U.PORTO

FEP FACULDADE DE ECONOMIA
UNIVERSIDADE DO PORTO

Comparative Study of VAE and GAN Based Models for Graph Anomaly Detection

Diogo Gomes Abreu

Dissertation

Master in Modelling, Data Analysis and Decision Support Systems

Supervised by:

PhD João Gama

PhD Bruno Veloso

2023

Acknowledgements

This dissertation marks the end of yet another stage of my academic life. These last two years at FEP have been an amazing experience that allowed me to learn and grow not only academically and professionally by gaining and developing valuable knowledge and skills, but also personally through the interactions with the many new people I met, for which I am deeply grateful.

I would like to express my sincere gratitude to all the professors of the MADSAD master's program for generously sharing their knowledge. Noteworthy recognition goes to my supervisor Professor João Gama and co-supervisor Professor Bruno Veloso, who not only provided invaluable guidance and support, leading me to successfully complete this final step of my academic journey, but also made me develop a deeper appreciation for the field of data mining.

I would like to seize this opportunity to send a heartfelt thanks to my mother, Olga, father, Manuel, and grandmother, Rosa, whose support, encouragement and love were essential to surpass this final hurdle. You made sure nothing was lacking for me to be able to accomplish my goals. I would also like to leave a special note of thanks to my brother, Eduardo, who generously lent me his laptop to accelerate some of the programming tasks for my dissertation. Your generosity and assistance were of immeasurable value during this journey.

I also want to express my profound appreciation and affection to Sofia. You are a source of support, encouragement, and inspiration ever since I met you. Your belief in me was what made me push through and be able to conquer the final hurdle in this academic journey in due time. I am immensely grateful to have you by my side, whether during challenging or joyful times. I am looking forward to keep sharing many more adventures with you.

Last but not least, I would like to thank my closest friend with whom I share my adventures and misadventures, fortune and misfortune. Their friendship has been a constant source of strength and a reminder to enjoy the life as it comes.

Abstract

Graph anomaly detection poses a critical challenge in various domains, such as network security or fraud detection, where data points' complex structures and connections can be leveraged to identify anomalous instances or interactions.

This dissertation aims to explore the complex problem of graph anomaly detection by applying generative deep learning models, namely generative adversarial networks (GAN) and variational autoencoders (VAE).

The primary goal of this project is to develop, train and test straightforward and comprehensible VAE and GAN models to perform the task of graph anomaly detection and compare their development and training processes, as well as evaluate their performance versus their complexity. This research aims to clarify some strengths and limitations of the models, as well as the complexity and limitations of graph data, especially when faced with limited computational resources.

Throughout the experimentation and analysis of the models, VAE-based models were more promising due to their relatively better performance, simpler architecture and training procedure, which was able to leverage the power and convenience of callbacks, such as early stopping.

During the development of this project, some of the nuances related to graph data were dealt with. A complex pipeline for data processing was created to use graph data. This led to the problem of high dimensionality, as it became impossible to use extremely large subgraphs, as they require exponentially more computational resources. Furthermore, the need for real-world data tailored to graph anomaly detection with ground-truth labels poses a major obstacle in the development of this field.

In summary, this dissertation tackles the challenges of working with high-dimensional data structures and offers insight into the development, training processes and application of simple VAE and GAN-based models to graph anomaly detection.

Keywords: Deep Learning, Generative Models, Graph Anomaly Detection

Resumo

Deteção de anomalias em grafos posiciona-se como um desafio crítico em várias áreas, como segurança de redes e deteção de fraude, onde a estrutura complexa e conexões de pontos de dados podem ser usados para identificar casos ou interações anómalas.

Esta dissertação pretende explorar o complexo problema de deteção e anomalias em grafos através da aplicação de modelos generativo de deep learning, nomeadamente generative adversarial networks (GAN) e variational autoencoders (VAE).

O principal objetivo deste projeto é desenvolver, treinar e testar modelos VAE e GAN simples e compreensivos para realizar a tarefa de deteção de anomalias em redes e comparar os processos de construção e treino, assim como avaliar o seu desempenho comparado à sua complexidade. Esta investigação tenciona clarificar algumas das forças e limitações destes modelos, assim como a complexidade e limitações dos grafos, especialmente quando confrontados com recursos computacionais limitados.

Através das experiências e análise dos modelos, concluiu-se que os modelos baseados em VAE são mais promissores devido ao seu desempenho relativamente melhor e arquitetura e processo de treino mais simples, no qual foi possível utilizar facilmente o poder e comodidade de callbacks, como a paragem antecipada.

Durante o desenvolvimento deste projeto, algumas das nuances relacionadas com grafos foram resolvidas. Um pipeline complexo de processamento de dados foi criado para permitir a utilização de grafos. Isto levou ao problema de alta dimensionalidade, pois tornou-se impossível utilizar grafos de dimensões exageradas devido a estes requererem exponencialmente mais recursos computacionais. Além disso, a necessidade de utilização de dados reais corretamente classificados e adaptados à tarefa de deteção de anomalias em grafos apresenta-se como o grande obstáculo ao desenvolvimento deste área.

Em suma, esta dissertação aborda os desafios de trabalhar com dados estruturados de alta dimensionalidade e fornece conhecimento sobre os processos de construção, treino e aplicação de modelos simples baseados em VAE e GAN para a tarefa de deteção de anomalias em redes.

Palavras-chave: Deep Learning, Modelos Generativos, Deteção de Anomalias em Grafos

Glossary

AUC-AP - Area Under the Average Precision
AUC-ROC - Area Under the Receiving Operating Curve
API - Application Programming Interfaces
AP - Average Precision
DCGAN - Deep Convolutional Graph Adversarial Networks
f-AnoGAN - Fast Unsupervised Anomaly Detection with Generative Adversarial Networks
GAN - Generative Adversarial Networks
KPIs - Key Performance Indicators
KL - Kullback–Leibler divergence
LSTM - Long Short Term Memory Network
OCAN - One-class Adversarial Nets
ROC - Receiver Operating Characteristic
VAE - Variational Autoencoders
WGAN - Wasserstein Generative Adversarial Networks

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Anomaly detection aims to identify rare behaviours and patterns in data. Its versatility enabled the application of several techniques to a wide range of areas and industries, such as networks, financial transactions, manufacturing and fake news detection, and contributes towards trustworthiness, efficiency, effectiveness and security in systems and processes. It has been explored by several different authors since the 19^{th} century, which developed diverse methods to approach this problem.

Graph anomaly detection is a branch of anomaly detection which functions under the assumption that real-world data cannot always be treated as independent data points. So, it leverages the power of graph structures to represent data with its connections and relationships to identify anomalies based not only on the attributes of the data but also on its structure.

In recent years, deep learning has been applied to graph anomaly detection. Due to its capabilities to learn complex patterns and representations from large and high-dimensional data, deep learning techniques can surpass the challenges imposed by the graph's complex, rich and high-dimensional structure, which traditional methods struggle with.

1.2 Problem Description

The problem of graph anomaly detection, more than the difficulties of regular anomaly detection tasks, such as their unknown nature, extremely imbalanced data and constant evolution of normal and anomalous behaviours, deals with the high-dimensional data of the complex structure of graph data.

Due to traditional methods difficulties in understanding the complex connections, deep learning methodologies such as GAN and VAE, surge as an alternative capable of dealing with the these complex structures of data.

Although not specifically designed for this task, these models have proved to be very useful in anomaly detection tasks in other types of data, such as images in the healthcare area. The application of such models brings forth other types of difficulties, such as developing model to efficiently detect anomalies, definition of hyperparameters and monitoring the training of these models.

1.3 Objectives

This study aims to develop and compare two different deep learning generation-based methods for graph anomaly detection using Variational Autoencoders (VAE) and Generative Adversarial Networks (GAN), which are increasingly being used in anomaly detection tasks. The comparison is conducted between two different datasets, and it aims to go beyond evaluating the performance indicators to analyse the developmental trajectories of both models.

Another key objective of this project is to design a robust pipeline tailored to handle high-dimensional graph data effectively.

1.4 Methodology

This research focuses on developing anomaly detection GAN and VAE models using TensorFlow. The data is divided into normal and anomalous. While training and validation sets are created with only normal subgraphs, anomalies are injected into the test set, as a one-class learning approach will be used.

While keeping a straightforward architecture, the models are developed through a trial-and-error approach to tune both the architecture and the hyperparameters in the training step. While VAE training uses a callback and validation set to avoid overfitting, GAN uses a custom training loop, which creates significant difficulties when integrating callbacks and a validation set, rendering it impractical to implement these elements.

Finally, to evaluate their performance, ROC, Precision-Recall curves and other metrics were used.

1.5 Dissertation Structure

The four chapters that compose this dissertation each correlate to a different stage of the project's development. Each chapter clarifies the trajectory by outlining the procedures used to reach the project's conclusion.

Starting with a literature review, which approaches the topics of anomaly detection, graph data and their combination, exposing their historical development, importance, and clarifying their current status, mainly focusing on GAN and VAE.

Chapter 3, methodology, is focused on demonstrating the process, as well as the final results of the experiences performed to obtain the used datasets and models.

Afterwards, chapter 4, aims to demonstrates the results of this project, by using the built models to detect anomalies in the test set and display the findings of the research.

The final chapter serves as a culmination of the projects, summarising the key insights of this project and the limitations and hardships faced while suggesting ways of further developing the presented topics.

Chapter 2

Related Work

2.1 Anomaly

2.1.1 Definition of Anomaly

Anomalies have been researched since the 19^{th} century (M.A., 1887) and have had numerous ways to refer to them, such as outliers, exceptions, aberrations, surprises, peculiarities or discordant observations(Bhuyan et al., 2014).

Similarly to its nomenclature, different authors defined them differently as the concept evolved. The early literature refers to an anomaly as an observation that deviates from the remainder of the data (Grubbs, 1969; Hawkins, 1980; M.A., 1887). One of the most notable definitions of anomaly is ”an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism” by Hawkins (1980).

Current literature adds that anomalies are not only single cases or observations but can also be a group of observations that do not comply with the remainder of the data. (Barnett & Lewis, 1994; Foorthuis, 2021). Foorthuis (2021) defines anomalies as ”a case, or a group of cases, that in some way is unusual and does not fit the general patterns exhibited by the majority of the data”.

2.1.2 Types of Anomalies

According to the literature, there are three different types of anomalies Ahmed, Naser Mahmood, and Hu (2016); Bhuyan et al. (2014); Pang, Shen, Cao, and Hengel (2021).

- Point anomaly
- Contextual anomaly
- Collective anomaly

The various authors agree on the characteristics of these different anomalies. Bhuyan et al. (2014) presents the Table 2.1 which summarises each of them with examples from Pang et al. (2021).

Table 2.1: Types of Anomalies Bhuyan et al. (2014)

Types	Characteristics	Example
Point anomaly	An instance of individual data is anomalous concerning the rest of the data.	Abnormal health indicators of a patient.
Contextual anomaly	(i) A data instance found anomalous in a specific context. (ii) The structure of the data set induces context. (iii) Two sets of attributes are used for defining a context: (a) contextual and (b) behavioural attributes.	Rapid credit card transactions in unusual spatial contexts.
Collective anomaly	(i) A collection of related data instances found to be anomalous concerning the entire data set. (ii) Collection of events is an anomaly, but the individual events are not anomalies when they occur alone in the sequence.	Exceptionally dense sub-graphs formed by fake accounts in social networks are anomalies as a collection, but the individual nodes in those sub-graphs can be as normal as real accounts.

2.2 Anomaly Detection

2.2.1 Definition of Anomaly Detection

The literature defines anomaly detection as identifying the instances or patterns of data that deviate from the remaining data. This enables the study of rare and interesting data patterns, which can support various decisions and actions in various areas. (Ahmed et al., 2016; Bhuyan et al., 2014; Ma et al., 2021)

2.2.2 Problems, Challenges and Complexities

Although anomaly detection might seem simple at first, as its purpose is to identify instances that do not conform with the remainder of the data, the implementation of techniques, such as the ones discussed in 2.2.3, face several challenges as presented by Bhuyan et al. (2014); Pang et al. (2021).

In many cases, anomaly detection deals with the unknown. This means that due to the nature of the problem, anomalies are only known once they occur. An example is areas where new anomalies keep arising, such as networks where novel types of cyber attacks continue to happen.

In a given application of anomaly detection, several anomalies with different characteristics may occur.

As seen in Table 2.1, there are three types of anomalies which have to be taken into account.

Naturally, when comparing normal behaviour to anomalies, we are faced with an imbalance problem, as the amount of normal instances overwhelms the rare and few anomalies present. This translates into the close-to-impossible and very expensive task of obtaining an

extensive collection of correctly labelled data. Also, using imbalanced data to train models might translate into biased models.

The constant evolution of normal behaviours is a case of concept drift (Gama, Žliobaitundefined, Bifet, Pechenizkiy, & Bouchachia, 2014) in anomaly detection. Due to that evolution, previously useful and accurate models became less effective and unable to distinguish between normal and anomalous behaviour.

2.2.3 Anomaly Detection Techniques

There are several ways to address the anomaly detection problems. According to Bhuyan et al. (2014), the main approaches to anomaly detection problems are divided into classification, statistics, clustering and information theory. As will be shown in this brief review of the four approaches, the best one depends on each case, as it varies according to the nature of the data and the objective of each anomaly detection task.

Classification-based Anomaly Detection

Classification anomaly detection techniques operate in two phases: training and testing. This type of approach relies on the normal instances to create a baseline for what is normal behaviour. They operate under the assumption that a classifier that can distinguish between normal and anomalous classes can be learned in a given feature space.

This technique is very restrictive due to instances that deviate from what the trained model considers normal, being considered anomalies, which makes it capable of detecting new anomalies, but it inevitably leads to false positives.

Another problem that leads to a high false positive rate is that in several application areas, normal behaviour is continuously evolving, making it difficult to keep the models updated.

Classification techniques can be further distinguished into one-class classification, when instances have only one class label (normal or anomalous), and multi-class classification, when there are multiple normal classes and the classifier distinguishes between them.

Examples of classification-based anomaly detection techniques are Bayesian networks, support vector machines, and neural networks. (Bhuyan et al., 2014; Chandola, Banerjee, & Kumar, 2009)

Statistical Anomaly Detection

According toChandola et al. (2009), statistical anomaly detection techniques operate under the assumption that normal instances occur in high probabilistic regions of stochastic models, while anomalies occur in low probability regions.

Chandola et al. (2009) further distinguish statistical methods in parametric techniques, which assume knowledge of the distribution and estimate its parameters from the data, and nonparametric techniques, which do not assume knowledge of the distribution.

On the other hand, Bhuyan et al. (2014) exemplifies techniques that derive from the statistical tests, such as mixture models, which try to separate noise from anomalies by assuming that normal and anomalous behaviour follow two distinct distributions and principal component analysis, which can reduce the dimensions of data without losing much information and find anomalies by computing the distance of each instance to the mean of the principal components.

Chandola et al. (2009) approach anomaly detection using principal component analysis differently, as they refer to it as a spectral anomaly detection technique. This technique works under the assumption that by embedding data in a lower dimensional subspace, the differences between normal and anomalous data become more prominent.

Clustering

Clustering-based techniques rely on clustering algorithms to group similar instances of data. Although they are mostly unsupervised, as they do not require labelled data, they can also operate semi-supervised.

three different assumptions can be made when using clustering-based techniques:

- Normal data instances belong to clusters, whereas anomalous data does not; however, noise in the data is also considered anomalous data
- When a cluster has both normal and anomalous data instances, normal data instances are closer to the nearest cluster centroid, while anomalies are distant to the closest cluster centroid (using distance measures)
- When using clusters of varying sizes, anomalous data is found in the smaller and sparser clusters and normal data in the larger and denser clusters (using size or density thresholds)

(Bhuyan et al., 2014; Chandola et al., 2009)

Chandola et al. (2009) consider the existence of nearest neighbour-based anomaly detection techniques, which work very similarly to the clustering techniques.

Information Theory

Information theory-based techniques operate under the assumption that anomalies create irregularities in the information contents of the data. So, information theory measures such as entropy, Kolmogorov complexity and information gain are used.

Despite not making any assumptions about statistical distributions and working unsupervised, their performance is highly dependent on the chosen measure. It might only detect anomalies if they are enough. On top of that, it is very hard to associate anomaly scores with these measures. (Bhuyan et al., 2014; Chandola et al., 2009)

2.2.4 Evaluation Metrics

According to Ma et al. (2021), the most commonly used metrics to evaluate anomaly detection performance are accuracy, precision, recall rate, F1-score, AUC-ROC and the AUC-AP (Average Precision). Liu et al. (2022) use similar measures to evaluate their model's performance. However, they chose to use macro metrics, as those can mitigate the problem of data imbalance. These metrics are summarised in Table 2.2, where tp and tn are true positives and true negatives, fp and fn are false positives and false negatives and, for the macro metrics, the means anomalous class and n the normal class.

Different problems have different requirements, as seen with the example of some problems being more sensible to false positives and others to false negatives.

Table 2.2: Metrics Summary

Metric	Formula
Accuracy	$\frac{tp+tn}{tp+tn+fp+fn}$
Precision	$\frac{tp}{tp+fp}$
Recall	$\frac{tp}{tp+fn}$
F1-Score	$2 \times \frac{Recall \times Precision}{Recall+Precision}$
AUC-ROC	area under the ROC curve
AUC-AP	area under the precision-recall curve
Macro-Precision	$\frac{P_a+P_n}{2} = \frac{tp_a}{2(tp_a+fp_a)} + \frac{tp_n}{2(tp_n+fp_n)}$
Macro-Recall	$\frac{P_a+P_n}{2} = \frac{tp_a}{2(tp_a+fn_a)} + \frac{tp_n}{2(tp_n+fn_n)}$
Macro-F1-Score	$\frac{F1_a+F1_n}{2} = \frac{P_a \times R_a}{P_a+R_a} + \frac{P_n \times R_n}{P_n+R_n}$

2.2.5 Examples of Areas of Application

As mentioned, anomaly detection techniques have a wide range of application areas. This subsection contains examples of some of those areas of application. For each example, the importance of anomaly detection in that area, the approach used by the authors, and the metrics used to evaluate the results will be explored.

studies identifying fake news in social media citeFakeNews. Social media changed the production, dissemination, and consumption of news, creating new opportunities whilst creating problems such as fake news, which threatens the credibility of the whole news ecosystem. Reis, Correia, Murai, Veloso, and Benevenuto (2019) who approached the problem using textual features, such as syntax and subjectivity, and news source features, such as bias and credibility. Then, they applied several supervised learning classifiers, such as k-Nearest Neighbours, Naive Bayes, Random Forests, Support Vector Machine, and XGBoost. Finally, the ROC curve and the macro F1-score were used to measure the classifier's effectiveness.

The spammer problem explored by Miller, Dickinson, Deitrick, Hu, and Wang (2014). As social networking platforms expand, so does spam activity, for example, unsolicited messages with the purpose of advertising, phishing or spreading viruses, which directly influences the reputation of the platforms where they occur. Miller et al. (2014) approached this problem using modified versions of the stream clustering algorithms DenStream and StreamKM++. Their approach revolves around building a clustering model on normal users and treating the outliers as spammers. ROC curves and various measures, including recall, false positive rate, and F-Measure, were used to evaluate the performance of their modified algorithms.

Payment card fraud approached by Branco et al. (2020) is a severe problem that causes multibillion-dollar losses worldwide and often enables criminal activity. In this problem, payments are executed without the cardholder's consent and then have to be reimbursed by the banks or merchants as they are liable for accepting the fraudulent payments. The solution presented by Branco et al. (2020) is different from the regular rule-based systems with machine learning classification models, as they use a recurrent neural network framework to detect fraud in real time. Due to RNNs being cyclical deep learning architectures, they can capture patterns from sequences of events without the need to profile cardholders. The performance measures used were the false positive rate and recall of the framework and the system's efficiency regarding latency.

2.3 Graph Anomaly Detection

Anomaly detection in graph data has been an important task since the start of anomaly detection, (Ma et al., 2021) because data objects cannot always be treated as independent points in a multi-dimensional space. Akoglu, Tong, and Koutra (2014) identifies four different reasons for the need to represent data in graphs:

- Inter-dependent nature of the data: most data objects share relationships and dependencies. In reality, most relational data can be thought of as interdependent.
- Powerful representation: using edges to link data objects, graphs naturally represent connections and relationships between them and can capture long-range correlations.
- Relational nature of problem domains: the occurrence of an anomaly could be related to another.
- Robust machinery: graphs are adversarially robust tools, which make it harder for anomalies to disguise themselves as normal.

In their survey, Ma et al. (2021) state that the reason for using graphs is the rich relationships between objects that can provide complementary information, corroborating Akoglu et al. (2014).

2.3.1 Types of Graphs

According to Akoglu et al. (2014); Ma et al. (2021), anomaly detection graphs can be either static or dynamic and plain or attributed.

This distinction is also presented by Akoglu et al. (2014). The distinction between static and dynamic refers to their behaviour over time. While in static graphs, the structure and connections are fixed, in dynamic graphs, those are modified over time as nodes and connections are added or removed. So, static graphs can be considered snapshots of a given graph database, while dynamic graphs are a sequence of those snapshots.

The other distinction between plain or attributed graphs refers to the information beyond the graph's structure. While plain graphs, the nodes and edges do not have any additional information or attributes associated with them, attributed graphs include that type of data, focusing not only on the structure but also on the data objects and their properties.

2.3.2 Types of Anomalies

In graph anomaly detection, the anomalies differ from what was seen in 2.1.2 due to the graph structure in which the data is presented. In (Ma et al., 2021), graph anomaly detection focuses on detecting anomalous nodes/ vertices, edges and/or sub-graphs.

(Ma et al., 2021) approaches the type of anomalies in a general way and distinguishes them into four different types of anomalies for static graphs.

- Anomalous nodes: significantly different individual nodes, often representing anomalous objects that appear individually. The authors further distinguish them into global anomalies, which only consider the node's attributes, and structural anomalies, which only consider the graph's structural information. And community anomalies, which take both the node's attributes and graph structural information into account.

- Anomalous edges: anomalous links between data objects, which represent unexpected or unusual connections or relationships between objects

- Anomalous sub-graphs: refer to nodes that collude and behave collectively with others to garner benefits. They are subsets of nodes and edges within larger graphs that represent regions or communities that deviate from the expected structure of the graph they are contained in.

- Anomalous graphs: graphs that deviate from the remainder graphs of a database of graphs

In dynamic graphs, Ma et al. (2021) add to them the concept of evolving structure, patterns and attributes, which they identify as rich temporal signals. Although dynamic graphs face challenges due to their large volume of data compared to static graphs, some structures that might appear normal in several snapshots of a dynamic graph are anomalies when considering the changes between each snapshot.

On the other hand Ranshous et al. (2015),specifically focus on anomalies in dynamic networks. Apart from using different nomenclature, using vertex instead of nodes, they do not consider anomalous graphs but mention the existence of event and change detection, which are exclusively found in dynamic graphs.

In event detection, events are isolated points in time where the graph is anomalous when compared to itself in its previous and following states.

In change detection, a change point is a point in time where the entire behaviour of the graph changes. This change remains until the next change point.

2.3.3 Problems, Challenges and Complexities

In 2.2.2, challenges of anomaly detection were presented, which are also present in graph anomaly detection. On top of those challenges, specific graph anomaly detection challenges are identified in the literature by Akoglu et al. (2014); Ma et al. (2021).

- Inter-dependent and dynamic objects: Due to the relational nature of the data, objects in graph data have long-range correlations and can no longer be treated as individual objects. Understanding these relations is even harder when considering the dynamics of real-world data. Due to that, anomaly detection has to consider specific cases, such as spreading and guilt association.

- Variety of definitions: anomalies in graphs are even more diverse than the ones shown in 2.2.2, as anomalies may not only be present in the attributes or combination of attributes but also on the graph's substructures or even the graphs in a graph database. The hardships are further increased when considering the existence of different types of graphs.

- High dimensionality and scale: detecting complex anomalies in graph substructures is challenging due to the large search space. This challenge is further increased with the added complexity of attributed graphs. So, the algorithms have to consider effectiveness, efficiency and scalability.

- Scarce ground-truth: labelled ground-truth anomalies are often unavailable, as their identification by domain experts is very expensive.

- Unknown and camouflaged anomalies: there are still undiscovered anomalies. Furthermore, anomalies in some areas mimic normal objects' behaviour and patterns to avoid detection.

2.3.4 Graph Anomaly Detection Techniques

The literature has several different approaches to solving graph anomaly detection problems. According to their characteristics, the authors group them into different categories.

Akoglu et al. (2014) approaches methodologies for static plain and attributed graphs and plain dynamic graphs. In static attributed graphs, three types of methods are approached. These are structural-based methods, community-based methods and relational learning-based methods.

Structural-based methods are methods that exploit the rich structure of the graph to uncover anomalies. In plain graphs, this method is further separated into feature-based methods, which make use of the characteristics of the graph, such as centrality measures and degree distribution, to transform the graph anomaly detection problem into an outlier detection problem and proximity-based methods, which use graph structures to study the closeness of the graph's objects through their auto-correlation and under the assumption that the closer the objects are, the more likely it is that they belong to the same class. This technique tries to identify rare substructures for attributed graphs according to the object's connections and attributes.

Community-based methods, in plain graphs, try to identify objects that do not belong to a particular community or neighbourhood by studying densely connected groups and identifying links across different groups. In attributed graphs, this method aims to identify community outliers, objects with attributes that deviate from the remainder of the community.

Relation learning-based methods study the complex relationships between objects using network-based collective classification algorithms to determine whether a node is anomalous.

In dynamic plain graphs, the methods presented by Akoglu et al. (2014) are focused on events and methods are classified based on the type of event they can detect. The type of events explored by the author is feature-based, which focuses on the characteristics of the graph, similar to what was seen in static graphs. Decomposition-based, decomposes the states of the graph into a matrix or tensor and compares their properties. Community-based focuses on monitoring communities and identifying changes, and window-based uses a time window. Meanwhile, Ranshous et al. (2015), which mainly focuses on dynamic graphs, presents five different methods: community-based, for node, sub-graph and change detection. Compression-based for edge and change detection. Decomposition-based for node event and change detection, distance-based for edge sub-graph and event, and probabilistic-based methods for node, edge, sub-graph and event detection. Ranshous et al. (2015) agrees with Akoglu et al. (2014) the definition for both community and decomposition based methods. Compression-based methods assume that the graph's normal behaviour can be efficiently compressed, while anomalous behaviour results in a longer and more complex encoding. Distance measures-based methods follow the assumption that the closer objects are, the more similar they are, so distance measures are used to compare the summary metrics for the different states of the graph and evaluate the similarity or difference of its objects. Probabilistic model-based

methods create a profile for normal behaviour according to probabilistic theory, distributions or scan statistics, such as the window-based events explored by Akoglu et al. (2014) and deviations from that profile being considered anomalies. Pourhabibi, Ong, Kam, and Boo (2020) Separates the methods in community, probabilistic, structural, compression and decomposition.

2.4 Deep Learning for Graph Anomaly Detection

Recently, graph anomaly detection problems have been tackled using deep learning methodologies, as these techniques cannot only detect potential anomalies more accurately but they can also do so in real time.(Ma et al., 2021)

Although deep learning techniques bring their set of challenges and complexities to the anomaly detection problem, they can also deal with several of the challenges shown in previous sections of this study. Pang et al. (2021) believe that deep learning can play a lead role in solving problems. Those are the low true positive rate caused by the rarity and heterogeneous nature of the anomalies, high dimensionality, relationships between data objects and complex anomalies, the inefficiency due to the costs of obtaining labelled data for supervised learning, the assumptions made during unsupervised learning and the semi-supervised methods being vulnerable to noisy data.

2.4.1 Problems, Challenges and Complexities

In their survey, Ma et al. (2021) identified some challenges of implementing deep learning in graph anomaly detection. While some of the problems identified by the authors stem from deep learning, others result from previously identified challenges. Due to the scarcity and costs of ground truth, designing deep learning methods' training objectives to tune their parameters is hard. Furthermore, the more complex are very dependent on those objectives, which leads to high costs in both time and resources. Another problem emerges from the need for labelled data for hyperparameter tuning, to which deep learning models are highly sensitive. The last problem tackled by the authors is interpretability, as evidence and justifications are needed to support decisions related to anomalies.

2.4.2 Deep Learning Methods

Ma et al. (2021) who focus on deep-learning techniques for graph anomaly detection, separate traditional methods from deep-learning methods and go over techniques that can be used to identify anomalies in the different substructures and graph databases according to the type of graph. The deep learning techniques are separated into seven according to the type of neural network they use, except for reinforcement learning.

- Deep Neural Networks
- Graph Convolutional Neural Networks
- Network Representation
- Graph Neural Network

- Reinforcement Learning
- Generative Adversarial Neural Networks
- Graph Attention Networks

On the other hand, Pang et al. (2021) propose a taxonomy for deep anomaly detection, separating methods into three different groups: deep learning for feature extraction, learning feature representations of normality and end-to-end anomaly score learning.

Deep learning for feature extraction methods can learn low-dimensional feature representations from the graphs and follow the assumption that the extracted feature representations maintain key information that distinguishes between normal and anomalous. This type of method that falls under this category only applies deep learning to perform the dimensionality reduction task.

Learning feature representations of normality distinguishes from the previous group techniques by combining feature learning with anomaly scoring. These are further separated into generic normality feature learning and anomaly-measure-dependent feature learning. These two are then divided according to how they formulate their objective function. Generic normality feature learning methods are not specifically created for anomaly detection. This class of methods learns the representations of data instances by optimising a general feature learning objective function. Nevertheless, the learned representations can still support anomaly detection as they are bound to capture some important underlying data patterns. These methods include autoencoders, generative adversarial networks, predictability modelling and self-supervised classification. Anomaly measure-dependent feature learning learn feature representations that are optimised for a certain existing anomaly measure. These methods include distance-based measures, one-class classification-based measures and clustering-based measures.

Lastly, the end-to-end anomaly score learning automates the entire anomaly detection process while not relying on existing anomaly measures, as it uses a neural network to quantify the anomaly scores. This method can use ranking, prior-driven, softmax likelihood models and end-to-end one-class classification according to how anomalies are scored.

2.4.3 Variational Autoencoders vs. Generative Adversarial Networks

As the previous section shows, variational autoencoders and generative adversarial networks fall under the generic normality feature learning methods. In recent years, this model type has been rising in importance and popularity as they are flexible, unsupervised learning algorithms capable of modelling complex distributions of high-dimensional data.

Variational Autoencoders follow a similar architecture to other deep autoencoders. They are composed of both an encoder and a decoder, as seen in Figure 2.1.

The difference between VAE and traditional autoencoders is that the latent variable distributions are constrained. This means that VAEs learn the probability density function of training data. So, contrary to the deterministic traditional deep autoencoders, VAEs are probabilistic. (González-Muñiz, Díaz, Cuadrado, García-Pérez, & Pérez, 2022a) For anomaly detection, the encoder transforms the input data into a low-dimensional representation, known as the latent variables. At the same time, the decoder attempts to accurately restore the low-dimensional representation to its original state by creating new and synthetic data samples.(Sun, Wang, Xiong, & Shao, 2018) The anomaly score is then calculated according to the

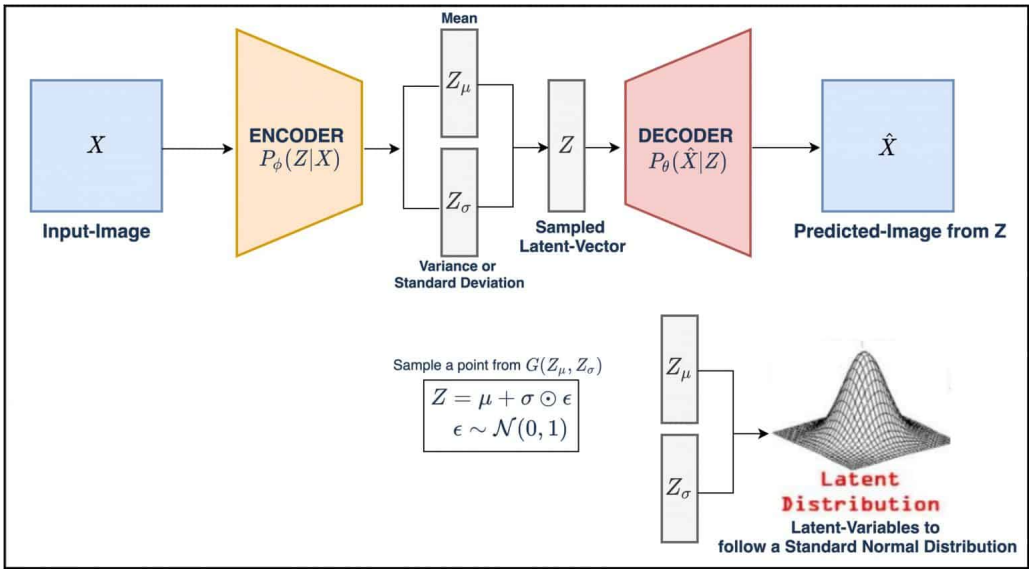

Figure 2.1: Variational Autoencoder architecture

reconstruction error, the difference between the real data and the synthetic data created by the decoder. According to Pang et al. (2021), autoencoder models follow the assumption that anomalies cannot be as effectively reformed from compressed space as normal instances.

Generative Adversarial Networks, first introduced by Goodfellow et al. (2014), have drawn much interest in the anomaly detection problem because of their outstanding abilities to generate simulated data and accurately capture the distribution of real data Ma et al. (2021).

GANs, represented by Figure 2.2 is a framework that trains two different models simultaneously. The generative model learns the distribution of the training data and generates synthetic data samples, while the discriminative model attempts to distinguish between the real and synthetic data. So, the generative model is trained to be able to deceive the discriminative model, while the discriminative model's training objective is to discern between the generated data and real data correctly.

According to Pang et al. (2021) for anomaly detection, GAN-based models assume that the generative model can more effectively generate normal data than anomalous data. So, generated data is compared to the real data using error measures to determine the anomaly score.

There are several variations of GANs as researchers attempt to develop these algorithms to improve their stability and performance. Some examples are the Wasserstein GAN (WGAN) and Deep Convolutional GAN (DCGAN).

WGAN is a type of GAN that is different from the original by Goodfellow et al. (2014), which, instead of the Jensen–Shannon divergence, uses the Wasserstein distance (Arjovsky, Chintala, & Bottou, 2017), which stabilises the training procedure. The importance of stabilising the training process comes from the traditional GAN weakness of failure to converge and mode collapse.

DCGAN are a type of GAN introduced by (Radford, Metz, & Chintala, 2015). The main difference between these two types of GAN is the use of deep convolutional networks in the architecture of the generator and discriminator, which allows the processing of high-

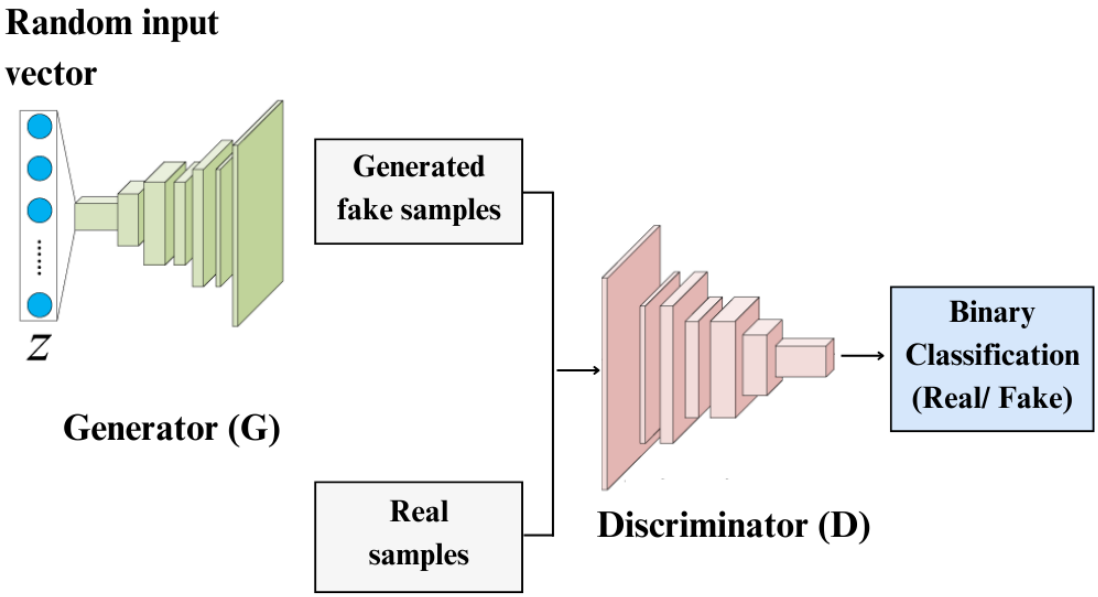

Figure 2.2: Generative adversarial networks architecture

dimensional data instead of fully connected neural networks, which are more appropriate for low-dimensional data.

2.4.4 Examples of Application of VAEs and GANs in Anomaly Detection

VAE and GAN-based models have already been applied to anomaly detection problems. In this section, a review of some of their applications will be presented as a means to show their utility, versatility, importance and popularity in anomaly detection problems.

In the healthcare area, Cozzatti, Simonetta, and Ntalampiras (2022) proposed a weakly-supervised approach using a convolutional variational autoencoder to detect respiratory diseases using respiratory patterns. The encoder generates a mean and variance for each sample in their model, creating the matching latent Gaussian distribution. A single point is sampled from the expected distribution and given to the decoder, which reconstructs the instance. The anomaly detection occurs by calculating the mean square error between the reconstructed and real data.

Xu et al. (2018) proposed the algorithm Donut, a VAE-based unsupervised anomaly detection algorithm for web applications Key Performance Indicators (KPIs). To train their model, evidence was modified with lower bound and missing data injection techniques. In detection, they used Markov chain Monte Carlo missing value imputation.

González-Muñiz, Díaz, Cuadrado, García-Pérez, and Pérez (2022b) applied variational autoencoders to detect anomalies in engineering systems, such as hydraulic systems. So, the variational autoencoder was trained to reconstruct healthy behaviour data of the system based on its underlying probability density function. The residuals obtained through the reconstruction error are then used in a component-wise classification algorithm to classify a sample as normal or anomalous.

Schlegl, Seeböck, Waldstein, Langs, and Schmidt-Erfurth (2019)developed f-AnoGAN,

an unsupervised learning model based on GAN, to detect image anomalies. Through the exclusive use of healthy/ normal data to train WGAN and an encoder that maps images to the latent space. The discriminator of the WGAN is also used to train the encoder. The encoder and the generator reconstruct the image for the anomaly detection process. Then, a combination of the reconstruction error and the residual connections of the discriminator is obtained. The authors demonstrated that the images created by the generator of the model were indistinguishable from real.

Zheng, Yuan, Wu, Li, and Lu (2018) developed One-Class Adversarial Nets (OCAN), a model composed of a long short-term memory network (LSTM) autoencoder and a complementary GAN, using only normal data. OCAN works by first using the LSTM-autoencoder to learn a low-dimensional representation of the benign users. Afterwards, the complementary GAN is trained so that the generator can create data points in the sparse area of benign users, and the discriminator distinguishes between the real and synthetic data. The discriminator can distinguish between benign and malicious users, as malicious users will be placed outside the learned region of benign users. After training, the model can update the user's representation as he performs new actions.

2.5 Comparative Analysis and Exploratory Focus

According to the reviewed literature, most of the research on the field of anomaly detection, focuses on developing state-of-the-art models, that aim to outperform their predecessors. This is also true to the more concrete field of graph anomaly detection.

The research performed diverges from the path followed by most authors, as it aims to conduct a comparative analysis between GAN and VAE-based models, exploring the nuances and difficulties of building, training and evaluating these models. Additionally, the difficulties of obtaining and exploring and working with graph data are explored.

Chapter 3

Methodology

3.1 Data Collection and Visualisation

3.1.1 Data Collection

This project aims to compare two different types of neural networks for anomaly detection. That means more than one dataset is required to compare the performance of the GAN and VAE models.

Unfortunately, as mentioned in the literature, it is a challenge to find datasets aimed at graph anomaly detection with ground-truth available to train and evaluate models. So, to circumvent this problem, two binary-classification graph datasets from the website "Network Data Repository"(Rossi & Ahmed, 2015) were found. Afterwards, the appropriate transformations were applied so that the datasets showed the charactersitics of anomaly detection datasets.

The two datasets used in this project have the same structure: a dataset composed of plain and static subgraphs, and the objective is to identify the subgraphs of the minority class correctly.

The original IMDb dataset is a collection comprising 1000 distinct, heterogeneous, plain, and static graphs. Within this dataset, there is a total of 19,773 individual nodes interconnected by 386,124 edges. Within this dataset, 500 of these graphs represent data associated with action movies, while the remaining 500 are dedicated to depicting information about romance movies.

The original Reddit dataset consists of a diverse set of 2000 distinct, static, and plain graphs. This expansive dataset boasts 859,254 nodes linked together by 3,982,032 edges. This dataset comprises 1000 graphs specifically focused on questions and answers, while the remaining 1000 graphs represent discussion forums.

3.1.2 Data Visualisation

The datasets are composed of three files each, which give information on the edges in the format [source, target], the sub-graph the edges belong to and the label of the sub-graph.

To aid in the visualisation of these graphs, NetworkX was used. The raw data can be efficiently transformed into NetworkX's graph objects and then presented for visualisation thanks to its integration with other libraries.

Examples of the plotted graphs can be seen in figures 3.1 and 3.2 for the IMDb and Reddit datasets, respectively.

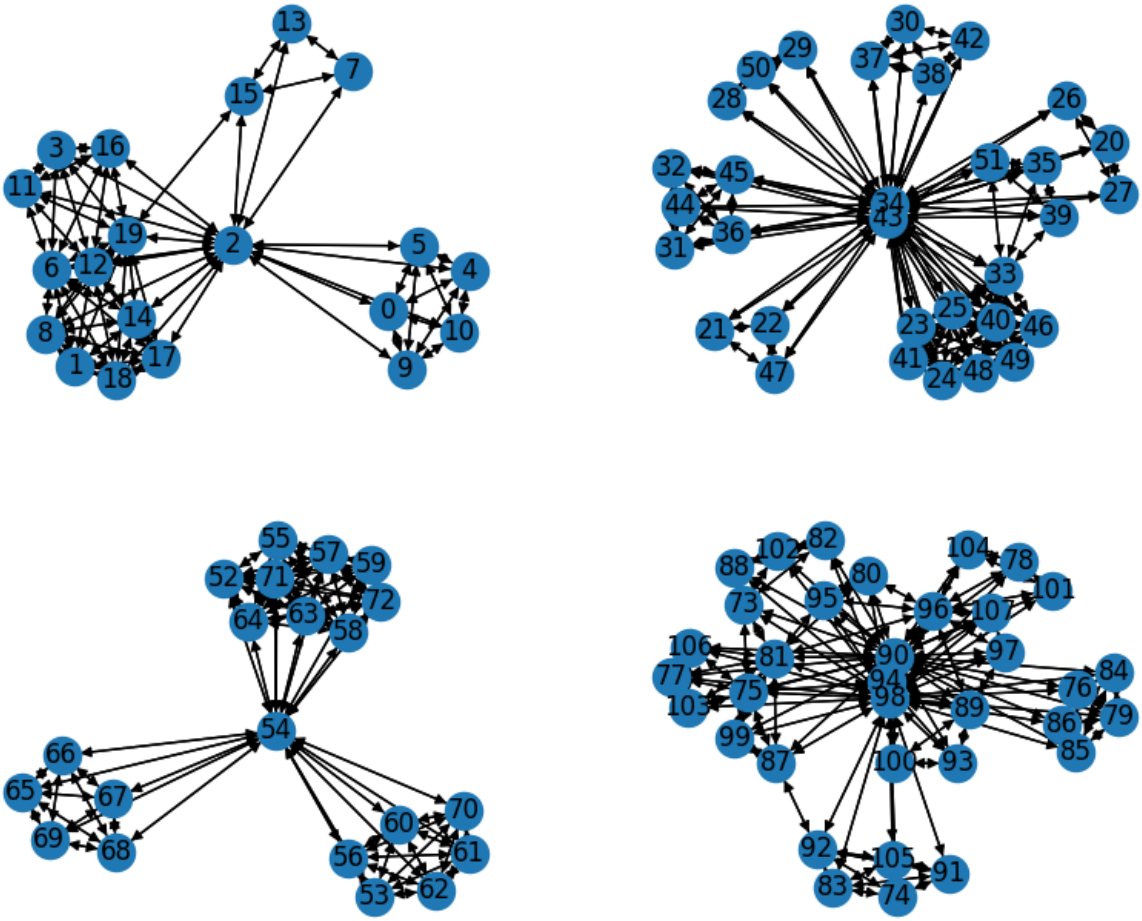

Figure 3.1: IMDb - Examples of Graphs

3.2 Data Pre-Processing

3.2.1 Data Pre-Processing Objective

Data pre-processing is crucial since it forms the basis for preparing the datasets for the modelling and evaluation steps.

The framework for developing this project is TensorFlow therefore, the data was processed so that a TensorFlow dataset object is constructed.

The TensorFlow dataset is a standardised data structure that ensures consistency, compatibility, and integration with the whole TensorFlow ecosystem. Furthermore, it enables the caching, shuffling, batching, mapping, shuffling and data splitting to create a train, validation, and test set.

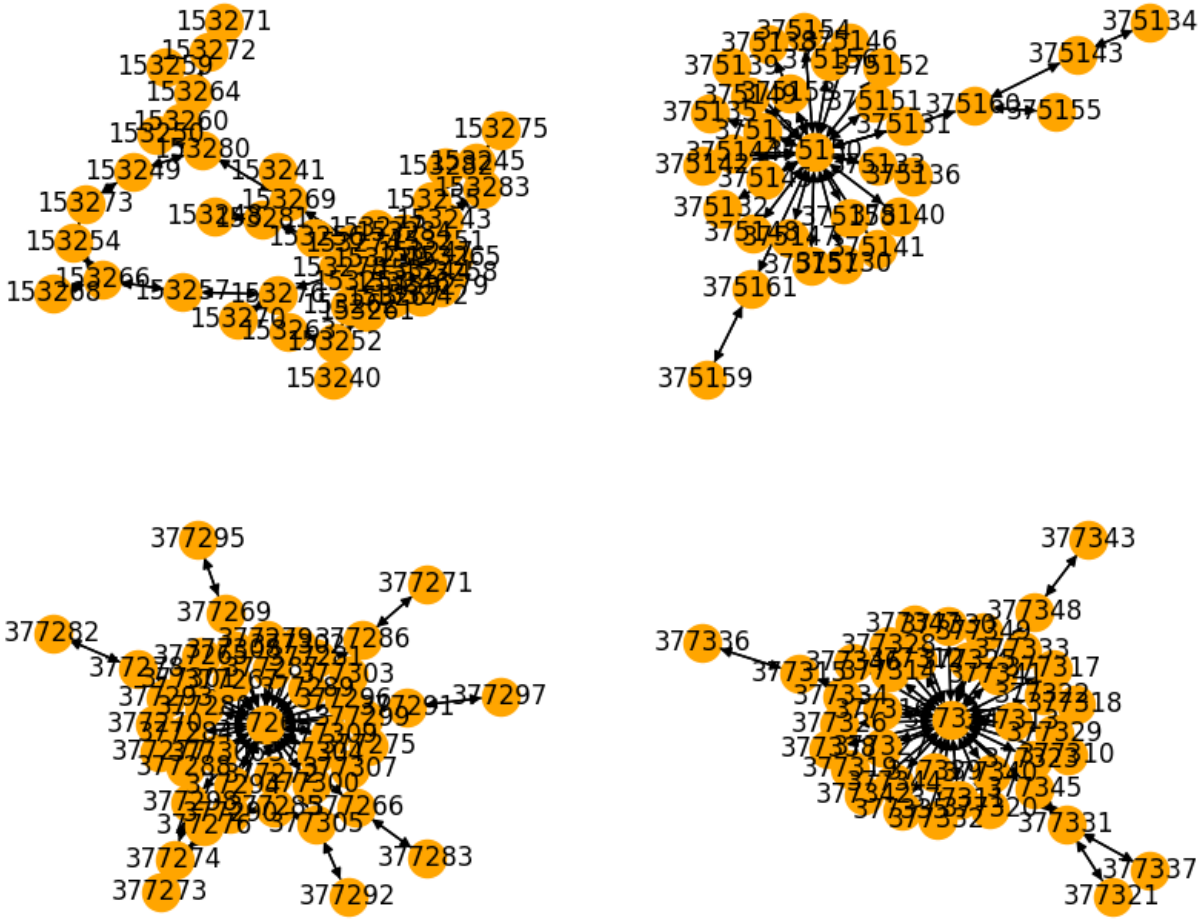

Figure 3.2: Reddit - Examples of Graphs

3.2.2 Data Pre-Processing Pipeline

As mentioned previously, the two graph datasets are separated into three files. The first file contains the edges in the format [source node , target node], the other has the indicator of the subgraph for each edge. And finally, the labels for normal (0) or anomalous (1) for each subgraph.

The data was initially loaded as a Pandas DataFrame. However, to enhance computational efficiency, it underwent some transformations. Those were ensuring that edges and subgraph indicators start at 0 and transforming the Pandas DataFrames into Numpy arrays, guaranteeing ease of iteration over the data.

With the data organised as NumPy arrays, constructing NetworkX graphs was the next step in the pipeline by iterating over the arrays. The graphs were then stored in a list, which streamlined the graph-related tasks, such as visualisation and obtaining adjacency matrices.

As the project was developed with limited computational resources and adjacency matrices are required for the following steps, the next step was to filter out graphs containing more than 200 nodes. This guarantees a balance between the size of the final TensorFlow dataset and the computational resources needed to process the adjacency matrices.

Afterwards, the adjacency matrices of the graphs are obtained as arrays by iterating over

the list of filtered graphs. Then, padding was performed to ensure uniformity and compatibility with the following steps so that the adjacency matrix of graphs with fewer nodes matched the dimensions of the one from the largest graph in the dataset, which is a crucial requirement to utilise the TensorFlow dataset. The adjacency matrices' final shape in the IMDb dataset is [136,136], while in the Reddit dataset, the final shape is [198,198].

Finally, the TensorFlow API is leveraged to build the dataset. This dataset is then processed to build the training, validation and test sets according to the problem's requirements.

A one-class learning approach was selected for the development of the models, therefore separating normal data and anomalous data in two different objects is required. Afterwards, for the VAE model, a 70-15-15 split in the normal dataset is performed for training, validation and test sets. While for the GAN a 80-20 split is performed for train and test sets. Subsequently, anomalies from the anomalous dataset are injected into the test set of both models, ensuring that the test set is composed of 90% normal data and 10% anomalous data.

Having obtained the dataset and performed the split according to the model to be used, the final phase of data preparation focused on optimising the data input pipeline. This is done by caching it to enhance data access speed during training. Perform random shuffling to prevent the model from learning sequential patterns in the data. Mapping the data to use only the adjacency matrix, not the label, as we will perform one-class learning. Batching for more frequent weight updates and easier convergence of the models during training. Repetition to refine the models using different data patterns and prefetching to maintain a continuous data flow during training and minimise idle time.

3.3 Model Development

3.3.1 Variational Autoencoder

The VAE is composed by a encoder, which has a sampling layer to introduce stochasticity to the model, and a decoder.

Encoder

The encoder has the objective of mapping the adjacency matrices to the latent space.

The encoder is built starting with an input layer, which receives adjacency matrices.

Following the input layer a flattening layer is applied to transform the matrix into a one-dimensional vector, which enables the use of dense layers.

The encoder consists of three dense layers with 256, 128 and 32 neurons and are activated using the Leaky ReLU activation function, to reduce dimensionality and capture the essential features of the data.

The sampling layer follows the dense layers. It involves two other dense layers to compute the deterministic variables mean and logarithm of variance of the latent space. The sampling layer takes these parameters and generates a sample using a lambda layer to apply the $z = \mu + e^{\log \sigma} \cdot \epsilon$, where ϵ is sampled from a standard Normal distribution. Incorporating this reparametrisation trick, gradient-based backpropagation through the deterministic components while separating the deterministic and stochastic parts of the model. The dimension of the latent space used was 8.

Decoder

The decoder takes the sampled points from the latent space and reconstructs the original adjacency matrix.

The decoder starts by passing the sampled point through three dense layers with 32,128 and 256 neurons to expand the latent space back to a higher-dimension.

Afterwards, a final dense layer with sigmoid activation and size equal to the encoder's flattening layer to then finish the reconstruction with a reshape layer, which restores the original shape of the adjacency matrices.

Loss Function

The loss function of the VAE has two different components, the reconstruction loss and the KLdivergence loss.

The first component the reconstruction loss uses binary cross-entropy, which is indicated from binary data. In this specific case the difference is scaled by a factor of 20 to amplify the importance of reconstruction accuracy.

$$\text{Binary Cross-Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \cdot \log(p_i) + (1 - y_i) \cdot \log(1 - p_i)\right) \tag{3.1}$$

N represents the total number of samples or data points.

y_i is the true binary label for the i-th sample.

p_i is the predicted probability that the i-th sample belongs to one of the classes.

The second component introduces the regularisation of the latent space. The KL divergence is used to verify the difference between the latent variables and a standard Normal distribution. The KL divergence term is multiplied by -0.5 to encourage the latent space to follow a standard Gaussian distribution.

$$\text{KL Divergence} = -0.5 \sum_{i} \left(1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2\right) \tag{3.2}$$

σ_i^2represents the variance of the latent variable i).

$\mu_i$represents the mean of the latent variable i).

The overall loss is the mean of these two measures. The objective of applying the mean is to balance the objectives of accurate reconstruction and regularisation of the latent space.

Training Procedure

A crucial step in the VAE's development is its training phase, which intended to ensure the model's convergence, minimising the loss function, while avoiding overfitting. To that end a validation set was used and the loss on the validation set monitored.

The Adam optimisation algorithm was chosen, as it demonstrated good performances during training.

The hyperparameters to be defined in this VAE model are the batch size, learning rate and latent space dimension. The learning rate of 0.00001 and the latent space dimension of 8 were defined for both datasets, as they demonstrated a balance between time to converge and stability based on empirical testing. The batch size, determined in the same way as the

previous hyperparameters is 32 in the IMDb dataset, while a batch size of 64 showed better performance in the Reddit dataset.

Initially several callbacks were employed to influence and monitor the training process.

The used callbacks were model checkpoint to ensure the best-performing model, that is the one with the lowest validation loss, was saved during training, the learning rate scheduler, to adjust the learning rate based on the epoch number and the early stopping callback, which ensured that when the model converged, training would stop.

In the final version of the model, only early stopping was used, as it seemed to be the most effective strategy. The early stopping callback analysed the loss function evolution on the validation set and once the model trained for 50 epochs without improving it stopped the training process and restored the model's weights to the epoch where the performance on the validations was best. This guaranteed that when trainning VAE overfitting was avoided and the training time smaller.

VAE Training Outputs

In the IMDb dataset, it took 887.46 seconds to finish training. The best weights were found in the epoch 57 and the model had a overall loss of 0.5671 on the training set and a overall loss of 0.3728 on the validation set. The evolution of the loss function can be seen in figure 3.3

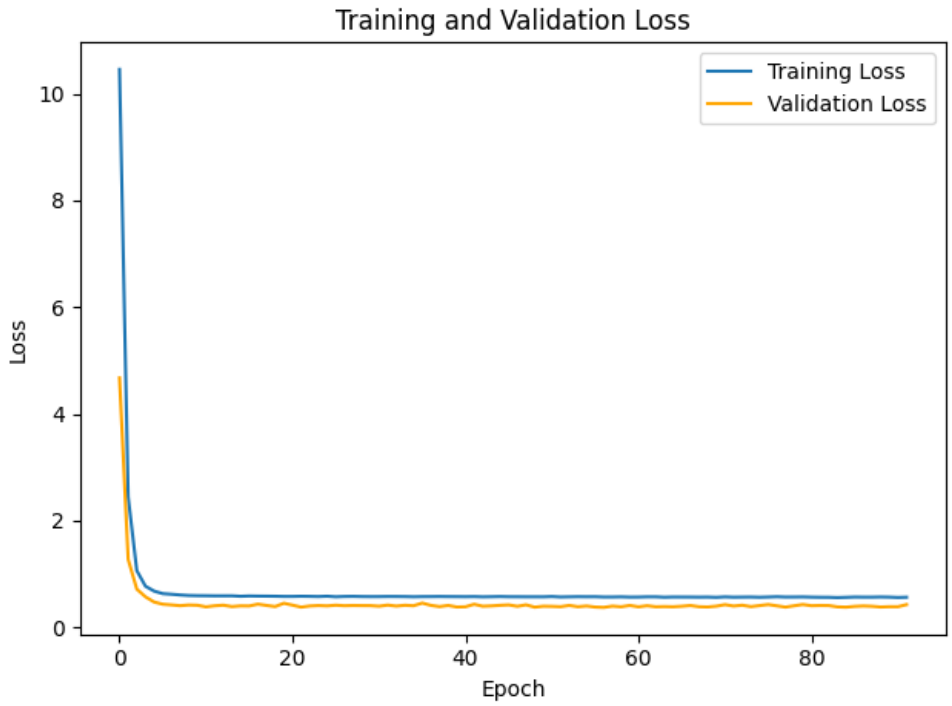

Figure 3.3: Evolution of the Loss Funtion in VAE training IMDb

In the Reddit dataset, it took 2426.1 seconds to finish training. The best weights were found in the epoch 211 and the model had a overall loss of 0.4977 on the training set and a overall loss of 0.0.4476 on the validation set.The evolution of the loss function can be seen in figure 3.4

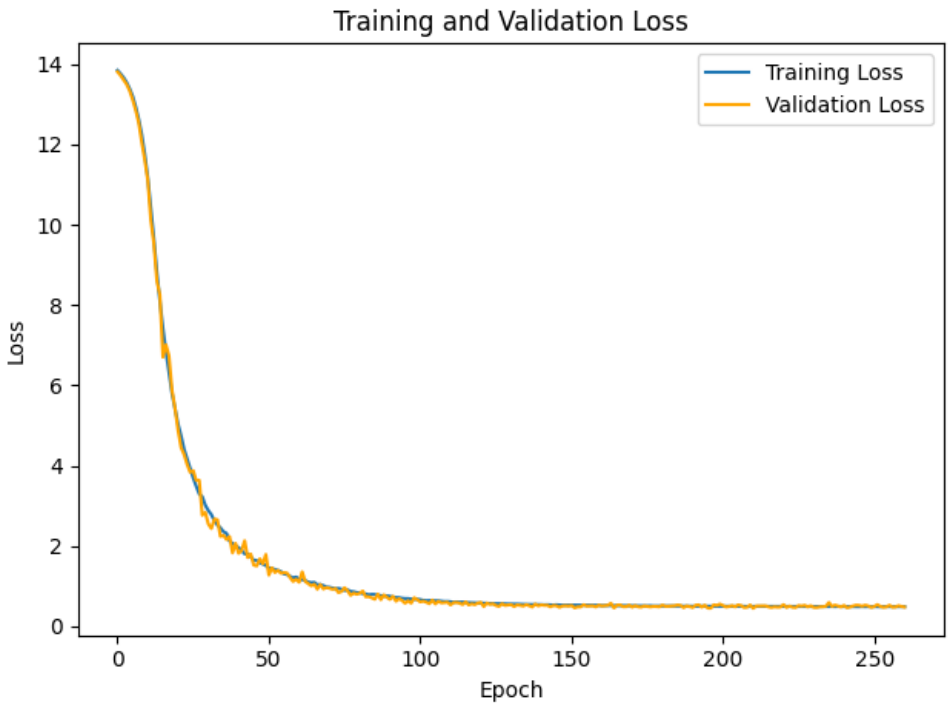

Figure 3.4: Evolution of the Loss Funtion in VAE training Reddit

3.3.2 Generative Adversarial Network

The GAN model is composed by a generator and a discriminator that are trained in an adversarial way.

Generator

The generator starts with a input layers that accepts a random noise.

The random noise is then passed through three dense layers with 32,64 and 128 are able to progressively transform the noise into a higher-level representations, while the Leaky ReLU allows for the capture of complex patterns in the data.

To prevent overfitting and improve the model's capabilities for generalisation, dropout layers with a rate of 0.3 are interpolated with the dense layers.

The final layer before generating a adjacency matrix is a dense function with sigmoid activation function. The size of this layer is defined by the number of nodes to the power of 2.

Finally a reshape layer that transforms the output of the previous layer into a adjacency matrix.

Discriminator

The discriminator starts with an input layer that accepts either generated graphs or real graphs.

The graph is then passed to a flattening layer for further processing.

The following step is to pass the data through three dense layers with activation Leaky

ReLU and 128, 64 and 32 neurons respectively, progressively learning representations to distinguish between real and generated inputs.

Similarly to the generator, dropout layers with a rate of 0.3 are interpolated with the dense layers to avoid overfitting.

The final layer is a dense layer with sigmoid activation function with a single neuron to output a single value, which determines if the discriminator thinks the input it is analysing is real data (0) or generated data (1).

Loss Function

The loss function chosen for both the generator and discriminator is binary cross-entropy, as this measure is indicated for binary classification problems.

The interaction for the GAN's loss measures the dissimilarity between the discriminator prediction and the ground truth label, if the discriminator guesses correctly, it is rewarded and the generator penalised, if it guesses incorrectly, the discriminator is penalised while the generator is rewarded.

Training Procedure

The GAN involves simultaneous training of two different models in adversarial manner, so a custom training loop was developed.

The training process starts by loading real graph data. Afterwards, fake data is generated using the generator. The data generation process is started by sampling a random uniform distribution, parameterised by the hyperparameters batch size and noise input dimension.

Discriminator Training

To train the discriminator, both real and generated data are passed to the discriminator, which makes prediction, attempting to classify real data as 0 and generated data as 1.

Afterwards, the predictions of the discriminator are injected with noise to strengthen the adversarial dynamics of the model. This takes into account that real data should be classified as 0 and generated data as 1. Through the use of a random uniform distribution, a perturbation of -0.15 is applied to generated samples, while a perturbation of +0.15 is added to the real ones. The addition of noise in the output aims to diversify the learning process and improve the model's robustness.

To quantify the performance of the discriminator, binary cross-entropy loss is applied. This measure can quantify the difference between predicted and actual labels, making it a powerful tool to guide the model's training.

To finish the discriminator's training, with the loss function computed, the process of backpropagation is initiated. The gradient is computed through differentiation, giving the insight of how the model's parameter change influenced its loss. After this, the gradient is applied to the optimiser, which uses this information to update the model. This continuous adaptation develops the model's capabilities to distinguish between real and generated data, which increases the difficulty the generator's task of tricking it.

Generator Training

As the discriminator increases its capabilities, so does the generator. The generator training process starts once again by generating data, with the same hyperparameters as the discriminator fake data.

The generated data is then passed to the discriminator, which, once again, attempts to distinguish between generated and real data samples.

The generator's loss, which is also binary cross-entropy is computed afterwards. However, the model is rewarded when it successfully tricks the discriminator and penalised when it fails to do so. This guarantees that the generator strives for generating data that is indistinguishable from real data.

Then , similarly to what occurs in the discriminator training, the backpropagation process starts to update the model and guide the optimiser in its learning process. The continuous adaptation develops the model's capabilities to generate data indistinguishable from the real data increasing the difficulty of the discriminator's task of correctly predicting.

Optimiser and Hyperparameters

Simialrly to the VAE training, the Adam optimisation algorithm was chosen, for both generator and discriminator, as it demonstrated good performances during training.

The hyperparameters to be defined in this GAN model are the batch size, learning rate from the generator and discriminator, the dimensions of the generator's noise input and the number of epochs for training. The generator's noise input, learning rates, and number of epochs for the model's were the same in both IMDb dataset and Reddit dataset. The dimension of the noise input was defined at 16, the generator's learning rate is 0.0001 and the discriminator's learning rate is 0.000001. The batch size used was 32 for the IMDb dataset and 64 for the Reddit dataset. While the IMDb dataset was trained over 350 epochs, the Reddit dataset required 500 epochs to complete training.

The adversarial training dynamic of the model's custom training loop and custom gradient update is crucial part of the model, but presents issues in the implementation of conventional callbacks. Some examples are the dynamic between generator and discriminator not aligning with the sequential nature of callbacks, and callbacks being focused on following the training of a single model. As a result these tools were not employed in the training procedure of the GAN model.

Training Outputs

In the IMDb dataset, it took 2611.75 seconds to finish training over 350 epochs. The final loss for the generator was 0.1.1868, and for the discriminator 0.4206. The evolution of the loss functions can be seen in figure 3.5

In the Reddit dataset, it took 3172.33 seconds to finish training over 500 epochs. The final loss for the generator was 1.7511, and for the discriminator 0.4209. The evolution of the loss functions can be seen in figure 3.6

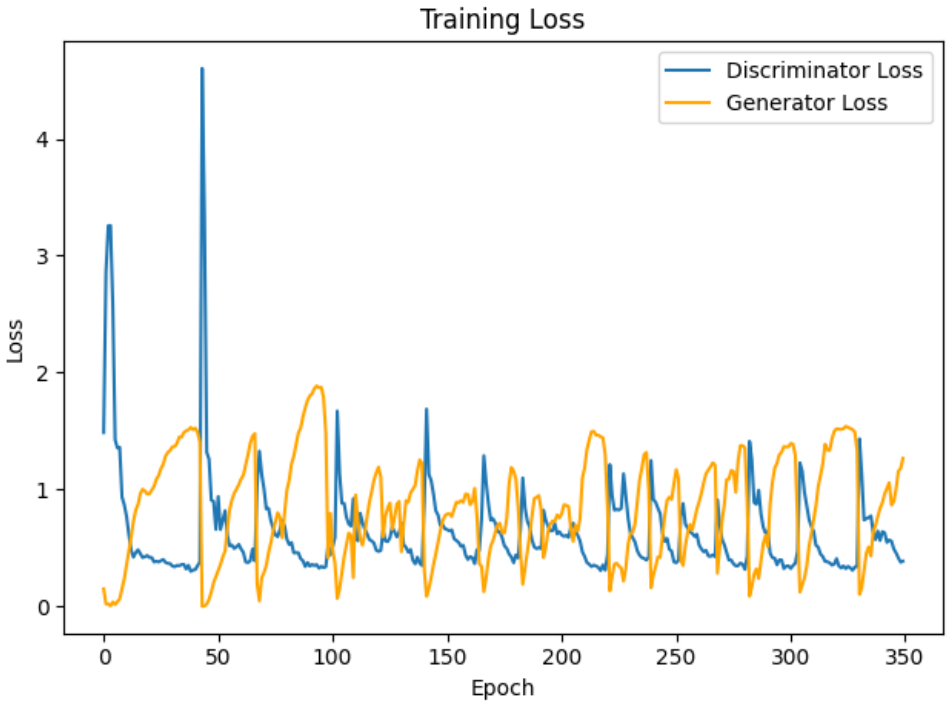

Figure 3.5: Evolution of the Loss Funtion in GAN training IMDb

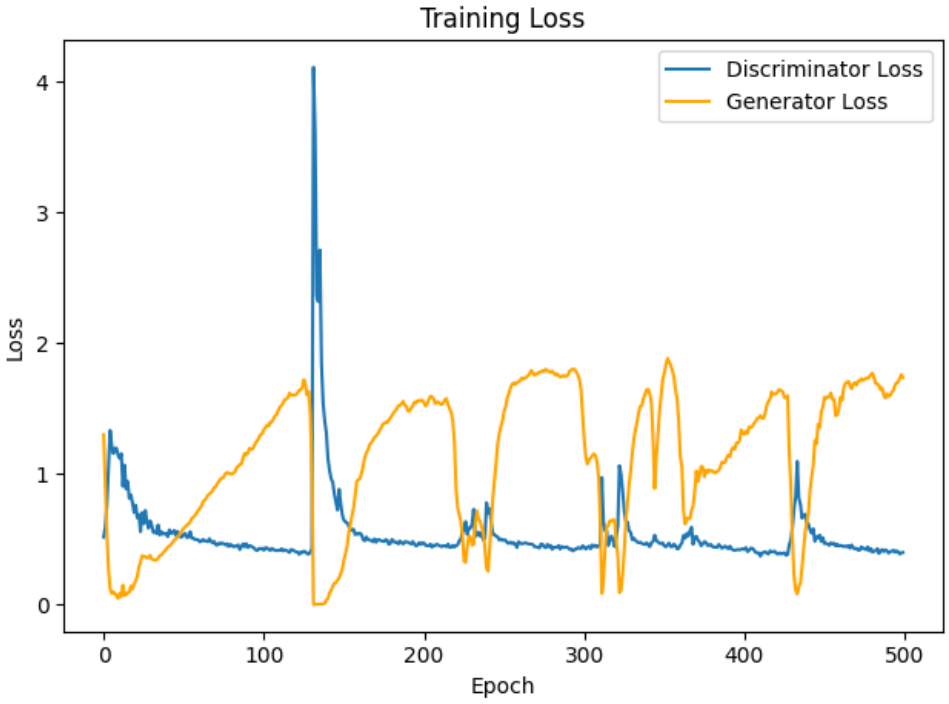

Figure 3.6: Evolution of the Loss Funtion in GAN training Reddit

3.4 Training, Evaluation and Tuning

The architecture and hyperparameters were systematically altered to enhance performance through an iterative process of fine-tuning during the construction of both VAE and GAN models. This strategy used a trial-and-error methodology to ensure the models were appropriate for detecting graph anomalies while keeping them as straightforward as possible. To this end, several combinations of layers, activation functions and number of neurons for the architecture where experimented with. While for the hyperparameters, the experiments involved learning rates, batch sizes, input for the GAN's generator, number of epochs for GAN training and latent dimension size for the VAE model.

Training a GAN involves a custom training loop as the generator and discriminator's training relies on adversarial training dynamics so, neither callbacks nor a validation set of data was used.

Contrary to the GAN's training, the VAE's training involved using a callback for early stopping and a validation set of data, which worked together to avoid overfitting the data and reduce the model's training time.

Finally, to evaluate the models, ROC and Precision-Recall curves offer comprehensive insight into the models' performance and support the thresholding process to determine whether instances are anomalous or normal.

Chapter 4

Model Testing and Evaluation

4.1 Research Objective

The project's primary objective is to compare GAN and VAE-based models for anomaly detection in graph data. The comparison that is proposed goes beyond the performance of the models, as it aims to compare the models' development, training and testing processes, identifying their strengths, weaknesses and the challenges inherent to them.

In addition to the primary objective, this project aims at uncovering challenges of dealing with well structured and high-dimensional graph data and the anomaly detection process in it.

4.2 VAE Anomaly Detection

The VAE model is trained to reconstruct normal instances in the training phase. This signifies that the model understands normal graphs' underlying patterns and structures.

In the test phase, the test data is passed to the model, and it tries to map it into the latent space and then reconstruct it back to its original shape and then it outputs the reconstruction loss. Since the model has accurately learned to reconstruct normal data, it is expected that when it attempts to reconstruct an anomaly, the reconstruction loss will be much larger than that of a normal instance. So, a threshold is applied to the reconstruction loss and instances above that threshold are classified as anomalies. This threshold is chosen by maximising the difference between the true and false positive rates.

IMDb Dataset Metrics

- Precision = 0.29
- Recall = 0.75
- Accuracy = 0.80
- F1-Score = 0.41
- ROC-AUC = 0.77
- PR-AUC = 0.21

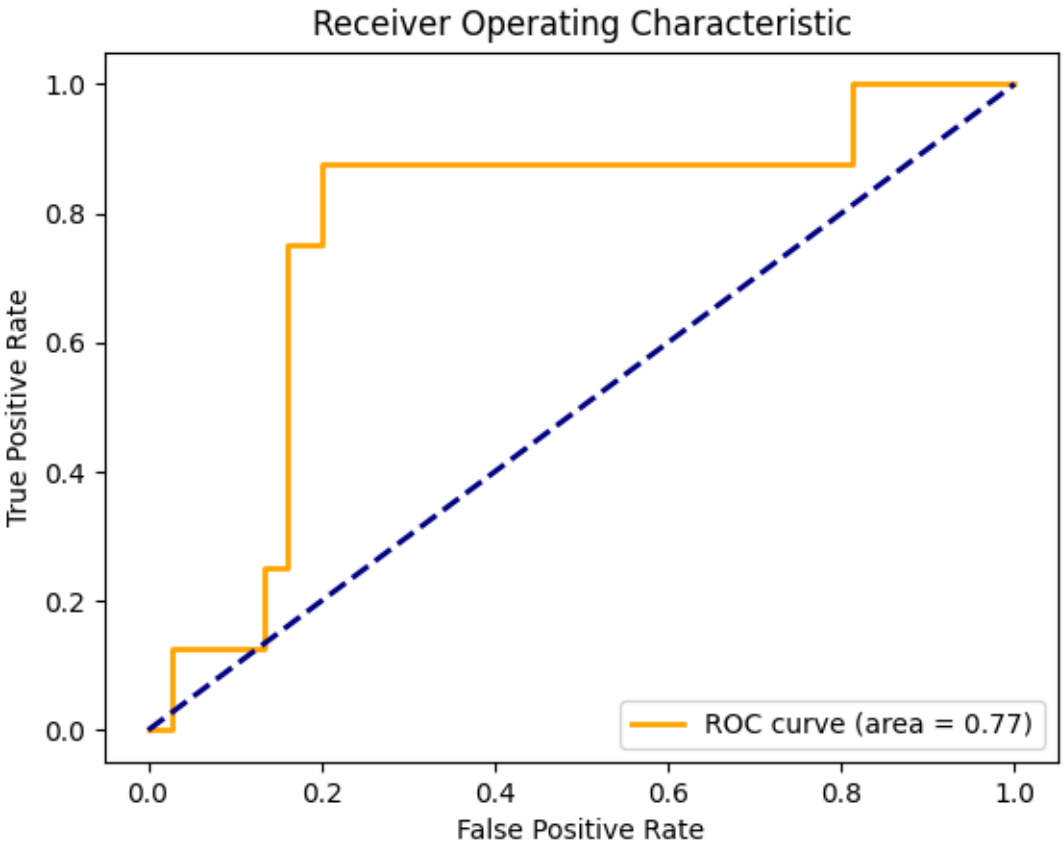

Figure 4.1: IMDb Dataset VAE Model ROC curve

Reddit Dataset Metrics

- Precision = 0.19
- Recall = 0.40
- Accuracy = 0.78
- F1-Score = 0.26
- ROC-AUC = 0.60
- PR-AUC = 0.14

4.3 GAN Anomaly Detection

In the training phase, the GAN model is trained adversarially to generate graphs that mimic normal behaviour and distinguish between those and real graphs. This signifies that the model understands the underlying patterns and structures of normal graphs and identifies the ones that deviate from that as fake.

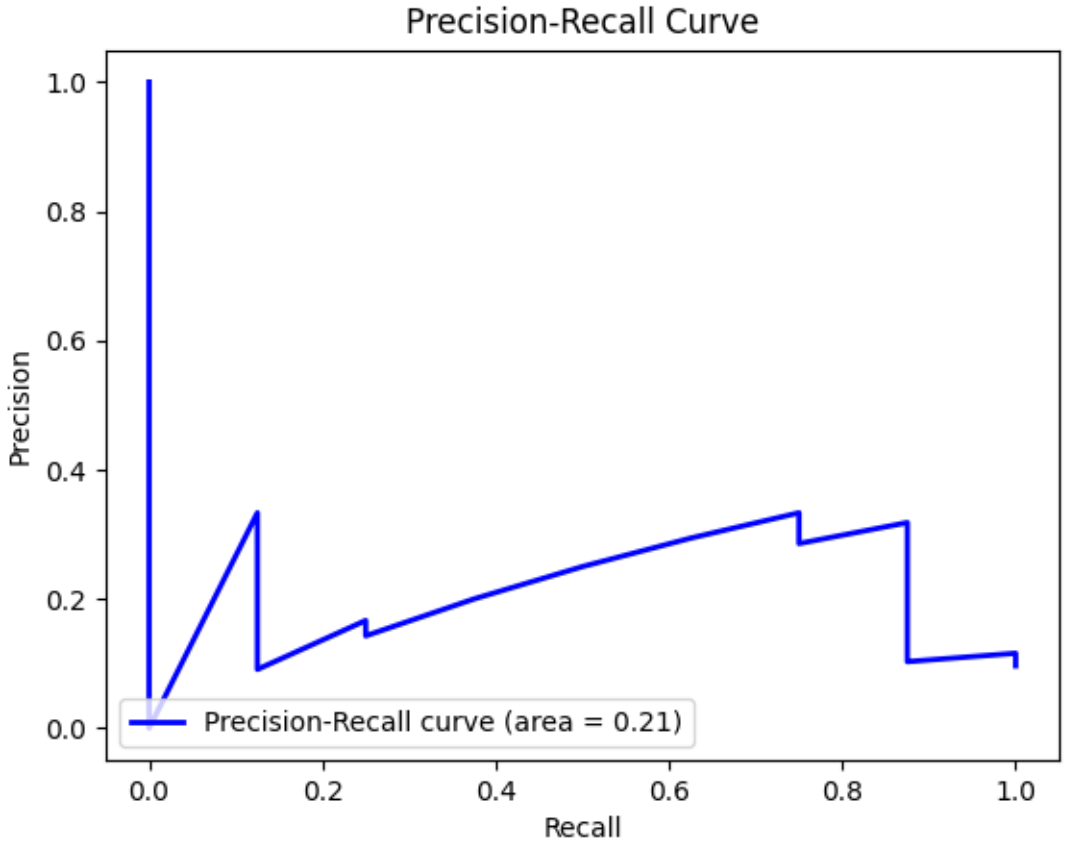

Figure 4.2: IMDb Dataset VAE Model Precision-Recall curve

In the test phase, the test data is passed to the discriminator part of the model. So, leveraging the fact that it was trained exclusively on normal instances, the discriminator can classify anomalies as fake data, as these instances, by definition, deviate significantly from the normal data points. The discriminator's output is a value between 0 for normal data and 1 for anomalous data; applying a threshold to this value makes it possible to identify the anomalies present in the dataset. This threshold is chosen by maximising the difference between the true and false positive rates.

IMDb Dataset Metrics

- Precision = 0.24
- Recall = 0.36
- Accuracy = 0.82
- F1-Score = 0.29
- ROC-AUC = 0.58
- PR-AUC = 0.19

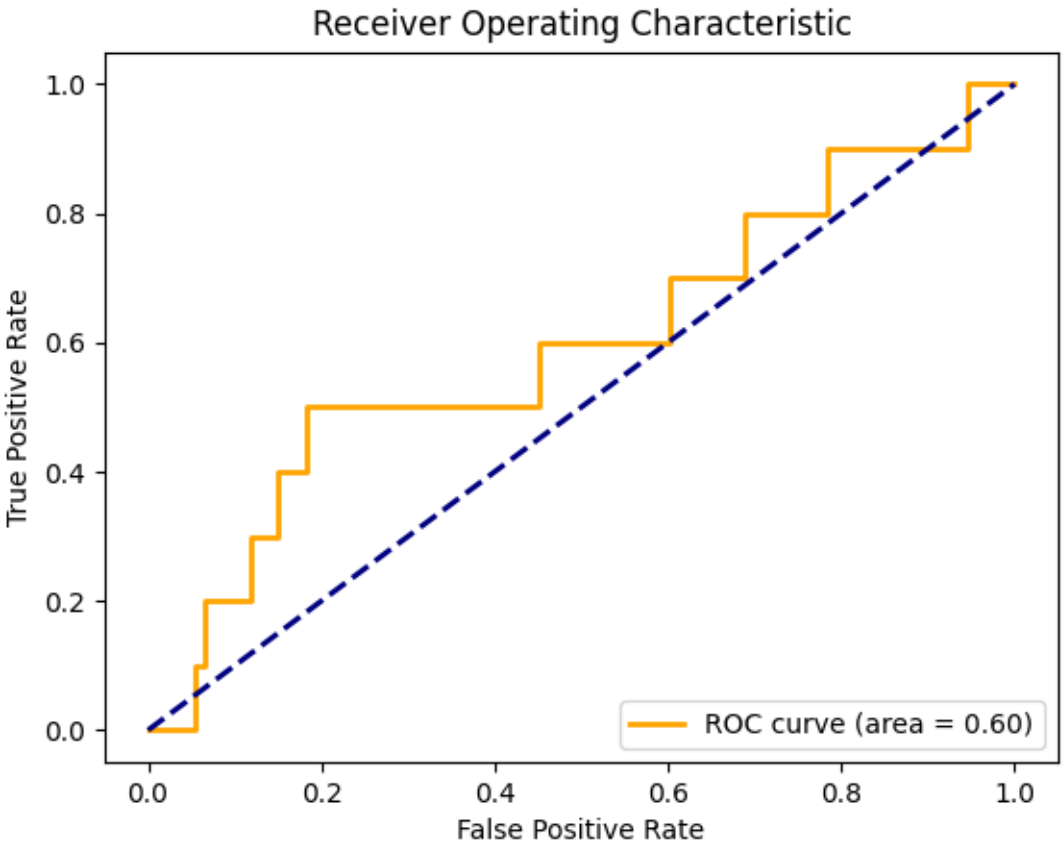

Figure 4.3: Reddit Dataset VAE Model ROC curve

Reddit Dataset Metrics

- Precision = 0.13
- Recall = 0.77
- Accuracy = 0.50
- F1-Score = 0.0.23
- ROC-AUC = 0.62
- PR-AUC = 0.12

4.4 Model Performance Comparison

As seen by the models' performance metrics, despite the low precision in both GAN and VAE models, the VAE models show a notably higher recall. Indicating they can more easily identify anomalies in this experiment.

Having developed and tested the models, it is apparent that VAE model offers marginally superior performance while being significantly more accessible in terms of their architecture

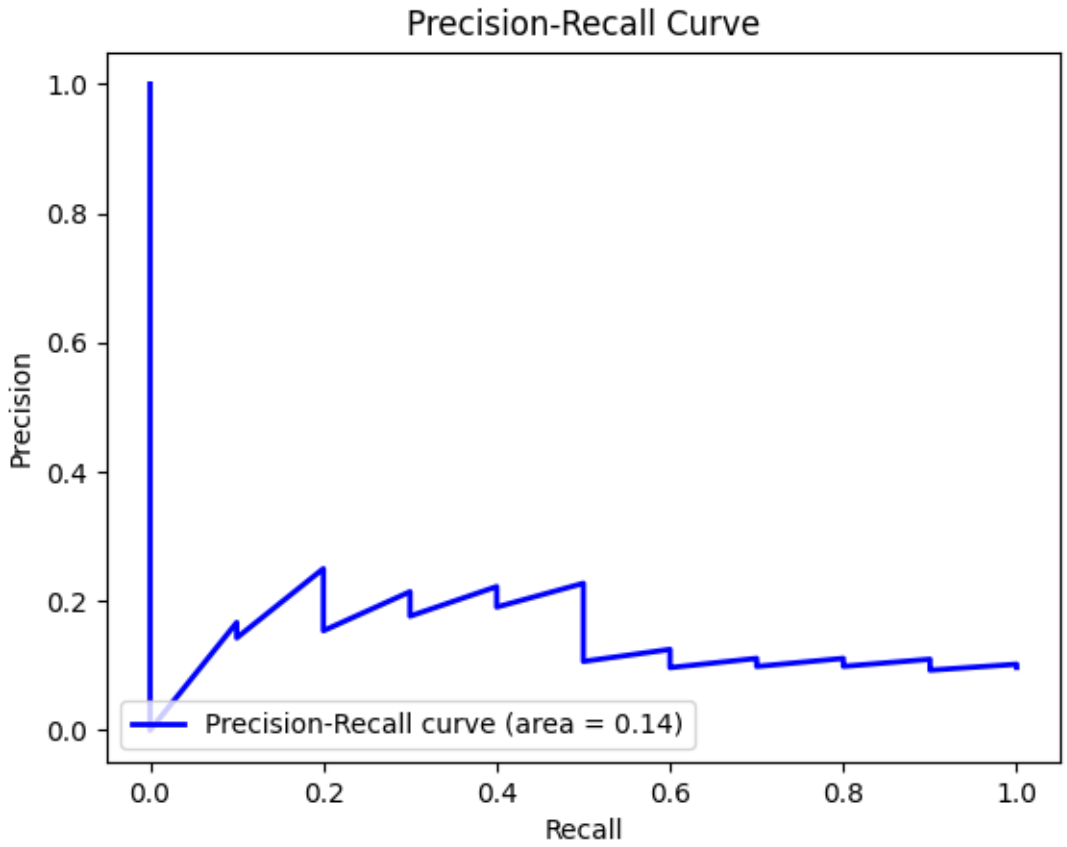

Figure 4.4: Reddit Dataset VAE Model Precision-Recall curve

and training while following a self-imposed rule to develop simple models and acknowledging the inherent difficulties and limitations of each approach.

4.4.1 IMDb Dataset

The VAE model gets a relatively high recall, which indicates that it can recognise a sizeable percentage of existing anomalies. Its low precision, however, raises the possibility of a large proportion of false positives. The F1-Score demonstrates a balance between precision and recall. The model has adequate discriminative capacity, as indicated by the AUC of the ROC and Precision-recall curves.

When compared to the VAE, the GAN model performs worse, with a much lower recall, which indicates that it misses more anomalies. Additionally, it exhibits a low accuracy and an acceptable F1-Score. When compared to the VAE, the AUC of the ROC and PR curves both indicate lower discriminative power.

4.4.2 Reddit Dataset

The VAE model has an average performance with a moderate recall but low precision. The AUC of the curves corroborate the moderate discriminative power.

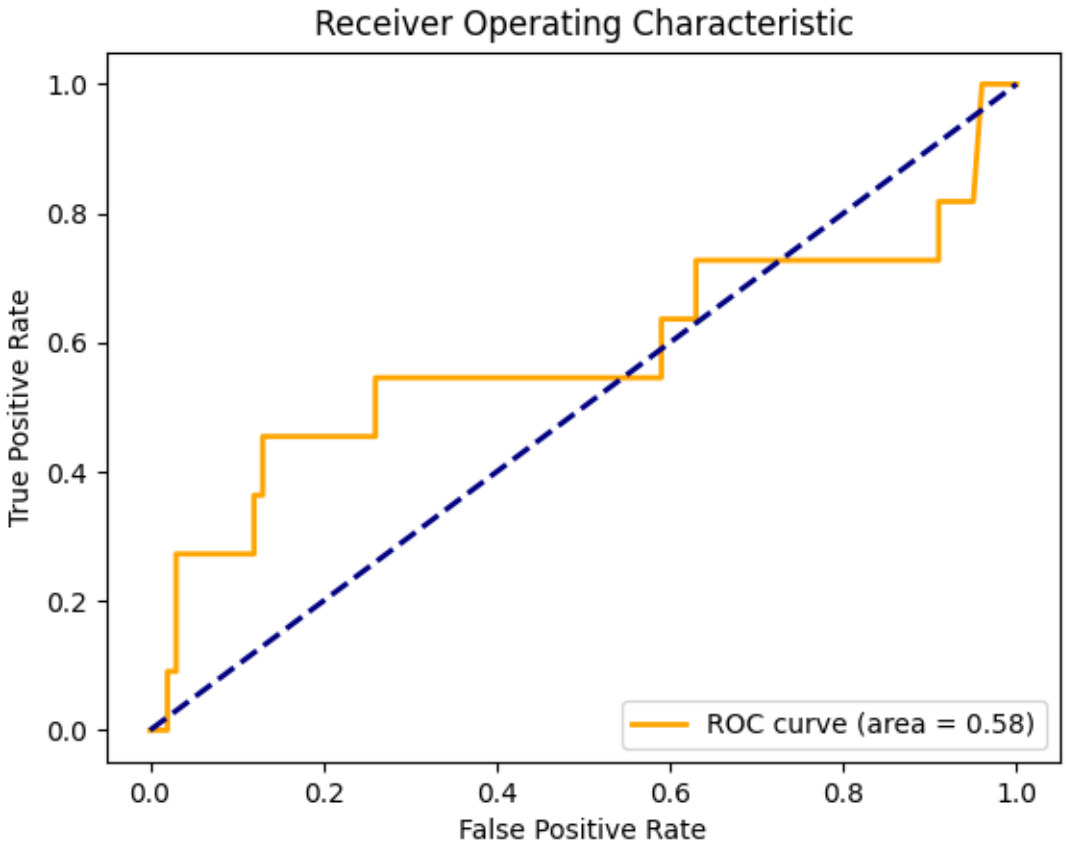

Figure 4.5: IMDb Dataset GAN Model ROC curve

The GAN model shows a notably high recall, which shows its effectiveness in identifying anomalies. The trade-off is the very low precision. This model also shows a notably lower accuracy. The AUC of this model's curves show a moderate discriminative power.

4.5 Identified Challenges and Limitation

4.5.1 Low Overall Performance

Achieving a good overall performance is itself a challenge when developing models for anomaly detection. This project aimed to experiment with two models not designed to deal with graph data specifically. Whilst it did not aim to perfect a singular model for anomaly detection, the results obtained from both models were lower than the initially set expectations. The results obtained reflect the challenges and limitations that surged during the research.

4.5.2 Limiting the Research to Simple Models

The use of more straightforward models was a self-imposed constraint when developing these models due to time and computational resources limitations.

Despite being allowed to finish the project in the available time frame, they presented

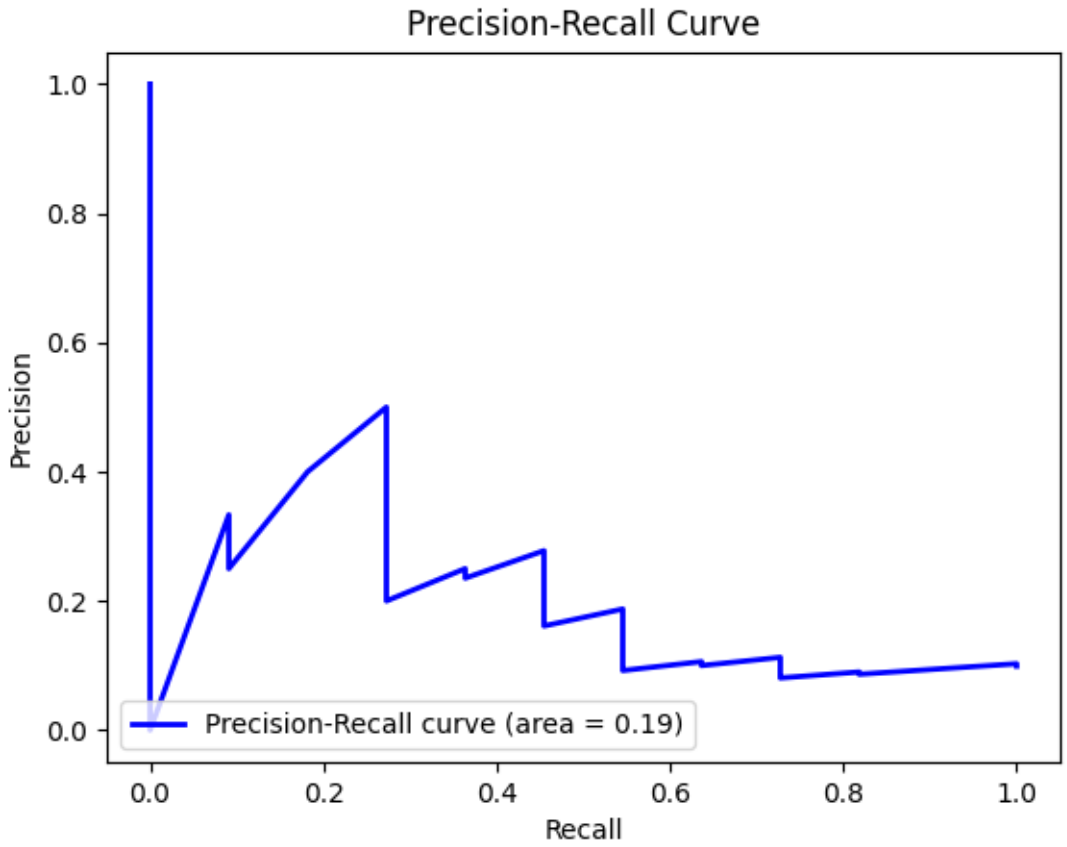

Figure 4.6: IMDb Dataset GAN Model Precision-Recall curve

limitations in terms of their capabilities to learn and understand the complex structure that is graph data. The use of these simpler models makes it easier to explain and understand the processes that occur in the models.

4.5.3 Hyperparameter Definition and Model Architecture

Hyperparameters play a central role in the performance of the models and finding the appropriate set is a complex task that, more than understanding the data, involves a trial-and-error process.

Along with hyperparameters, the models' architecture had to be created considering the available computation resources while still learning and understanding the complex and intricate graph data.

4.5.4 High-Dimensional Input Data

As mentioned in chapter 3, the adjacency matrix of the datasets were very large as they had shape [136, 136] for the IMDb dataset and shape [198,198] for the Reddit dataset. Compared to image data with shape [28,28], where VAE and GAN are normally applied. This brings forth data sparsity, increased computational complexity, and requires more complex models to deal with the data.

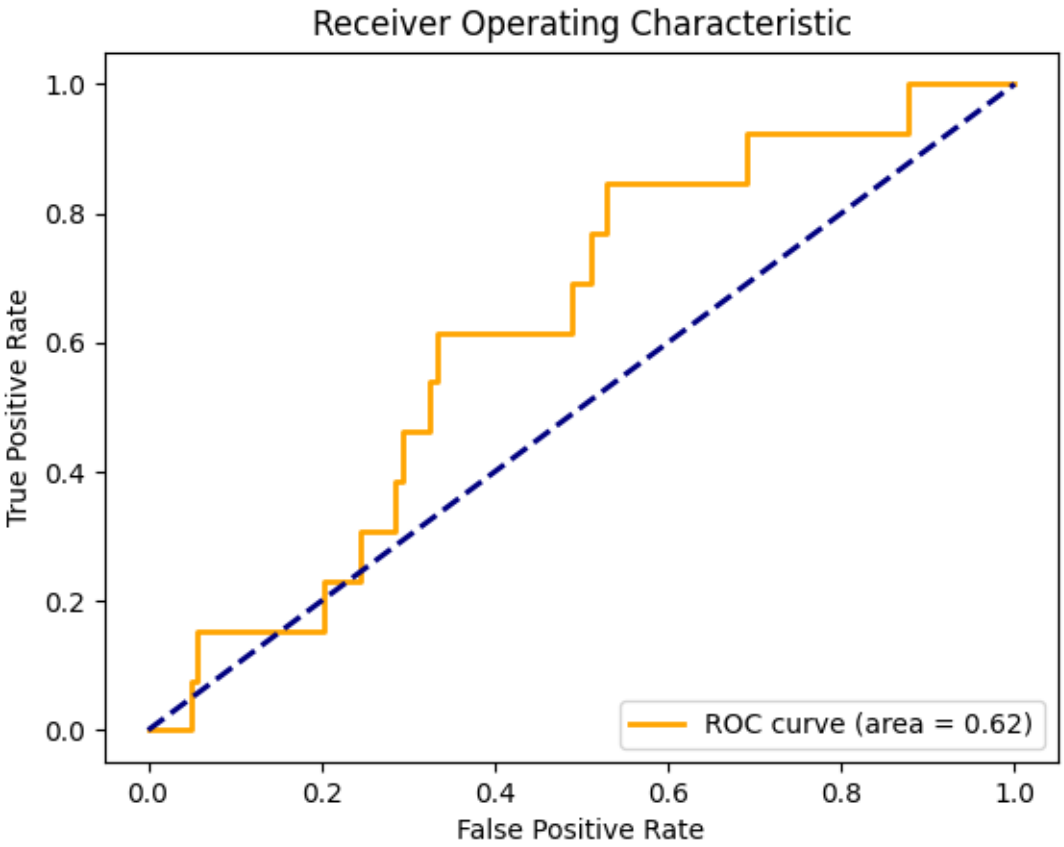

Figure 4.7: Reddit Dataset GAN Model ROC curve

Even though the data processing pipeline, explained in chapter 3 was effective in transforming the raw data into useful TensorFlow datasets using the subgraphs' adjacency matrices, the computing time was exponentially larger in the Reddit dataset as the matrices were larger.

Furthermore, a reduction of the dataset Reddit was done to avoid graphs with over 200 nodes. This decision was taken due to the extremely large size of the subgraphs (over 3000 nodes) present in this dataset, that triggered the software to crash.

Furthermore, the datasets used are binary classification datasets, which were processed to exhibit the characteristics of an anomaly detection dataset. This can lead to the data class processes representing anomalies not being different enough from the normal data.

4.5.5 Lack of Callbacks for GAN Training

Callbacks are an essential tool for training neural networks. Through the use of callbacks, the number of runs and the training time of the VAE models was considerably lower than the GAN models. Unfortunately, due to its unique architecture, conventional callbacks do not apply to GAN training.

Furthermore, the development of a callback, such as the early stop employed in VAE training, cannot be guided by the loss functions to determine when the GAN converges. This is because the generator and discriminator loss functions are not indicative of the capabilities

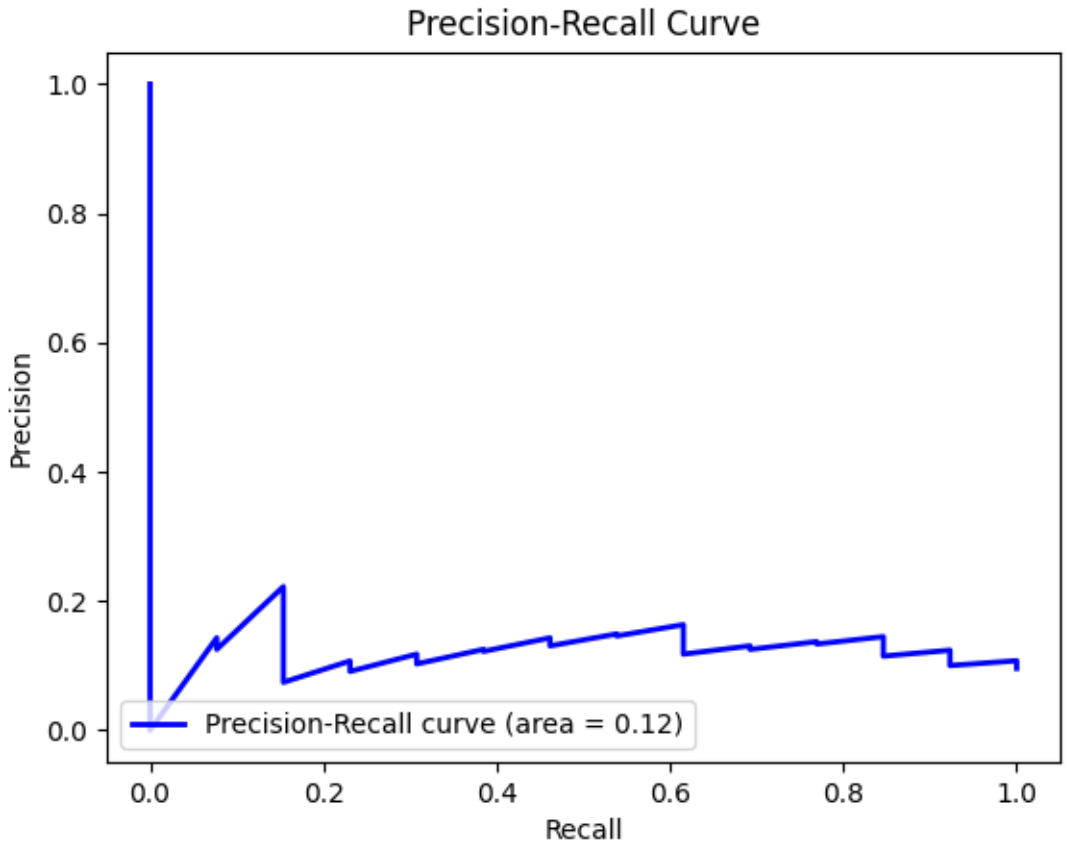

Figure 4.8: Reddit Dataset GAN Model Precision-Recall curve

of the model but the relation between the generator and discriminator.

4.5.6 Model Collapse during GAN Training

Another challenge of GAN models is the model collapse. Model collapse occurs when the generator or discriminator becomes exceptionally proficient at their task compared to their counterpart, and it becomes virtually impossible for the other component to improve.

The problem of model collapse requires experimenting with architectures of both generator and discriminator, as well as finding a balance between the learning rates for both components, to guarantee that neither component far exceeds the performance of its adversary.

4.5.7 Balancing the VAE's Loss Function

The loss function of the VAE models has two components: reconstruction loss and KL-divergence.

The reconstruction loss is responsible for measuring the difference between the initial input and the output, while the KL-divergence focuses on measuring the difference between the latent space distribution and a prior standard Gaussian distribution.

If the reconstruction loss overpowers the KL-divergence, the model becomes too focused on reconstructing the input correctly, ignoring the prior standard Gaussian distribution and eventually overfitting. If the regularisation loss is dominant, the model prioritises shaping the latent space as close as possible to the prior distribution at the expense of reconstructing the data correctly. The challenge is finding the correct weights to balance the components to guide the model through its training correctly.

Chapter 5

Conclusion

In conclusion, this project aimed to explore the application of generative deep learning models, more specifically, to apply variational autoencoders and generative adversarial networks to graph anomaly detection. Instead of pursuing the development of highly sophisticated GAN and VAE models, the primary goal was to craft straightforward and uncomplicated models and concentrate effort on conducting a comparison of these models, elucidating their respective strengths and limitations, as well as the process of building and training them for the specific task of graph anomaly detection.

By comparing the process of developing and training both models, it is possible to conclude that VAE models are much simpler and easier to build and train models, as they show less intricate nuances. Also, the ease of implementation of callbacks makes it much more time efficient. Furthermore, the VAE models had marginally better performances than the GAN models, as observed by the calculated metrics.

One of the key points of the research is the complexity of working with high-dimensional graph data. This kind of data demands substantial computational resources, leading to model training time and scalability challenges. The built data pipeline was able to effectively create the TensorFlow datasets, but the processing time grew exponentially with the size of the adjacency matrices and even demonstrating incapability when trying to use the whole Reddit dataset, which had subgraphs with over 3000 nodes.

Another point worth noting is that for this project a choice to use simple, straightforward and easy to understand. By making such choice the performance of the models clearly suffered, but this simplicity provides a clearer insight of the underlying mechanisms of the models and allowed for a shorter development time.

Lastly, a challenge that was already identified in the literature was obtaining real-world anomaly detection graph datasets with ground-truth labels. This poses an obstacle to rigorous experimentation and model evaluation. So, a critical step towards developing the field of graph anomaly detection is to curate and benchmark datasets tailored for this task.

While this study produced valuable insights into the use of Variational Autoencoders and Generative Adversarial Networks for graph anomaly detection, there are still many points that require further research and exploration in this topic.

It was reiterated through this dissertation that the aim of the project never was to develop a state-of-the-art model. So, something that is of interest for further research is to refine models to obtain better results in the task of graph anomaly detection.

A point that is worth nothing is that balanced graph binary classification datasets were

altered to exhibit anomaly detection datasets characteristics. So, a way to further develop this comparison of models is to use data that is specifically curated and tailored for the problem of graph anomaly detection.

The use of adjacency matrix proved difficult when the subgraphs of the datasets were too large as seen with the Reddit dataset. So, investigating ways to treat larger and more complex graph data efficiently to make this models more scalable is a promising area.

Bibliography

Ahmed, M., Naser Mahmood, A., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19-31. Retrieved from https://www.sciencedirect.com/science/article/pii/S1084804515002891 doi: https://doi.org/10.1016/j.jnca.2015.11.016

Akoglu, L., Tong, H., & Koutra, D. (2014). *Graph-based anomaly detection and description: A survey.* arXiv. Retrieved from https://arxiv.org/abs/1404.4679 doi: 10.48550/ARXIV.1404.4679

Arjovsky, M., Chintala, S., & Bottou, L. (2017). *Wasserstein gan.* arXiv. Retrieved from https://arxiv.org/abs/1701.07875 doi: 10.48550/ARXIV.1701.07875

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data.* Wiley. Retrieved from https://books.google.pt/books?id=B44QAQAAIAAJ

Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014, First). Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys & Tutorials*, *16*(1), 303-336. doi: 10.1109/SURV.2013.052213.00046

Branco, B., Abreu, P., Gomes, A. S., Almeida, M. S. C., Ascensão, J. a. T., & Bizarro, P. (2020). Interleaved sequence rnns for fraud detection. In *Proceedings of the 26th acm sigkdd international conference on knowledge discovery & data mining* (p. 3101–3109). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3394486.3403361 doi: 10.1145/3394486.3403361

Chandola, V., Banerjee, A., & Kumar, V. (2009, 07). Anomaly detection: A survey. *ACM Comput. Surv.*, *41*. doi: 10.1145/1541880.1541882

Cozzatti, M., Simonetta, F., & Ntalampiras, S. (2022). *Variational autoencoders for anomaly detection in respiratory sounds.* arXiv. Retrieved from https://arxiv.org/abs/2208.03326 doi: 10.48550/ARXIV.2208.03326

Foorthuis, R. (2021, Oct 01). On the nature and types of anomalies: a review of deviations in data. *International Journal of Data Science and Analytics*, *12*(4), 297-331. Retrieved from https://doi.org/10.1007/s41060-021-00265-1 doi: 10.1007/s41060-021-00265-1

Gama, J. a., Žliobaitundefined, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014, mar). A survey on concept drift adaptation. *ACM Comput. Surv.*, *46*(4). Retrieved from https://doi.org/10.1145/2523813 doi: 10.1145/2523813

González-Muñiz, A., Díaz, I., Cuadrado, A. A., García-Pérez, D., & Pérez, D. (2022a). Two-step residual-error based approach for anomaly detection in engineering systems using variational autoencoders. *Computers and Electrical Engineering*, *101*, 108065. Retrieved from https://www.sciencedirect.com/science/article/pii/S0045790622003226 doi: https://doi.org/10.1016/j.compeleceng.2022.108065

González-Muñiz, A., Díaz, I., Cuadrado, A. A., García-Pérez, D., & Pérez, D. (2022b). Two-step residual-error based approach for anomaly detection in engineering systems using

variational autoencoders. *Computers and Electrical Engineering*, *101*, 108065. Retrieved from https://www.sciencedirect.com/science/article/pii/S0045790622003226 doi: https://doi.org/10.1016/j.compeleceng.2022.108065

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

Grubbs, F. E. (1969). Procedures for detecting outlying observations in samples. *Technometrics*, *11*(1), 1–21. Retrieved 2022-12-01, from http://www.jstor.org/stable/1266761

Hawkins, D. (1980). *Identification of outliers*. Springer Netherlands. Retrieved from https://books.google.pt/books?id=P8ZPAQAAIAAJ

Liu, F., Ma, X., Wu, J., Yang, J., Xue, S., Beheshti, A., … Aggarwal, C. C. (2022). *Dagad: Data augmentation for graph anomaly detection.* arXiv. Retrieved from https://arxiv.org/abs/2210.09766 doi: 10.48550/ARXIV.2210.09766

M.A., F. E. (1887). Xli. on discordant observations. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *23*(143), 364-375. Retrieved from https://doi.org/10.1080/14786448708628471 doi: 10.1080/14786448708628471

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. Retrieved from https://doi.org/10.1109%2Ftkde.2021.3118815 doi: 10.1109/tkde.2021.3118815

Miller, Z., Dickinson, B., Deitrick, W., Hu, W., & Wang, A. H. (2014). Twitter spammer detection using data stream clustering. *Information Sciences*, *260*, 64-73. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025513008037 doi: https://doi.org/10.1016/j.ins.2013.11.016

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021, mar). Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, *54*(2). Retrieved from https://doi.org/10.1145/3439950 doi: 10.1145/3439950

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 doi: https://doi.org/10.1016/j.dss.2020.113303

Radford, A., Metz, L., & Chintala, S. (2015). *Unsupervised representation learning with deep convolutional generative adversarial networks.* arXiv. Retrieved from https://arxiv.org/abs/1511.06434 doi: 10.48550/ARXIV.1511.06434

Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C., & Samatova, N. F. (2015). Anomaly detection in dynamic networks: a survey. *WIREs Computational Statistics*, *7*(3), 223-247. Retrieved from https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.1347 doi: https://doi.org/10.1002/wics.1347

Reis, J. C. S., Correia, A., Murai, F., Veloso, A., & Benevenuto, F. (2019). Supervised learning for fake news detection. *IEEE Intelligent Systems*, *34*(2), 76-81. doi: 10.1109/MIS.2019.2899143

Rossi, R. A., & Ahmed, N. K. (2015). The network data repository with interactive graph analytics and visualization. In *Aaai*. Retrieved from https://networkrepository.com

Schlegl, T., Seeböck, P., Waldstein, S. M., Langs, G., & Schmidt-Erfurth, U. (2019). f-anogan:

Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, *54*, 30-44. Retrieved from https://www.sciencedirect.com/science/article/pii/S1361841518302640 doi: https://doi.org/10.1016/j.media.2019.01.010

Sun, J., Wang, X., Xiong, N., & Shao, J. (2018). Learning sparse representation with variational auto-encoder for anomaly detection. *IEEE Access*, *6*, 33353-33361. doi: 10.1109/ACCESS.2018.2848210

Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., … Qiao, H. (2018). Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In (p. 187–196). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. Retrieved from https://doi.org/10.1145/3178876.3185996 doi: 10.1145/3178876.3185996

Zheng, P., Yuan, S., Wu, X., Li, J., & Lu, A. (2018). *One-class adversarial nets for fraud detection.* arXiv. Retrieved from https://arxiv.org/abs/1803.01798 doi: 10.48550/ARXIV.1803.01798

Appendix A

Appendix

A.1 Tool Selection

A.1.1 Python

Python is a interpreted, object-oriented, high-level programming language with dynamic semantics. This programming language will be the main tool for the development of this project as there are several packages that support the development and implementation of deep learning algorithms. Furthermore, due to Python's vast community, there is an extensive documentation and quantity of available resources available.

A.1.2 NetworkX

The NetworkX is an open-source library focused on creating, visualising and analysing graph data. The benefits of using this library is the ease of use due to its user-friendly and intuitive API, the versatility to work with several types of graphs and the its integration with other libraries required to manipulate the data, namely NumPy and pandas, and visualise it with Matplotlib.

A.1.3 TensorFlow

The main framework to be used in this project is TensorFlow. TensorFlow is a comprehensive, open-source, end-to-end machine learning framework. TensorFlow is notably flexible, as it can be used for a wide-range of applications, scalabel, meaning it is designed to be able to handle both small and large-scale projects and being one of the most popular deep learning frameworks, offers a large community support. Furthermore, Keras, a high-level neural networks Application Programming Interface (API),has a seamless integration in TensorFlow making it accessible, while still being flexible.

A.1.4 Scikit-learn

Scikit-learn is a popular and frequently used Python machine learning library. It is an open-source framework that offers a straightforward and useful tools for data analysis and modelling, such as metrics and support the graphic representation of the models.