destination node. whereas an edge in undirected graph may be well-defined by either a tuple or a set.

Graphs can have weights. For example, the weight of edge represents the amount of transaction between two users which are represented as two nodes.

Graph naturally arose as a model in many fields that involving relationships between objects. The applications include network system, social network and fraud detection. In addition, temporal graph is a graph that changes with time. Temporal graph is also known as dynamic graph. Temporal graph may represent truer realities because some networks do change overtime. For example, in graph anomaly detection and fraud detection, studies suggest that temporal attribute is significant in detecting credit card fraud. (Cheng et al., 2020)

Among the references, fraud is often regarded as same term as anomaly in graph detection. Informally, fraud is defined as deceiving others in gaining advantages but resulting possible losses in deceived objects. Beside financial frauds, it may include internet frauds such as abusing advertising system and auction system. Indeed, there are as many frauds and fraudsters change their strategies as the anti-fraud systems deploy.

Existing methods for fraud detection in graphs encompass a range of techniques that leverage the structural properties and dynamics of the graph data. One commonly employed approach is anomaly detection, which aims to identify nodes or edges that deviate significantly from the expected patterns in the graph. Anomaly detection algorithms, such as outlier detection and clustering-based methods, can be applied to uncover unusual behaviors or transactions that might indicate fraudulent activity.

In addition to graph-based algorithms, machine learning techniques that are not explicitly graph-based have also been employed for fraud detection. These methods focus

on analyzing the attributes and features associated with nodes or edges in the graph to detect fraudulent patterns.

Graph-based deep learning has caught attentions in current research communities. Graph-based deep learning, particularly with graph neural networks (GNNs), has achieved mix-result in many applications including fraud detection. GNNs leverage the graph structure to capture complex dependencies between nodes and edges, enabling the detection of fraudulent patterns

Despite having motivations in using temporal graph, its computational methods are not as simple as in static graph. Indeed, a recent review Ma et al. (2021) suggests that it is a future direction in incorporating temporal graph with machine learning.

As fraud issues hurt a company's reputation and it involves confidential data, public fraud-related datasets are extremely rare even in anonymized form. Therefore, some researchers have to find a way to generate synthetic data.

1.2 Problem Statement

Since the advent of the internet, online fraud has had a negative impact on societies, particularly in the context of the ever-increasing volume of financial transactions. The task of detecting fraud becomes overwhelming as the data continues to grow exponentially. To address this challenge, graph neural networks (GNNs) have been applied to identify fraudulent activities. GNNs are neural networks specifically designed to process graph-structured data, making them well-suited for analyzing transactional graphs.

However, one major obstacle in developing effective fraud detection systems is the limited availability of public, published fraud datasets. This lack of accessible data hampers the rapid update and improvement of anti-fraud systems. In response, researchers have turned to generative models, such as autoencoders, to generate synthetic datasets that can be used to augment the limited existing data. By using generative models, researchers can create realistic fraud scenarios and increase the diversity of the training data, facilitating the development of more robust fraud detection algorithms.

Another related challenge arises from the unintended consequences of stringent anti-money laundering (AML) regulations, which can lead to restricted access to financial services, including difficulties in opening bank accounts. While these regulations are essential for safeguarding the financial system, they often disproportionately affect vulnerable populations. Therefore, there is a need to develop innovative fraud detection systems that can operate within more lenient regulatory frameworks, striking a balance between fraud prevention and facilitating increased access to financial services for a wider range of individuals.

The use of generative models, such as autoencoders, to augment data might help

overcome the scarcity of labeled fraud instances. However, the use of graph autoencoders are rarely discussed in the fraud detection. Therefore, this is the gap this project want to fill.

1.3 Research Question

1. Does the custom variational graph autoencoders (VGAE) simulate the fraudulent behaviour? Is generated dataset resemblance to real dataset?
2. Does graph neural network works in detecting fraud?
3. How employed graph-based anomaly detection algorithms perform in detecting fraud?
4. How to assess the quality of generated data?
5. How generated data affects the models in detecting fraud?

1.4 Research Objective

1. To train the generative model that generate data
2. To apply graph-based anomaly detection algorithm to detect fraud

1.5 Research Scope

- The dataset from real world scenario is either limited or confidential. Thus, the dataset must be masked before publishing for public use.
- The considered scenario is transaction related. (e.g.: money transaction between bank accounts) The experiment focuses on fraudulent transactions.
- The main algorithms are graph neural network (GNN).
- Theoretical perspective is not emphasised. This paper neither mathematically prove convergence of algorithm nor doing algorithm analysis.

1.6 Significance of Study

- The study contributes to the research community's understanding of graph autoencoders. By investigating novel approaches such as integrating node feature reconstruction in GAEs and analyzing the use of reconstructed data in other graph-based anomaly detection algorithms, the research expands the knowledge base and opens avenues for further exploration. It stimulates ongoing research and facilitates the development of more sophisticated and effective fraud detection methodologies.
- Fraudulent activities can cause financial losses and harm a platform's reputation. Prioritizing fraud detection and prevention is crucial for platforms to maintain trust and attract users. It is also important to develop detection systems that work under lenient regulations, allowing more individuals to access financial services securely.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this literature review section, we shall review definition and recent applications in fraud detection. For the graph anomaly detection (GAD) algorithms and generative model for graph, readers may consult the summaries from the papers (Guo & Zhao, 2020; Ma et al., 2021; Pourhabibi, Ong, Kam, & Boo, 2020).

There is not many papers discuss the combination of temporal, generative and graph in fraud detection. Hence, this project aims to investigate the effectiveness of graph autoencoder (GAE) and its method in detecting frauds.

2.2 Deep Learning

Artificial intelligence is a field includes machine learning where machine learning includes deep learning. For historical context, readers may consult the textbook by Goodfellow Goodfellow, Bengio, and Courville (2016). Deep learning model is to learn representation that is best for its task. Developing efficient machine learning requires good feature engineering whereas deep learning model learn to do that. This lead to end-to-end model for many task where input and output are fed into the model without preprocessing.

To better navigate the deep learning's literatures, author proposed a roadmap of deep learning. Any deep learning model can be mixed of these or extension of simpler model. For example, CNN could be considered as extension to MLP, and convolutional layers could be added to GAN.

- Multi-Layer Perceptron (MLP)
- Backward propagation
- Loss Functions
- Optimization Algorithms
 - SGD, AdaGrad, RMSProp, Adam

- Convolutional Neural Network (CNN)
- Embedding Layer
- Adversarial Techniques
- Generative Adversarial Network (GAN)
- Parameter initialization
- Batch Normalization
- Regularizing Techniques
- Autoencoder (AE), Variational autoencoder (VAE)
- Recurrent Neural Network (RNN)
 - Valina RNN, LSTM, GRU
- Transformers

Message-passing style is the popular implementation in GNN research community. GraphSage, GCN and GAT are fall under this class.

2.3 Data Analytics

To author's readings, there is no unanimous agreement on the exact definition of data analytics. Frankenfield (2022), Stedman (n.d.) and ACCA (n.d.) share the same idea that data analytics involve "discover meaningful pattern from raw data" .

In the paper by Razzak, Imran, and Xu (2019), it implicitly states that data analytics perform more sophisticated analysis on the data. The survey (Tsai, Lai, Chao, & Vasilakos, 2015) refer to the data analytics as the whole knowledge discovery in database (KDD) process. Data analysis is a part of data analytics whose objective is to discover the hidden information in the data.

Early use of the "data analytics" term can found in the article (Tyagi, 2003). The data analytics is often discussed in a certain context or application. Thus, the further interpretations is the defining properties of data analytics. For example, a cycle in a graph itself does not have meaning but the cycle in a transaction graph could be suspected to be a fraudulent activity (Hajdu & Krész, 2020).

2.4 Fraud

In the broadest sense, fraud is any activity that use misdirections to gain advantages. Indeed, it is impossible to summarise the types of fraud in few pages (Beals, DeLiema, & Deevy, 2015). Nevertheless, the paper by Beals et al. (2015) focuses on financial fraud. This is another paper by Taylor, Goeringer, Winter, and Khalilian (2020) focuses on classifying internet fraud.

A vulnerability is defined as a glitch of a system while an exploit is an attempt that leverage vulnerability to acquire advantages. There is one possible similarity between exploitations (or abusing) and frauds. Cressey's fraud triangle proposes that a person carries out fraud when he discovers opportunity, he persuades himself so he can relieve his monetary pressure; opportunity, pressure and rationalization together that cause a person whether commits fraud (Akkeren, 2018). Therefore, it can be the system's weakness that provide the opportunities for fraudsters carrying out fraud. In other word, detecting the flaws of system can be a part of fraud prevention. To avoid possible confusions, this project implicitly extrapolate the term "fraud" when necessary.

2.5 Temporal Graph

Informally, temporal graph is a graph that changes its contents as time elapsed. In some literatures, temporal graphs (Michail, 2016) can be also known as *Dynamic, evolving* (Ferreira, 2004), and *time-varying* (Flocchini, Mans, & Santoro, 2013) graphs.

There is a paper provides introductory materials to temporal graph but from an algorithmic perspective (more theoretical) such that it discusses the notion of *connectivity, path* and *information dissemination*. In this project, temporal graph is defined as a sequence of graphs for mathematical convenience (Michail, 2016). It is not necessary that the data should contains time but rather it is the change that form a sequence of event. Therefore, LSTM and RNN are revelant model to exploit such tempoeral feature.

2.6 Fraud Detection and Graph Anomaly Detection (GAD)

In Malaysia, there is a paper concludes that interviewed auditors agree that use of data analytics provides better fraud detection (SASTRY, LEE, & TEOH, 2021).

Fraud is known as an application area under anomaly detection. Furthermore, some frauds are naturally modelled as a graph. There are at least 3 type of anomalies in graph in fraud related scenario. They are abnormal relation (edge), malicious users (node), malicious user group (subgraph) and graph.

2.6.1 Related Survey

Ma et al. (2021) have done a comprehensive survey on summarising various papers GAD with deep learning. The paper's proposed techniques are based on either node, edge, sub-graph or graph. The paper curates a list of published dataset, algorithm and models. The paper identifies that deep learning based GAD relies on DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014) and GCN which are designed for static graph. This implicitly implies some algorithms does not fully utilized temporal information in detecting anomalies.

Pourhabibi et al. (2020) have done a systematic review about fraud detection using GAD approaches. The review also provide a taxonomy (i.e.: classification framework) for classifying various GAD algorithm based on methods, application area, properties of the input graph (4 dichotomies), data tag availability, and types of anomalies. The review too identifies the lack of research on fraud detection using temporal graph. Indeed, it has been demonstrated that temporal information is a feature that can distinguish between legit and illegal transactions in the paper (Cheng et al., 2020).

2.6.2 Graph anomaly detection (GAD)

Y.-J. Zheng, Zhou, Sheng, Xue, and Chen (2018) proposed generative-adversarial network - dual autoencoders (GAN-DAE) that uses GAN and gaussian mixture model to detect telcom fraudulent transaction at the receiving bank. However, the input features used cannot be naturally represented in terms of graph.

LSTM is a type of network capable of learning a sequence of data. One recent application is LSTM-GNN combination exploiting time series data and neighbourhood information to predict patient outcomes (Rocheteau, Tong, Velickovic, Lane, & Liò, 2021; Rocheteau, Tong, Veličković, Lane, & Liò, 2021). P. Zheng, Yuan, Wu, Li, and Lu (2018) proposed the one-class adversarial nets model (OCAN) that involves LSTM and GAN but it does not exploit the relationship between nodes. Indeed, the paper did not model the problem in terms of graph. In application of OCAN to detect vandal on wikipedia, the F_1 score is 0.9. Indeed, the paper and this project shares the same ideas exploiting tempoeral feature and using GAN to remedy lack of data (P. Zheng, Yuan, Wu, Li, & Lu, 2018).

The paper (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021) provides excellent introduction to GNN. It should be noted that not all reviewed GAD algorithms by these paper (Ma et al., 2021; Pourhabibi, Ong, Kam, & Boo, 2020) is GNN based. Lastly, this section organized a Table 2.1 containing references, application area and comment to the recent literature that apply the graph as model in detecting fraud.

Table 2.1: List of Literature

algorithm/ framework	references	application area/ dataset	comment	gap
BIRDNEST	(Hooi et al., 2015)	Suspicious On-line Rating	bayesian	-
GraphGAN	(Wang et al., n.d.)	variety real dataset		temporal exploitation is unknown
GAN-DAE	(Y.-J. Zheng, Zhou, Sheng, Xue, & Chen, 2018)	transactional data	camouflaged, non-graph based	temporal exploitation is unknown
GAN	(Sethia, Patel, & Raut, 2018)	credit card fraud		temporal exploitation is unknown
OCAN (LSTM + GAN)	(P. Zheng, Yuan, Wu, Li, & Lu, 2018)	vandals on wikipedia		Exploit temporal feature and generative-adversarial but relationship between users is not exploit
BRNADS	(Manjunatha & Mohanasundaram, 2018)	Social Network	framework involving data streaming and warehouse	framework does not integrate multiple GAD algorithms
SEDANSPOT	(Eswaran & Faloutsos, 2018)	variety real and synthetic		using RHSS as only baseline algorithm
EvolveGCN	(Pareja et al., 2019)	variety real and synthetic	not include fraud dataset	-
F-FADE	(Chang et al., 2020)	variety real and synthetic		-
Node2vec	(Zhou et al., 2021)	internet finance dataset	framework involving data streaming and warehouse	-

2.7 Deep Generative Model for Graph

There is a systematic review regarding of generative model for graph discussing taxonomy, comparison and evaluation metric. The paper includes references to graph sequential generating model. However, there is no fraud application related paper found in the paper's review literature. (Guo & Zhao, 2020)

At high level, there are unconditional and conditional. Conditional generative model is to learn a distribution $p_{\text{model}}(G|y)$ given that input graphs G and information y . In this project, the information could be isFraud label. Unconditional generative model is to learn a distribution $p_{\text{model}}(G)$ using input graphs.

Under the unconditional model, there are one-shot and sequential generation. Former model results a full graph whereas latter construct the graph sequentially whose result is essentially same as temporal graph.

The graph autoencoder are considered generative model that do one-shot generation. However, the graph autoencoder developed by Kipf and Welling (2016) focuses on link reconstruction instead of reconstructing node features together.

2.8 Chapter Summary

To sum up the literature review, there are variety GAD algorithms incorporated with machine learning (ML) are deploying in detecting anomalies including frauds, but few of them fully utilized the temporal feature. Indeed, among the paper reviewed, not many discuss the effect of using generated data using autoencoder in training the classifier model. Furthermore, the graph autoencoder focuses on link prediction but does not reconstruct node the features. This is a gap that this project wishes to fill.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter explains the rationale the decision made in the fraud detection task and research plan.

Figure 3.1 depicts data science methodology. The essence of the methodology is iterative; steering previous steps based on feedback of current step. For example, the choice of preprocessing depends much which model will be used. However, due to obvious confidential issues of financial dataset, the features in public transaction datasets are masked limiting the choice of data analysis. Therefore, the project focuses on developing the models that detect fraud and generate data.

Figure 3.2 depicts the how project progresses towards result.

3.2 Dataset

Financial datasets are typically confidential and not publicly available. To protect privacy, these datasets must be masked or anonymized before being published for public use. There are variety transactions datasets or simulator (Le Borgne, Siblini, Lebichot, & Bontempi, 2022; Lopez-Rojas, Elmir, & Axelsson, 2016; Suzumura & Kanezashi, 2021) but the transactions is not temporal and not easily represented as graphs.

By far, the Elliptic dataset (Weber et al., 2019) is the largest labelled and masked public transaction dataset. The dataset represents Bitcoin transaction graphs. The project uses this dataset for experimentation. The details about datasets such as anonymisation shall refer to the dataset's paper (Weber et al., 2019)

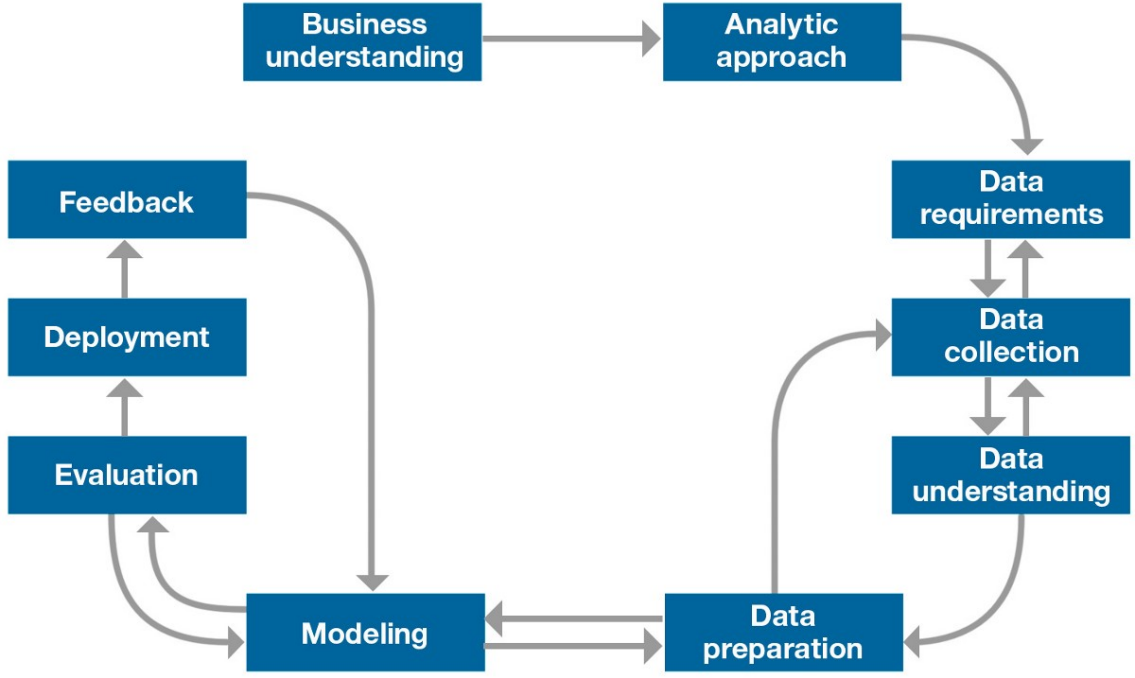


Figure 3.1: Data Science Methodology

3.3 Modelling

3.3.1 Representing the data as Temporal Graphs

The Figure 3.3, 3.4 and 3.5 depicts how the data in Elliptic datasets can be modelled as a graphs. If the nodes are labelled with the transaction id, then it is an illicit transaction. When a user wants to send bitcoins to another user, they create a transaction. A transaction includes the recipient's Bitcoin address, the amount of bitcoins being sent, and other required information. The transaction is digitally signed with the sender's private key to ensure authenticity. Throughout this process, the transactions are recorded on the blockchain, creating an immutable and transparent history of the ownership and movement of the bitcoins. This is why nodes representation a transaction block connecting to the next transactions where the users use "previous transactions" to trade.

3.3.2 Graph Neural Network (GNN)

Message Passing Graph Neural Networks (GNNs) are a class of neural network architectures that is capable for the tasks such as node classification, link prediction, and graph-level prediction. The most popular approach in GNNs is message passing, where information is exchanged between nodes in a graph.

The Figure 3.6 depicts a single layer of a simple GNN at n^{th} layer. The in-

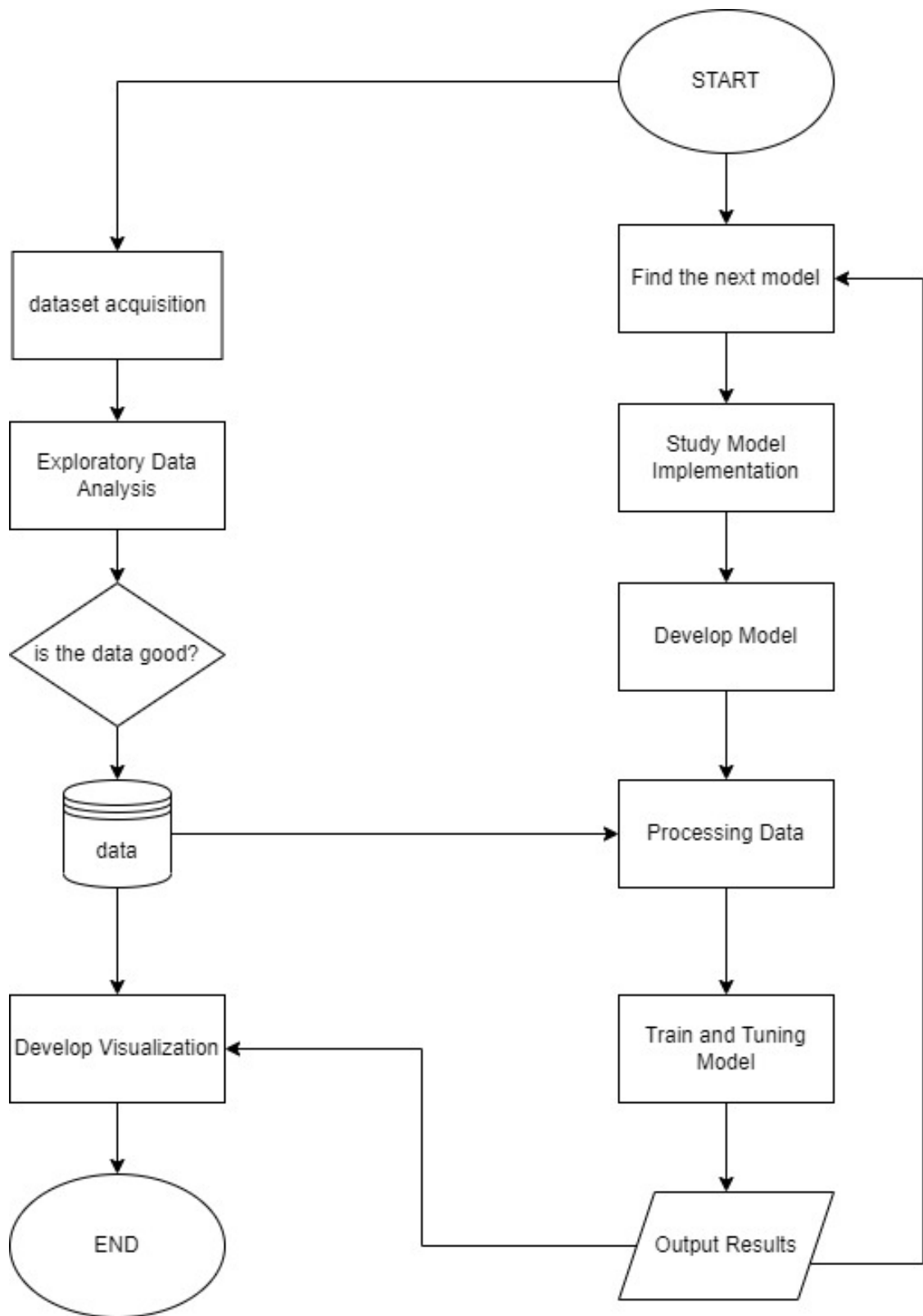


Figure 3.2: Research Plan

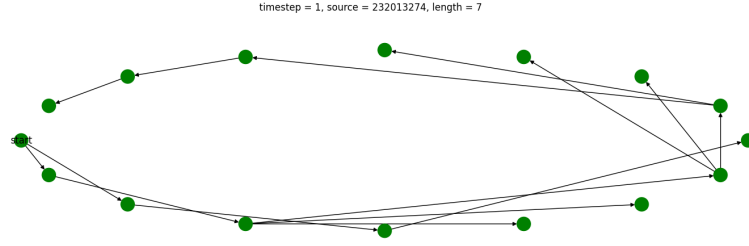


Figure 3.3: transaction graph starting from 232013274 transaction id node

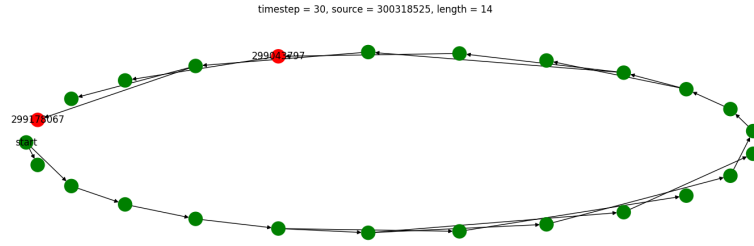


Figure 3.4: transaction graph starting from 300318525 transaction id node

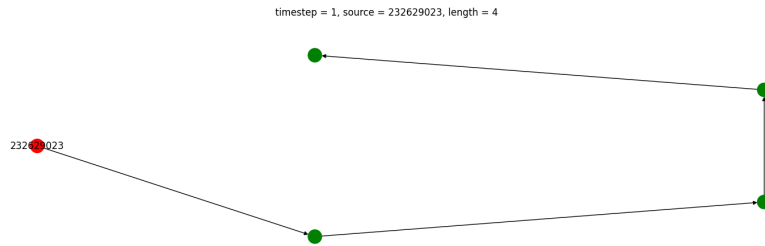


Figure 3.5: transaction graph starting from 232629023 transaction id node

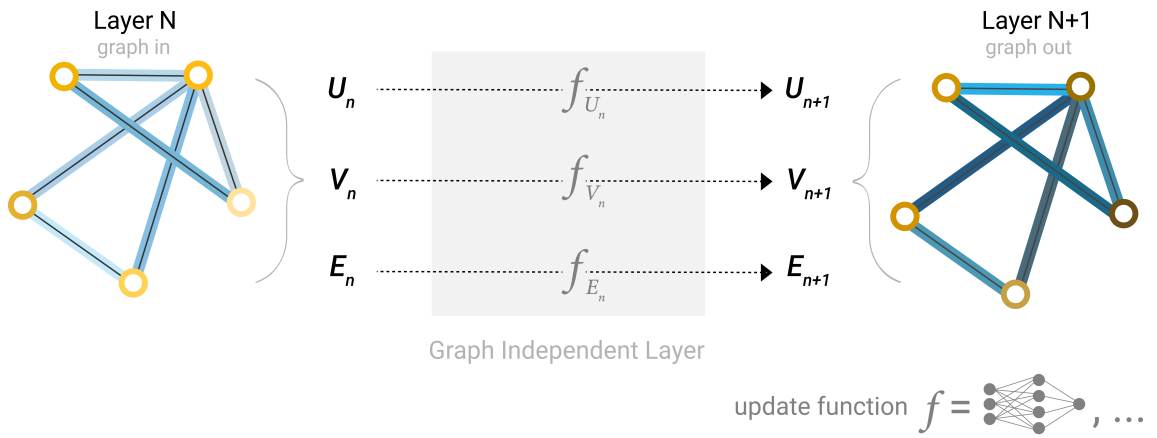


Figure 3.6: source: (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021). A single layer of a simple GNN

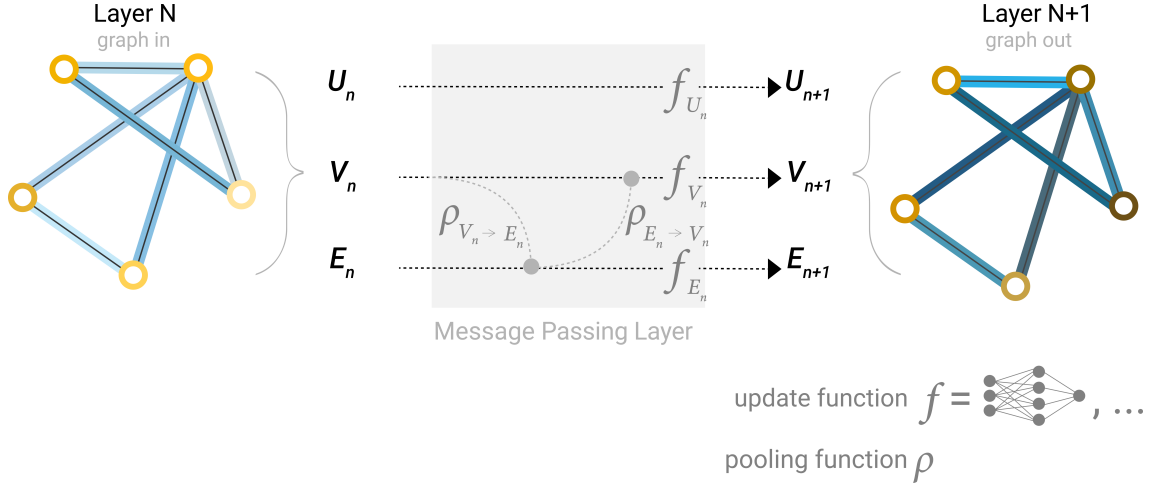


Figure 3.7: source: (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021). A single layer of message-passing style GNN

put is a graph, and each part (V, E, U) is passed to a update function to produce a new graph $G' = (V', E', U')$. The subscript of update function f distinguish a different MLP (Sanchez-Lengeling, Reif, Pearce, & Wiltchko, 2021). This layer does not use connectivity information.

In a single layer of message-passing style GNN as depicted in the Figure3.7, each node initializes with a feature vector that represents its attributes or state. Next, nodes exchange messages with their neighbors, aggregating and transforming the feature vectors of their neighboring nodes. This process enables nodes to gather information from their neighboring nodes. The aggregated messages are then passed to the neighboring nodes. Optionally, an activation function can be applied to the messages before they are forwarded to subsequent nodes in the graph.

Various aggregation techniques, such as summation, averaging, or attention mechanisms, can be used to aggregate the messages. Based on different aggregation techniques, there are 3 GNN models used in this project: Convolutional (Kipf & Welling, 2017), GAT (Veličković et al., 2018) and GATv2 (Brody, Alon, & Yahav, 2022).

Historically, convolutional originates in images classification whereas transformers are proposed to learn from a long sequence of texts. Detailed discussion of convolutional and transformers are beyond the scope of this project. Interested readers should refer to the related literatures.

3.3.3 Variational Graph Autoencoder (VGAE)

A graph autoencoder (GAE) is a neural network architecture that learns compact representations of graph-structured data. It encodes a graph into a lower-dimensional latent space and reconstructs the original graph from this representation. It does not use

label thus it is suitable for unsupervised task. The encoder maps the graph's nodes and edges to a latent space, while the decoder reconstructs the graph. The model minimizes the reconstruction error during training.

A variational graph autoencoder (Kipf & Welling, 2016) is a neural network architecture that combines GAE with variational inference techniques. VGAEs learn probabilistic latent representations of graphs, allowing for uncertainty modeling. VGAE can indirectly generate new data. The encoder part of VGAE outputs two vectors of mean and log-variance. The latent spaces can be sampled from these means and log-variance before passing to the decoder.

In short, the reconstructed data generated by VGAE exhibits slight differences compared to the original data, allowing for the generation of new data points.

3.4 Model Evaluation

Confusion matrix provides a comprehensive summary of the classification results by comparing predicted labels with the ground truth. The confusion matrix is typically organized into four quadrants: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). From the confusion matrix, several performance metrics can be derived to assess the effectiveness of the fraud detection system:

1. Accuracy: Accuracy measures the overall correctness of the system's predictions and is calculated as $(TP + TN) / (TP + TN + FP + FN)$. It represents the proportion of correctly classified instances out of the total number of instances. However, accuracy alone may not be sufficient when dealing with imbalanced datasets where the majority of instances are non-fraudulent.
2. Precision: Precision quantifies the system's ability to correctly identify fraud cases among the predicted positives. It is calculated as $TP / (TP + FP)$ and represents the ratio of true positives to the total number of instances predicted as positives. Precision indicates the proportion of correctly identified fraud cases out of all predicted fraud cases.
3. Recall (also known as sensitivity or true positive rate): Recall measures the system's ability to correctly detect fraud cases among all the actual positives. It is calculated as $TP / (TP + FN)$ and represents the ratio of true positives to the total number of actual positives. Recall indicates the proportion of actual fraud cases that are correctly identified by the system.
4. F1 Score: The F1 score is the harmonic mean of precision and recall, providing a balanced measure of the system's performance. It is calculated as $2 * (Precision * Recall) / (Precision + Recall)$.

Recall) / (Precision + Recall). The F1 score combines both precision and recall into a single metric, offering a comprehensive evaluation of the system’s effectiveness in detecting fraud cases.

These performance metrics derived from the confusion matrix enable a deeper understanding of the fraud detection system’s strengths and weaknesses. Accuracy provides an overall measure of correctness, while precision, recall, and the F1 score offer insights into the system’s ability to correctly identify fraud cases and minimize false positives and false negatives. By analyzing these metrics, developers and researchers can fine-tune the system and optimize its performance to enhance fraud detection accuracy.

In short, the higher these value, the better the model perform.

Beside evaluation based on confusion matrix, the VGAE can be tested such that how classifier model performs if training on original data, reconstructed data and mix of both. In ideal scenario, the models that use only reconstructed data during training should be perform equally to those use original data. This is also the reason why the experiment is setup to train models using different composition of datasets.

3.5 Experiment Setup

The main library used is PyTorch and PyTorch Geometric. PyTorch Geometric is an extension of PyTorch that focuses on graph-related tasks. It provides a variety of GNN layers and utilities for graph manipulation.

However, PyTorch Geometric’s GNN layers operate on adjacency matrices which are in coordinate format and shall be referred as sparse representation whereas the full adjacency matrices as dense representation. PyTorch Geometric provides utility functions to convert between these two representations. This is because sparse representation is more memory efficient than dense representation but otherwise, they are same.

The dataset is split into 49 graph data, each corresponding to the 49 timesteps. The first 34 graphs are used for training and remaining 15 are for testing. In other word, all models will be trained inductively. Note that timesteps are not used in model training. Each graph data contains a pair of adjacency matrix and node feature matrix. To diagnose training performance, each graph data is further split into a train-validation subset using scikit-learn’s `train_test_split` with an 85:15 ratio. The split is performed at the node level.

There are two phase of experiment adn two class of models used: custom VGAE and fraud classifier models. Firstly, one custom VGAE is developed and trained. The VGAE is a generative capable of generating data. The quality of VGAE can be measured by assessing the quality of generated data. To assess this, we train the fraud classifier

models using different composition of original and reconstruct data. In ideal, the model that trained on reconstructed data should perform equally to model that use original model.

3.5.1 Fraud Classifier GNN Model

The task of models is to predict whether given node is fraud or not considering node associated input graph.

The GCN model used here consists of three layers. The first layer performs graph convolution on the node features, reducing them to 128-dimensional node embeddings. These embeddings are then passed to the second layer, which further reduces them to a 2-dimensional vector. Finally, the last layer is a linear layer that outputs the fraud score. Higher scores indicate higher fraudulence, while lower scores suggest the opposite. The pseudo Pytorch Python code would be as follow:

```
GCN()
  GCNConv(165,128)
  GCNConv(128,2)
  Linear(2, 1)
```

For this experiment, both the GAT and GATv2 models share common parameters and settings, differing only in the aggregation part. The GAT model comprises three layers. The first two layers are GAT convolutional layers and last layer is MLP. The model also return similar as in GCN model. The pseudo Pytorch Python code would be as follow:

```
GAT(input_dim = 165, hidden_dim = 32, output_dim=1,heads=2)
  GATConv(input_dim, hidden_dim, heads)
  GATConv(heads * hidden_dim, hidden_dim, heads)
  Linear(heads * hidden_dim, hidden_dim)
  Dropout(0.5)
  Linear(hidden_dim, output_dim)
```

3.5.2 Graph Reconstruction – VGAE

The VGAE model originally propose to only reconstruct adjacency matrix. To also reconstruct node feature, the author develop a node autoencoder and integrate it to VGAE. The integrated VGAE shall refer as custom VGAE in this project. In the node autoencoder, its decoder part reconstruct node feature from concatenated of latent spaces from its encoder and GAE’s encoder. Indeed, the node autoencoder used here is a vanilla autoencoder but with distinct latent spaces obtained from the graph encoder. The custom

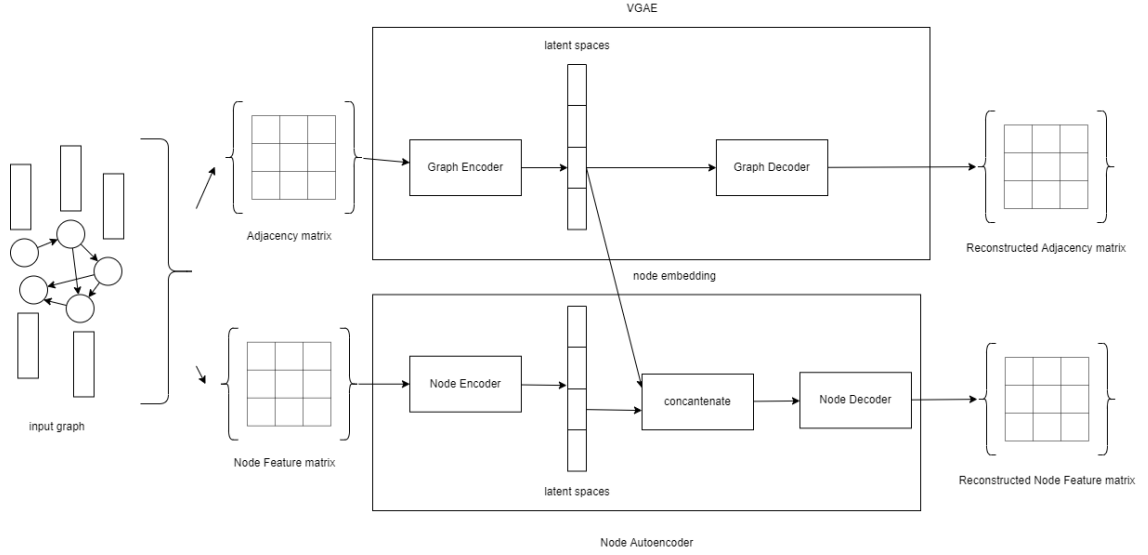


Figure 3.8: Custom VGAE model

VGAE model architecture used in this project is show in the Figure 3.8. The pseudo Pytorch Python code would be as follow:

For this project, the graph decoder takes inner product of latent spaces z then apply sigmoid to recreate the adjacency matrix. Thus, the one of the dimensions of latent spaces correspond the number of nodes n so that the shape of inner product $z^T z$ is a square matrix with the shape of $n \times n$ representing the reconstructed adjacency matrix. No more activation layers applied in the graph decoder. This means that the learnable parameters are located in graph encoder part.

3.5.3 Training

VGAE

The custom VGAE models undergo 600 epochs of training using the Adam algorithm with a learning rate of 0.001. These trained models will be used to reconstruct the data. The reconstructed data are then incorporated into the training process of the fraud classifier GNN models, with the hope that this integration might improve the models' performance.

The training process of the custom VGAE incorporates three loss functions into one loss function. The first is the reconstruction error, which measures the discrepancy between the original graph and its reconstructed version. The second is the KL divergence (Kullback-Leibler), which helps to regularize the latent space of the graph encoder. Lastly, the mean squared error (MSE) loss is utilized to compare the node features with their corresponding reconstructions. By summing these loss functions, the custom VGAE

optimizes the reconstruction accuracy of the edges while simultaneously preserving the integrity of the node features in the graph.

In the PyTorch Geometric implementation, the reconstruction loss function is defined as the binary cross entropy between positive edges, which are the edges that exist in the graph, and negative sample edges, which are edges that do not exist in the graph. During the calculation, the number of negative samples is the same as the number of positive edges.

The choice of using this reconstruction loss function is driven by considerations of efficiency and simplicity. A more naive approach would involve calculating the binary cross entropy between two full adjacency matrices, but this would require additional computations. Such an approach would be more complex for the model, as it would not only have to learn to predict the presence of edges but also to suppress the absence of other edges. Unless, the graph is not sparse.

Fraud Classifier GNN Model

All fraud classifier GNN models undergo 25 epochs of training using the same Adam algorithm with a learning rate of 0.01 and weight decay of 0.00001. The models are trained using various combinations of original and reconstructed data derived from VGAE. The mixing compositions consist of purely original data (100% original, 0% reconstructed), a balanced mix of original and reconstructed data (50% original, 50% reconstructed), and exclusively reconstructed data (0% original, 100% reconstructed) using VGAE.

To ensure consistency in the size of the training datasets across all models, a straightforward approach is adopted. The initial set of 34 graphs is considered the original data set. For the 50-50 mix, an equal number of original and reconstructed graphs are combined to maintain balance. Additionally, to create the 100-0 data, the original data set is duplicated, resulting in two identical sets of original graphs. Lastly, 0-100 composition is two identical sets of reconstructed data. This methodology guarantees uniformity and fairness in the training process across all models.

Overall, a total of nine models are obtained by training three different types of GNN models with the aforementioned dataset compositions. The source code is located at the repo <https://github.com/taimoon/graph-for-fraud-detection>.

CHAPTER 4

DATA ANALYSIS, RESULTS AND DISCUSSION

4.1 Elliptic Dataset Summary

In the Elliptic dataset, there are 203,769 node transaction and 234,355 direct edge transaction flow. The nodes are identified by unique transaction ID. 2% of the nodes (4,545) are illicit and 21% (42,019) are licit. Remaining nodes are unclassified. Each node has 166 features whose first 94 features are local information and remaining 72 are aggregated features. The aggregation performs are one-hop backward/forward from the center node. Of course, these features are masked. There is not associated edge features. Note that each node is an transaction and the edge represent the flow from previous to next transaction.

In addition to these node features, each node is also assigned which approximates the time when the corresponding transaction was confirmed by the Bitcoin network. The timestamps are divided into 49 distinct time steps, spaced approximately two weeks apart. Within each time step, there exists a *single connected component* comprising transactions that occurred within a three-hour window of each other; there are no edges connecting nodes across different time steps. The single connected component should look similar as the Figure 4.3 whose all nodes are connected.

As mentioned in previous paragraph, there is no edge relating nodes across time steps, then the datasets can split into 49 independent graphs. It does forming a sequence of graph based on time steps, but there is no relationship between them. Although it is temporal graph by definition, it is not useful. As alternative, the transactions must be made before and after; the edges are connected while maintaining the correct chronological order. This is the temporal information contain in the graph.

Remaining (77%) the transactions are unclassified and big in size relative to labelled transactions, it results the visualized graph have many gray-coloured nodes as shown in the Figure 4.3.

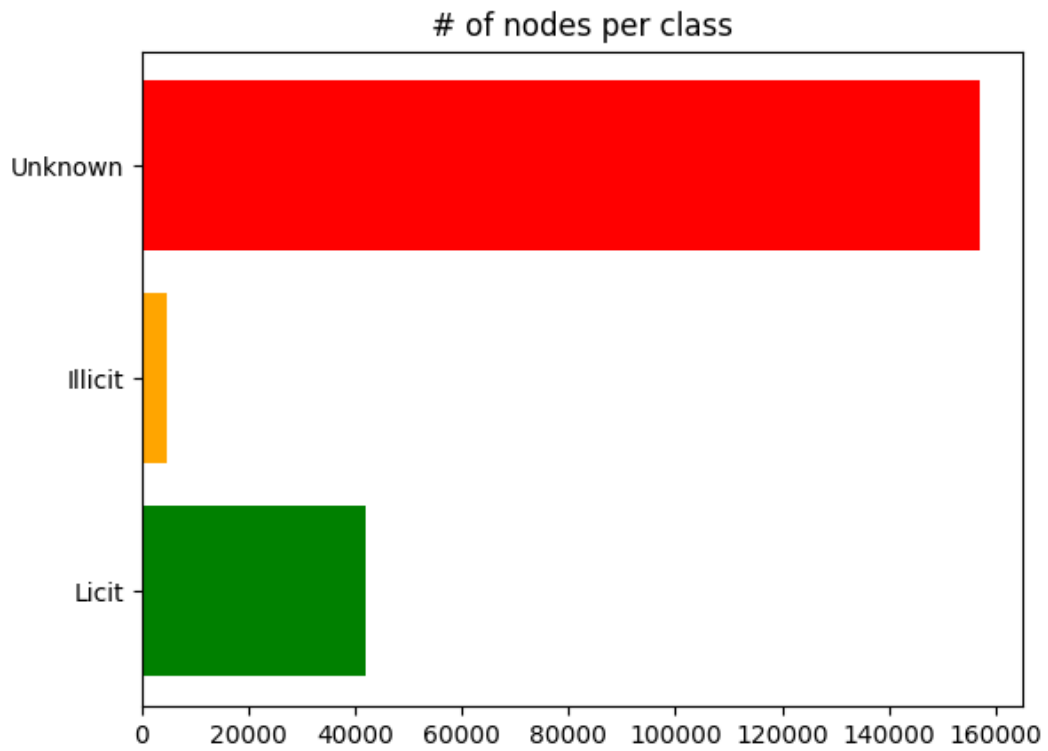


Figure 4.1: Count of transactions by classes

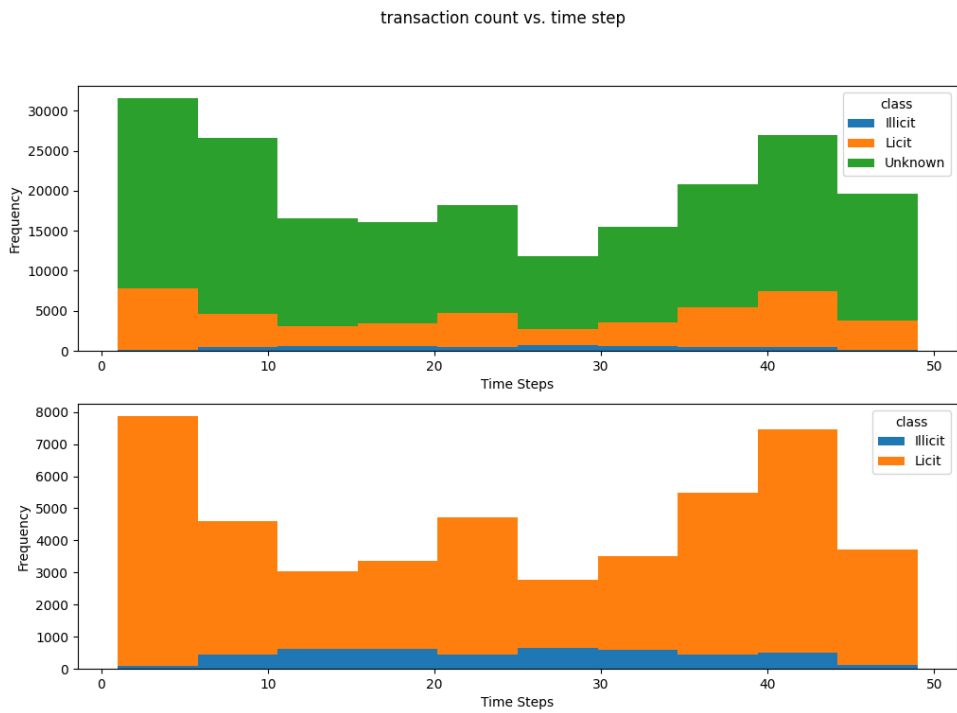


Figure 4.2: Count of transaction nodes by class across the time steps

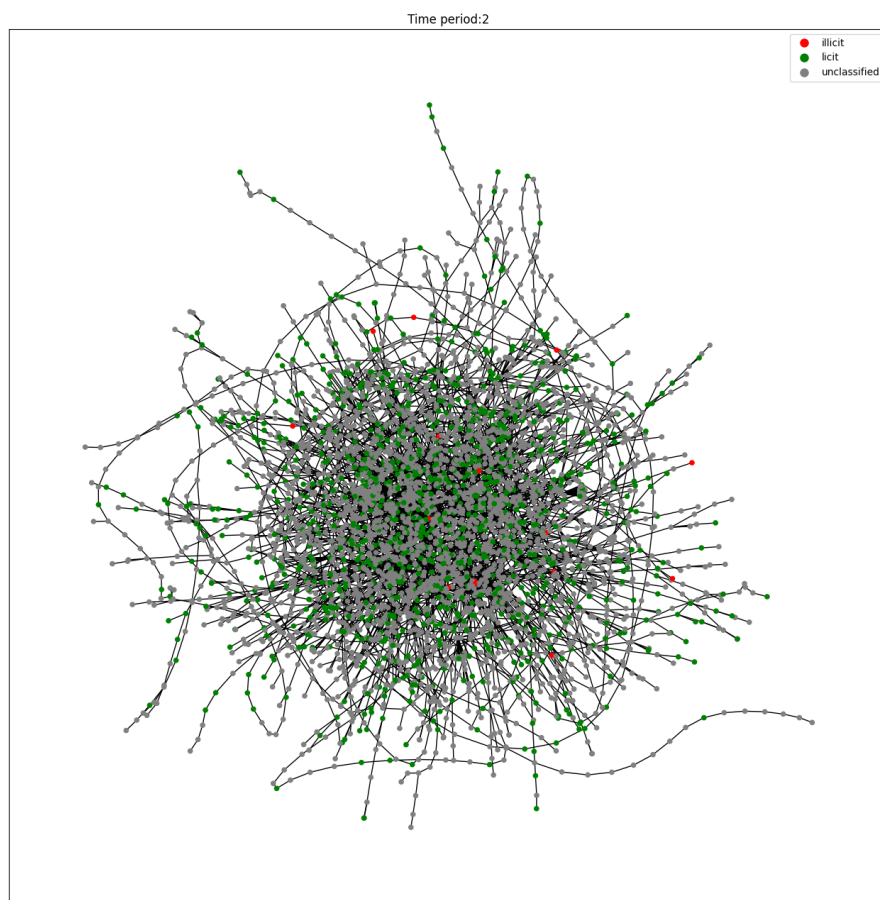


Figure 4.3: Transaction Graph at time steps = 2

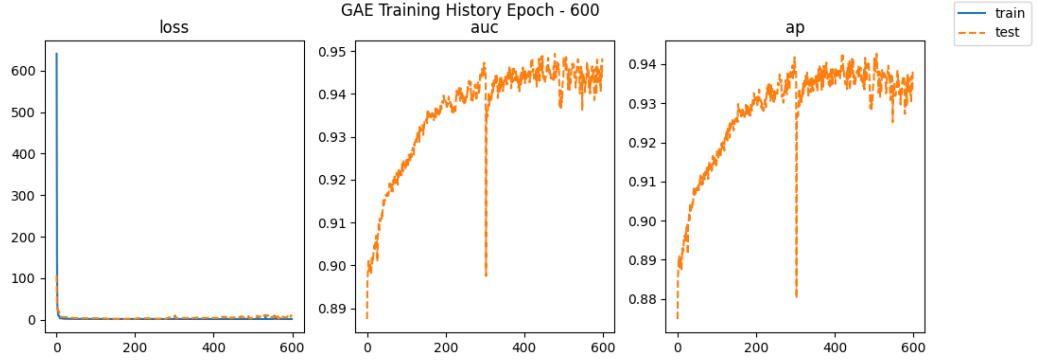


Figure 4.4: Training History of VGAE

4.2 VGAE

In this custom VGAE, the accuracy of the reconstructed edges is evaluated using the Area Under the ROC (AUCROC) Curve and average precision (AP) metrics. Monitoring the loss plot, depicted in Figure 4.4, is essential in training deep learning models. The plot helps determine if the model is converging and making progress towards minimizing the loss. Based on this information, informed decisions can be made regarding hyperparameter tuning, model architecture modifications, or adjusting the training strategy to improve overall effectiveness. The convergence shown in the loss plot implies that the model is gradually minimizing the loss function and approaching an optimal state. Similarly mention in the section experiment setup, in the Figure 4.4, the AUCROC and average precision are calculated between positive and negative edges rather than between two full adjacency matrices.

To reconstruct the graph from VGAE's output, the continuous values in the returned adjacency matrices need to be discretized. This is achieved by applying a threshold to convert the continuous values into binary values representing the presence or absence of edges in the reconstructed graph.

The effect of the threshold can be evaluated by measuring the discrepancy between the reconstructed and original graphs using metrics calculated based on the full adjacency matrices. The transaction graphs are sparse where the presence of edges is much lower compared to the absence of edges, then precision and recall are more appropriate metrics to consider. The area under the precision against recall curve shown in Figure 4.5 is the average precision value. By properly scaling the graph, it can be estimated that the area under the curve is close to 0. It is extremely bad comparing with the training history. It must be noted that the comparison is not appropriate because they use different sample.

The comparison provides important insight. This is because the loss value is calculated based on smaller subset of positive and negative edges rather than full adjacency matrices. As discussed in the section experiment setup, the defined loss function does

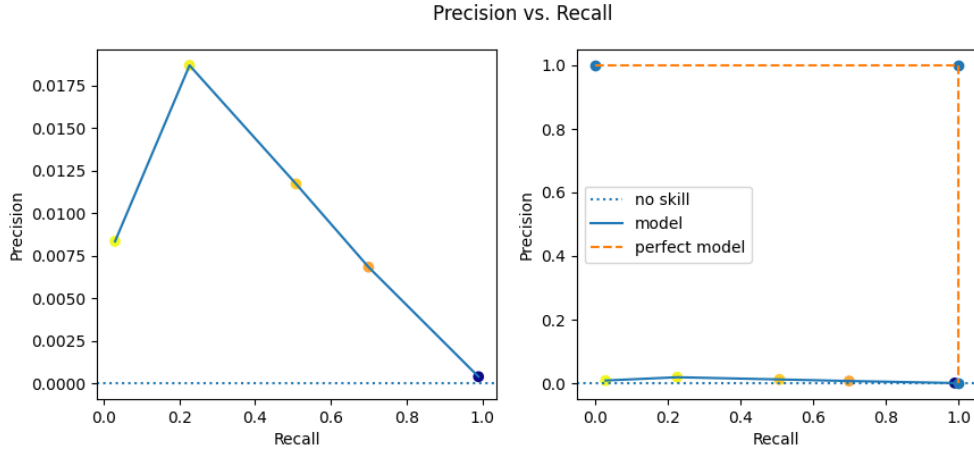


Figure 4.5: Precision Recall Curve Plot

not penalize the model predict wrong edge presence that is outside of the subset. Consequently, the model overestimate the edge presences causing low precision but high recall.

The overestimation causes the reconstructed adjacency matrices explodes in number of edges. As shown in the Figure 4.6, the number of edges in reconstructed graphs increases exponentially as threshold value decreases. Ideally, the points should lies on 0 so that the size is less than 10 times of original. The highest threshold value still double the edge number of original graph. The increased size in reconstructed data required more memory during the next phase of training and this is the reason of discussing the impact of different thresholding value.

It could be resolved by discarding all edges which are not originally in the input graph; using the original information. This project does not adopt this to see how the classifier models behave on these redundant reconstructed data.

4.3 Model that use Original Dataset

The validation set refers to the remaining 15 testing graphs unless mention otherwise. From the Figure 4.7, GAT and GATv2 models have higher AUCROC and lower loss than GCN model; GAT variant model is better than GCN. In additional, all models converge after 5th epoch.

The number of illicit transactions are very low comparing to the number of licit transactions. High accuracy does not imply that all illicit transactions are detected since detecting all licit transactions is enough to contribute high accuracy. Therefore, precision and recall are more meaningful metrics. In the context of fraud detection, high recall indicates that most illicit transactions are flagged, while high precision means that most flagged transactions are actually fraudulent, minimizing false alarms.

Achieving a balance between precision and recall is crucial. If the priority is to de-

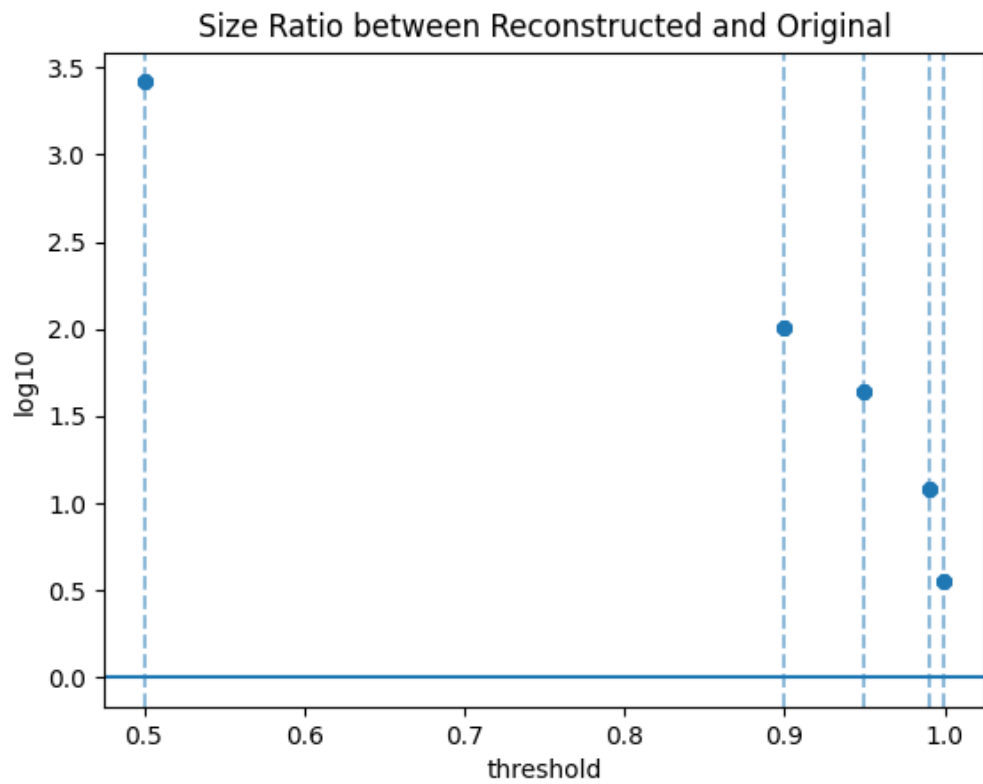


Figure 4.6: Size ratio between reconstructed and original edge numbers

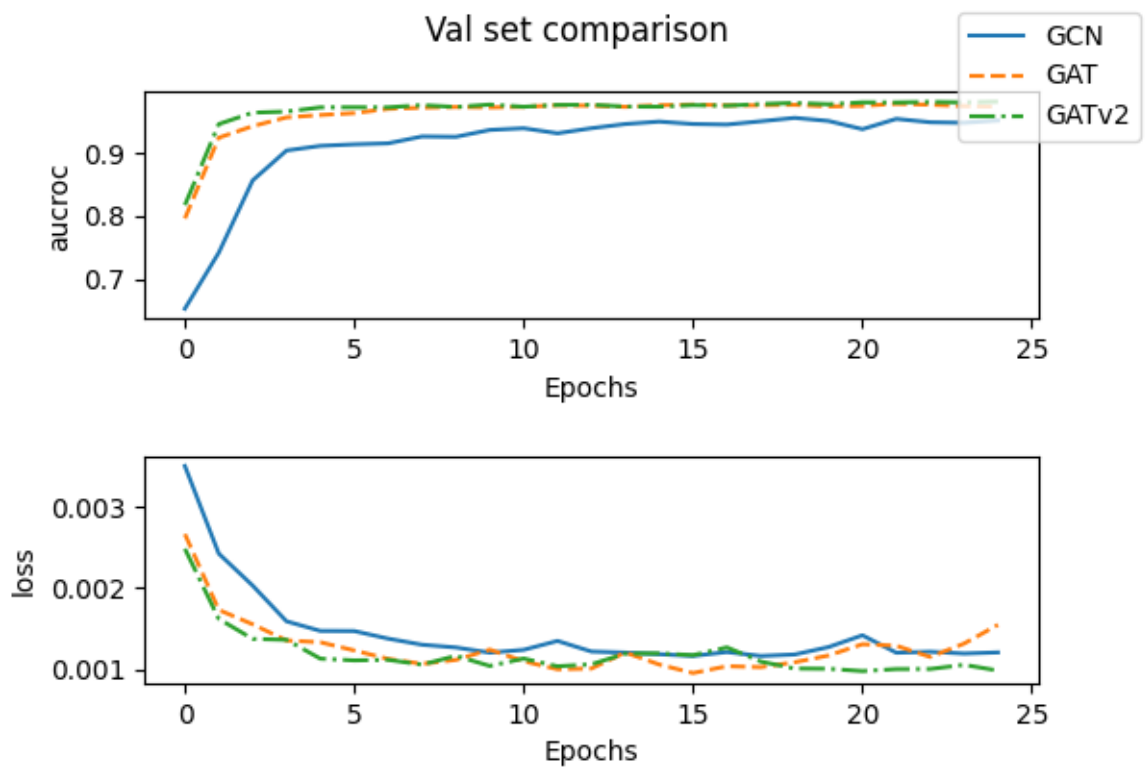


Figure 4.7: Model Performance Comparison (Train Using Original Data)

tect all illicit transactions (high recall), the model may generate many false alarms, resulting in low precision. Conversely, if the goal is to minimize false alarms (high precision), the model may overlook some fraudulent activities. Finding the right balance between precision and recall is essential for an effective fraud detection system.

Each metrics in the Figure 4.8 are calculated for each timestep graph and the vertical line divides first 34 training datasets and remaining 15 testing datasets. As in the figure, all models generally does not recall illicit transactions for future and unseen observations. In training datasets, the models do have the ideal performance having high precision and high recall. High precision and high recall means that most illicit transactions are detected while not misfire too many licit transactions.

4.4 Visualizing the Prediction

The Figure 4.10 visualizes the prediction made by the fraud classifier GAT model at 12th time steps.

4.5 Model that use Reconstructed Data

The figures 4.11, 4.12 and 4.13 visualize the performance of models which are trained using different dataset composition tested against original dataset.

The suffix mix indicates that the model is trained on a balanced 50-50 mix of original and reconstructed data, while the suffix recon signifies that the model is exclusively trained on 0-100 reconstructed data generated by the custom VGAE. When it comes to accuracy and F1 micro score, it is generally observed that the recon and mix models perform relatively worse compared to the models trained solely on original data. This suggests further explanations on why the inclusion of reconstructed or mixed data might worsen the performance of the models.

Reconstructed data tends to result in lower performance compared to original data, as some information is lost during the reconstruction process. However, the fluctuation patterns of the evaluation metrics for reconstructed models closely resemble those of the original models, indicating that the custom VGAE is able to include the underlying patterns in reconstructing graph that is resemble to original graph. Notably, in the first 3 time steps, recon models have higher recall but lower precision than original counterpart.

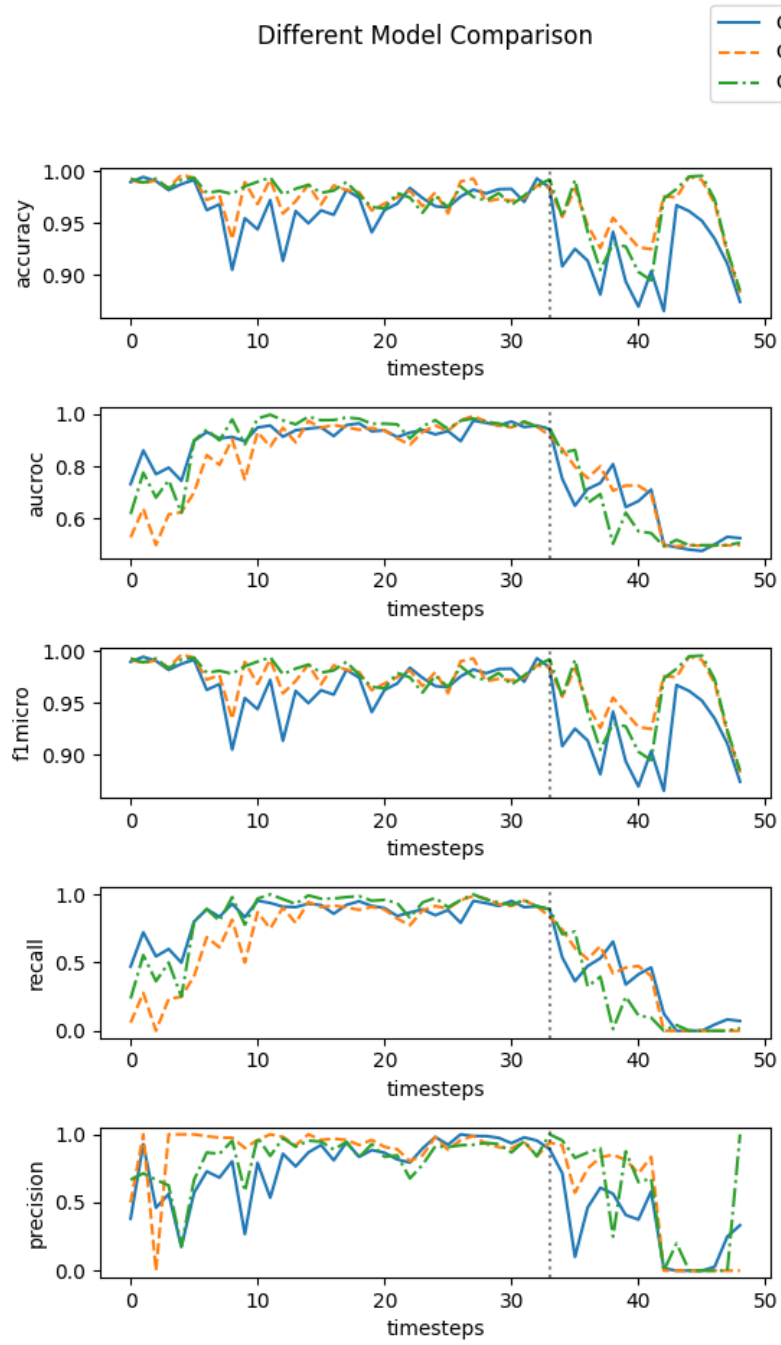


Figure 4.8: Model Comparison

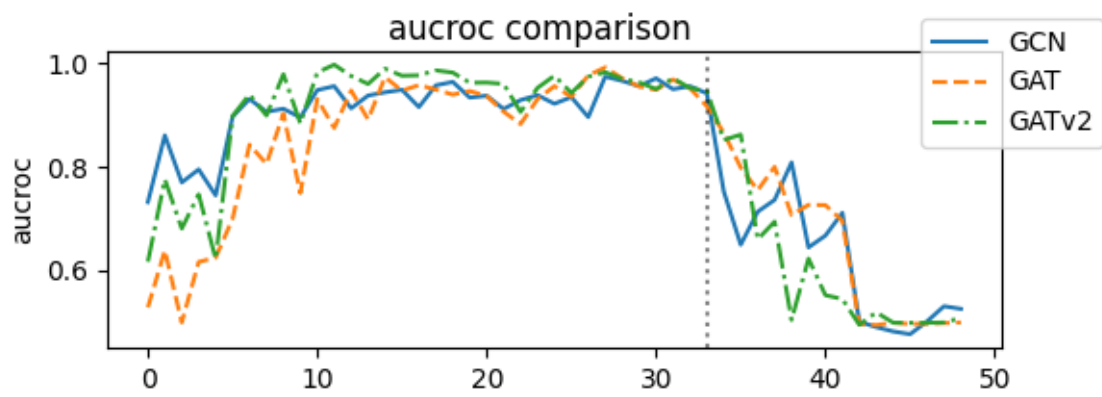


Figure 4.9: AUCROC: Model Comparison (Train Using Original Data)

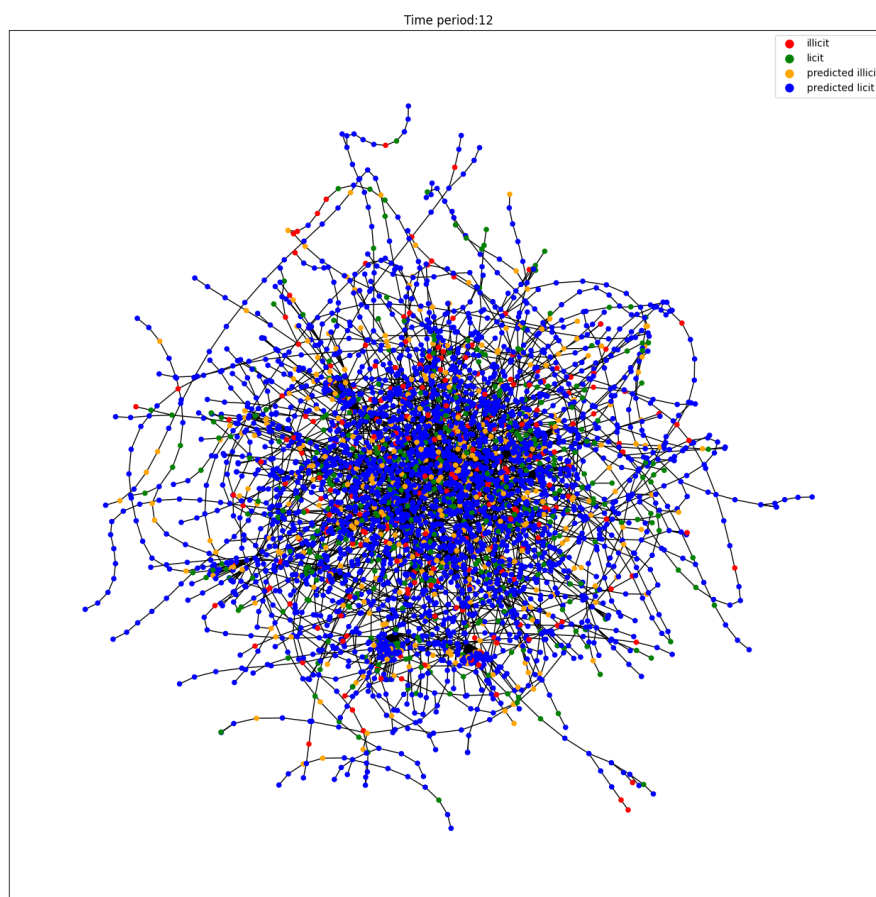


Figure 4.10: Transaction Graph labelled by GAT model at time steps = 12

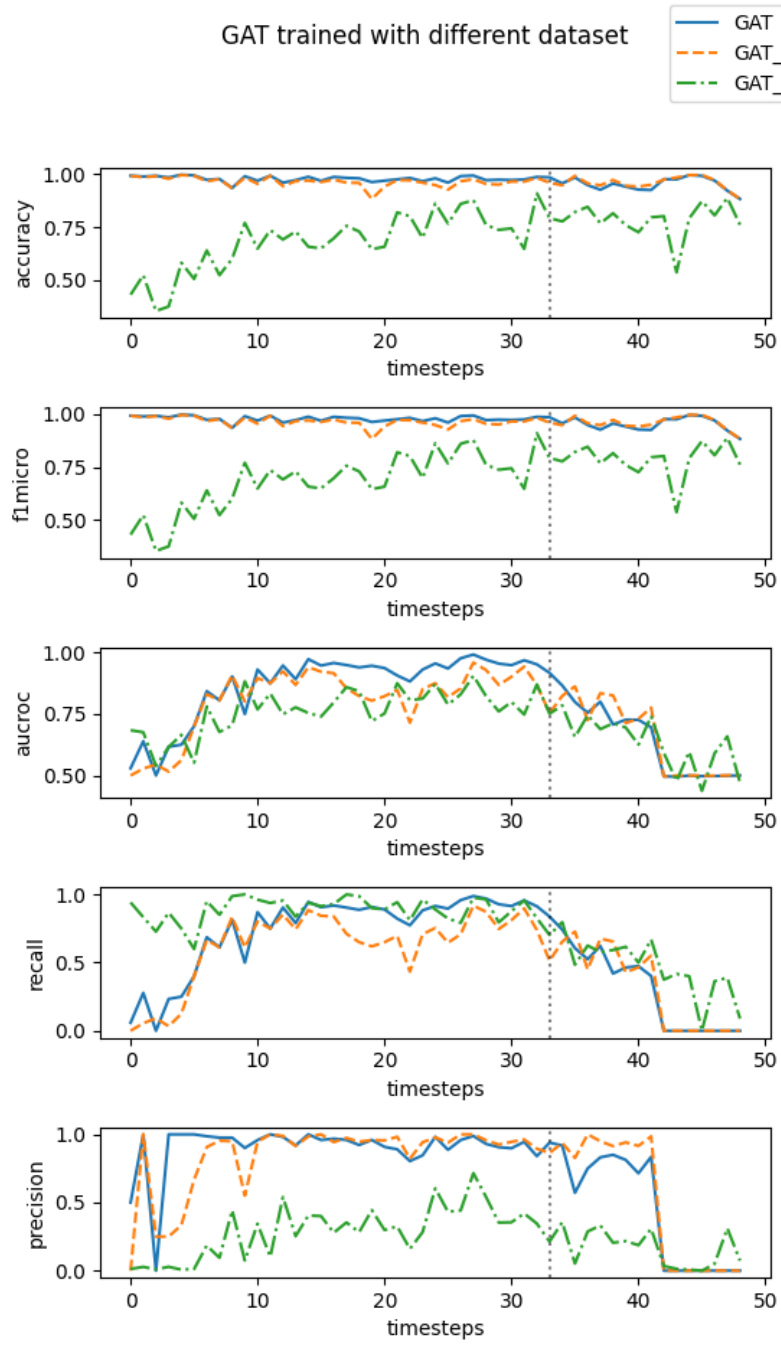


Figure 4.11: GAT trained on different dataset

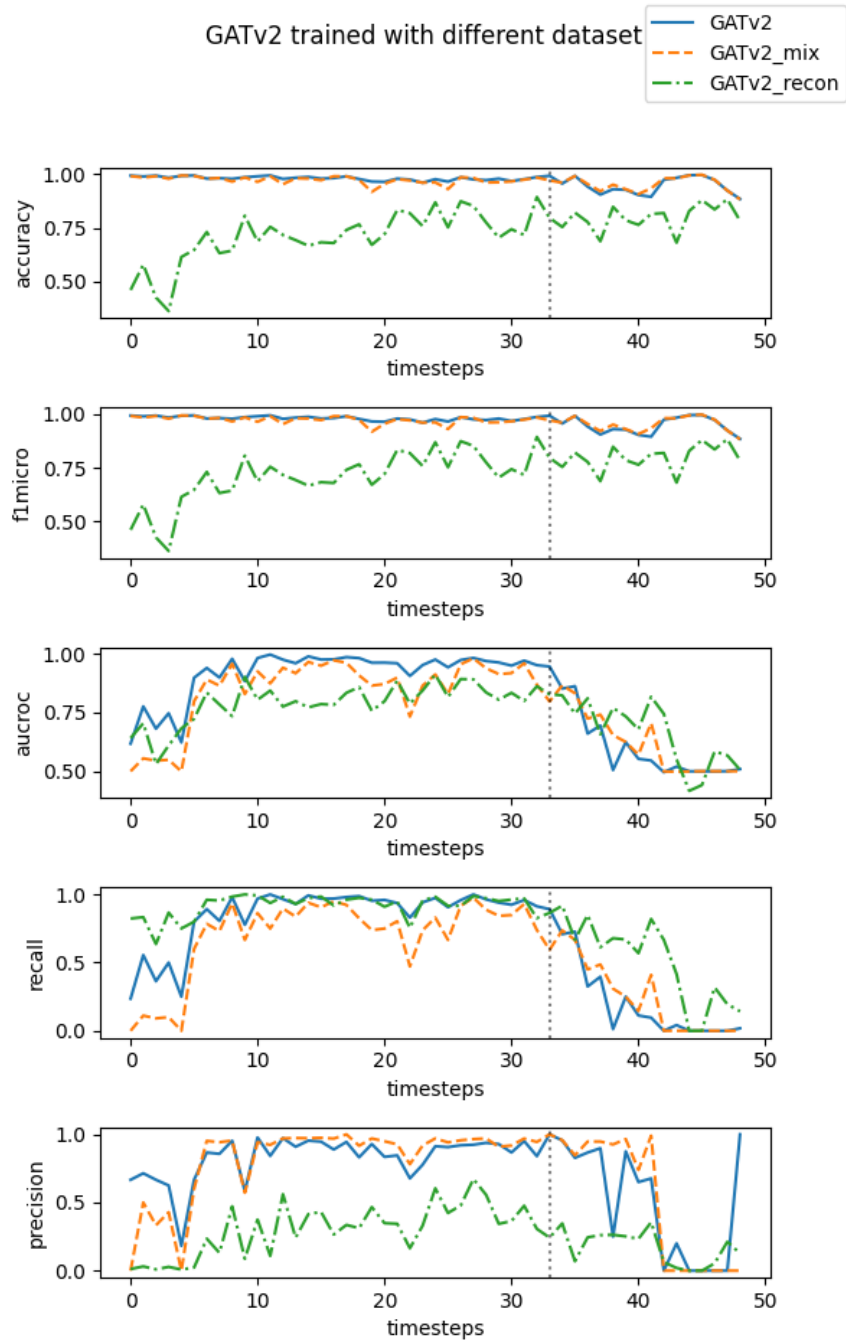


Figure 4.12: GATv2 trained on different dataset

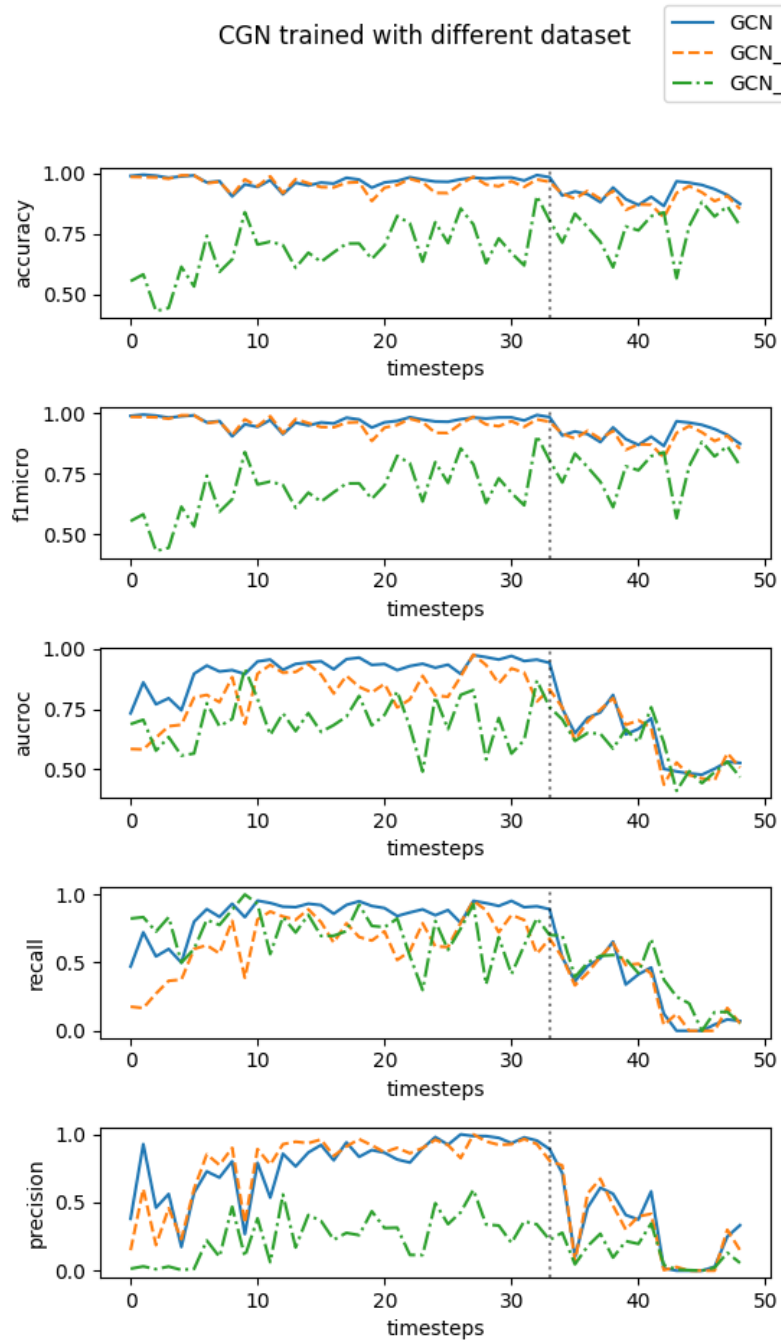


Figure 4.13: GCN trained on different dataset

4.6 Summary

In conclusion, this project focused on developing a custom Variational Graph Autoencoder (VGAE) that incorporates a node autoencoder into the VGAE framework. However, it was observed that when the defined loss function did not penalize the custom VGAE for wrongly predicting edge presence, it leads to an overestimation of edge presence and resulting in an explosion of edges in the reconstructed graph.

To investigate the impact of using generated data, nine models were trained on three different dataset compositions: purely original data, purely reconstructed data, and a mixture of both. To ensure fairness during model training, these datasets were equal in size. Three types of Graph Neural Network (GNN) models were implemented and tested: Graph Attention Network (GAT), GATv2, and Graph Convolutional Network (GCN). This resulted in a total of nine trained models that were evaluated and discussed.

Generally, regardless of the dataset composition, GAT and GATv2 performed better than GCN in terms of the Area Under the ROC Curve (AUCROC). However, they were not effective in recalling illicit transactions in the inductive setting. The models that utilized reconstructed data during training performed worse than those that used only original data. Despite this, the models exhibited similar fluctuation patterns across various metrics, indicating that the reconstructed data closely resembled the original data. This suggests that the custom VGAE was able to simulate fraud to a certain degree.

Answering to the research question.

1. Does graph neural network works in detecting fraud? The GNN models cannot inductively detect illicit transactions that are unseen during training time.
2. Does the custom VGAE simulate the fraudulent behaviour? Is generated dataset resemblance to real dataset? The generated dataset by the custom VGAE is resemblance to original dataset but with caveat of overestimation of edge presence.

The project has reported the observations thus achieved the objectives partially because the model does not perform well in inductive setting regardless the use of augmented dataset.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In summary, this project has developed a custom VGAE for data augmentation purpose. To evaluate the VGAE, 3 different models (GAT, GATv2, GCN) are trained on 3 different dataset composition which comprising of original and reconstructed data; total of nine models are trained. this project highlights the challenges and limitations of utilizing reconstructed data for fraud detection. Further research is needed to address the issue of overestimation and to improve the recall of illicit transactions. Additionally, exploring different GNN architectures and loss functions could potentially enhance the performance of fraud detection models. Overall, this work provides insights into the application of custom VGAEs and their potential in simulating fraud patterns.

5.2 Recommendation

Based on the findings and observations from this project, several recommendations can be made to further study the behavior of fraud detection models using graph neural networks (GNNs) and improve their performance:

1. Utilize Adversarially Regularized Graph Autoencoder (ARGAE): Incorporating ARGAE, which includes a discriminator to differentiate between real and reconstructed graphs, can enhance the robustness of the model. This adversarial technique may improve the quality of the reconstructed data, leading to better fraud detection outcomes.
2. Explore Various Network Architectures: Experimenting with different network architectures can help increase the capabilities of the models. Consider adding pre-processing and post-processing layers to the GNN model to capture more complex patterns and improve its ability to detect fraud.
3. Develop Alternative and Computationally Efficient Loss Functions: Investigate the development of alternative loss functions that consider the full adjacency matrices

in training the Graph Autoencoder (GAE). These loss functions should be efficient computationally while ensuring accurate reconstruction.

4. **Employ Ensemble Methods:** Explore ensemble methods, which involve combining multiple models, to improve the overall performance of fraud detection. Ensemble techniques can leverage the strengths of different models and enhance their predictive

REFERENCES

- ACCA. (n.d.). *Data analytics and the auditor*. Retrieved 2022-11-18, from <https://www.accaglobal.com/in/en/student/exam-support-resources/professional-exams-study-resources/p7/technical-articles/data-analytics.html>
- Akkeren, J. V. (2018, June). Fraud triangle: Cressey's fraud triangle and alternative fraud theories. In D. C. Poff & A. C. Michalos (Eds.), *Encyclopedia of business and professional ethics*. Cham, Switzerland: Springer. Retrieved from <https://eprints.qut.edu.au/214688/>
- Beals, M., DeLiema, M., & Deevy, M. (2015). *Framework for a taxonomy of fraud*. Stanford Center On Longevity. Retrieved from <https://longevity.stanford.edu/framework-for-a-taxonomy-of-fraud/>
- Brody, S., Alon, U., & Yahav, E. (2022). How attentive are graph attention networks?
- Chang, Y., Li, P., Sasic, R., Afifi, M. H., Schweighauser, M., & Leskovec, J. (2020). F-FADE: frequency factorization for anomaly detection in edge streams. *CoRR*, *abs/2011.04723*. Retrieved from <https://arxiv.org/abs/2011.04723>
- Cheng, D., Xiang, S., Shang, C., Zhang, Y., Yang, F., & Zhang, L. (2020, Apr.). Spatio-temporal attention-based neural network for credit card fraud detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01), 362-369. Retrieved from <https://ojs.aaai.org/index.php/AAAI/article/view/5371>
<https://doi.org/10.1609/aaai.v34i01.5371>
- Eswaran, D., & Faloutsos, C. (2018). Sedanspot: Detecting anomalies in edge streams. In *2018 IEEE International Conference on Data Mining (ICDM)* (p. 953-958). <https://doi.org/10.1109/ICDM.2018.00117>
- Ferreira, A. (2004). Building a reference combinatorial model for manets. *IEEE Network*, 18(5), 24-29. <https://doi.org/10.1109/MNET.2004.1337732>
- Flocchini, P., Mans, B., & Santoro, N. (2013). On the exploration of time-varying networks. *Theoretical Computer Science*, 469, 53-68. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0304397512009504>
<https://doi.org/https://doi.org/10.1016/j.tcs.2012.10.029>
- Frankenfield, J. (2022). *Data analytics: What it is, how it's used, and 4 basic techniques*. Retrieved 2022-11-18, from <https://www.investopedia.com/terms/d/data-analytics.asp>
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (<http://www.deeplearningbook.org>)
- Guo, X., & Zhao, L. (2020). A systematic survey on deep generative models for graph generation. *CoRR*, *abs/2007.06686*. Retrieved from <https://arxiv.org/abs/2007.06686>

- Hajdu, L., & Krész, M. (2020, August). Temporal network analytics for fraud detection in the banking sector. In (p. 145-157). https://doi.org/10.1007/978-3-030-55814-7_12
- Hooi, B., Shah, N., Beutel, A., Günnemann, S., Akoglu, L., Kumar, M., ... Faloutsos, C. (2015). BIRDNEST: bayesian inference for ratings-fraud detection. *CoRR*, *abs/1511.06030*. Retrieved from <http://arxiv.org/abs/1511.06030>
- Kipf, T. N., & Welling, M. (2016). Variational graph auto-encoders.
- Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks.
- Le Borgne, Y.-A., Siblini, W., Lebichot, B., & Bontempi, G. (2022). *Reproducible machine learning for credit card fraud detection - practical handbook*. Université Libre de Bruxelles. Retrieved from <https://github.com/Fraud-Detection-Handbook/fraud-detection-handbook>
- Lopez-Rojas, E. A., Elmir, A., & Axelsson, S. (2016). Paysim: a financial mobile money simulator for fraud detection..
- Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q. Z., ... Akoglu, L. (2021). A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 1-1. <https://doi.org/10.1109/TKDE.2021.3118815>
- Manjunatha, H. C., & Mohanasundaram, R. (2018). Brnads: Big data real-time node anomaly detection in social networks. In *2018 2nd international conference on inventive systems and control (icisc)* (p. 929-932). <https://doi.org/10.1109/ICISC.2018.8398937>
- Michail, O. (2016). An introduction to temporal graphs: An algorithmic perspective. *Internet Mathematics*, 12(4), 239-280. Retrieved from <https://doi.org/10.1080/15427951.2016.1177801> <https://doi.org/10.1080/15427951.2016.1177801>
- Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., ... Leiserson, C. E. (2019). *Evolvegcnn: Evolving graph convolutional networks for dynamic graphs*. arXiv. Retrieved from <https://arxiv.org/abs/1902.10191> <https://doi.org/10.48550/ARXIV.1902.10191>
- Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, aug). DeepWalk. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM. Retrieved from <https://doi.org/10.1145/2623330.2623732> <https://doi.org/10.1145/2623330.2623732>
- Pourhabibi, T., Ong, K.-L., Kam, B. H., & Boo, Y. L. (2020). Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems*, 133, 113303. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0167923620300580> <https://doi.org/10.1016/j.dss.2020.113303>
- Razzak, M. I., Imran, M., & Xu, G. (2019, mar). Big data analytics for pre-

- ventive medicine. *Neural Computing and Applications*, 32(9), 4417–4451. <https://doi.org/10.1007/s00521-019-04095-y>
- Rocheteau, E., Tong, C., Velickovic, P., Lane, N. D., & Liò, P. (2021). Predicting patient outcomes with graph representation learning. *CoRR*, *abs/2101.03940*. Retrieved from <https://arxiv.org/abs/2101.03940>
- Rocheteau, E., Tong, C., Veličković, P., Lane, N., & Liò, P. (2021). *Predicting patient outcomes with graph representation learning*.
- Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltchko, A. B. (2021). A gentle introduction to graph neural networks. *Distill*. (<https://distill.pub/2021/gnn-intro>) <https://doi.org/10.23915/distill.00033>
- SASTRY, S., LEE, T. H., & TEOH, M. T. T. (2021, Jul.). The use of blockchain technology and data analytics in the audit profession. *Quantum Journal of Social Sciences and Humanities*, 2(4), 67-86. Retrieved from <https://www.qjssh.com/index.php/qjssh/article/view/89> <https://doi.org/10.55197/qjssh.v2i4.89>
- Sethia, A., Patel, R., & Raut, P. (2018). Data augmentation using generative models for credit card fraud detection. In *2018 4th international conference on computing communication and automation (iccca)* (p. 1-6). <https://doi.org/10.1109/CCAA.2018.8777628>
- Stedman, C. (n.d.). *data analytics (da)*. Retrieved 2022-11-18, from <https://www.techtarget.com/searchdatamanagement/definition/data-analytics>
- Suzumura, T., & Kanezashi, H. (2021). *Anti-Money Laundering Datasets: InPlusLab anti-money laundering datadatasets*. <http://github.com/IBM/AMLSim/>.
- Taylor, K., Goeringer, S., Winter, E., & Khalilian, M. (2020). *A taxonomy of fraud experienced by network service providers* (Tech. Rep.). SCTE ISBE.
- Tsai, C.-W., Lai, C.-F., Chao, H.-C., & Vasilakos, A. V. (2015, October). Big data analytics: a survey. *Journal of Big Data*, 2(21). <https://doi.org/10.1186/s40537-015-0030-3>
- Tyagi, S. (2003, January). Using data analytics for greater profits. *Journal of Business Strategy*, 24(3). <https://doi.org/10.1108/02756660310734938>
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks.
- Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., ... Guo, M. (n.d.). *Graphgan: Graph representation learning with generative adversarial nets*. Retrieved from <https://arxiv.org/abs/1711.08267> <https://doi.org/10.48550/ARXIV.1711.08267>
- Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). *Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics*.

- Zheng, P., Yuan, S., Wu, X., Li, J. Y., & Lu, A. (2018). One-class adversarial nets for fraud detection. *ArXiv, abs/1803.01798*.
- Zheng, Y.-J., Zhou, X.-H., Sheng, W.-G., Xue, Y., & Chen, S.-Y. (2018). Generative adversarial network based telecom fraud detection at the receiving bank. *Neural Networks, 102*, 78-86. Retrieved from <https://www.sciencedirect.com/science/article/pii/S08933608018300698> <https://doi.org/https://doi.org/10.1016/j.neunet.2018.02.015>
- Zhou, H., Sun, G., Fu, S., Wang, L., Hu, J., & Gao, Y. (2021). Internet financial fraud detection based on a distributed big data approach with node2vec. *IEEE Access, 9*, 43378-43386. <https://doi.org/10.1109/ACCESS.2021.3062467>