Robust Graph Convolutional Networks for Adversarial Resilience and Anomaly Detection

MD Shakib Khan

A Thesis
in
The Concordia Institute
for
Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science (Quality Systems Engineering) at
Concordia University
Montreal, QC, Canada

August 2024

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Md. Shakib Khan

Entitled: Robust Graph Convolutional Networks for Adversarial Resilience and Anomaly Detection

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Quality Systems Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. J. Bentahar

_____ Examiner
Dr. J. Bentahar

_____ Examiner
Dr. A. Ayub

_____ Thesis Supervisor(s)
Dr. A. Ben Hamza

_____ Thesis Supervisor(s)
Dr. A. Youssef

Approved by _____
Dr. F. Naderkhani (GPD) Chair of Department or Graduate Program Director

_____
Dr. M. Debbabi Dean of Gina Cody School

Abstract

Robust Graph Convolutional Networks for Adversarial Resilience and Anomaly Detection

MD Shakib Khan

Adversarial attacks and anomaly detection are closely related, both focusing on identifying irregularities that deviate from normal patterns across various data types, including graph-structured data. Adversarial attacks on graphs pose a significant threat to graph convolutional networks (GCNs), involving intentional manipulation of graph data to mislead GCNs into making incorrect predictions. Standard GCNs, while powerful, often exhibit vulnerabilities to adversarial attacks that can significantly degrade their performance in anomaly detection tasks. These networks also have inherent limitations, such as their inability to effectively consider higher-order neighbor information, restricting their capacity to capture the full context of a node within the graph. To address these challenges, this thesis introduces an iterative graph filtering framework, which builds upon the graph signal processing concept of iteratively solving graph filtering using the fixed-point iterative method. The proposed framework is designed to enhance resilience against adversarial attacks while improving anomaly detection capabilities. The thesis makes two main contributions: a flexible spectral modulation filter that selectively attenuates high-frequency components of graph signals; and a robust aggregation mechanism that efficiently captures information from higher-order node neighbors, expanding the networks receptive field without increasing computational complexity. Extensive experiments are conducted on benchmark datasets to evaluate the effectiveness of the proposed methods. The results demonstrate significant improvements in anomaly detection accuracy and adversarial robustness compared to strong baselines. This highlights the potential of the proposed framework for reliable graph-based downstream tasks, paving the way for robust GCNs that can handle the complexities and adversarial threats inherent in real-world applications.

Acknowledgments

I would like to express my profound gratitude to the exceptional professors who have played pivotal roles in my Master's journey. Professor Abdessamad Ben Hamza and Professor Amr Youssef have been true inspiration throughout my academic pursuits. Professor Ben Hamza dedication to fostering innovation and critical thinking has been truly remarkable. His encouragement to explore new ideas and tackle challenges head-on has been instrumental in my growth as a researcher. The invaluable feedback and advice he provided not only kept me motivated but also ignited a passion for continuous learning. Professor Youssef constant encouragement has been crucial in helping me navigate the complexities of graduate studies.Additionally, I extend my thanks to my lab mates, Hasib Zunair, Zadid Hasan, Zaedul Islam, Md. Tanvir Hassan, Abu Taib Mohammed Shahjahan whose constant support has been instrumental in enriching my experience. Their great ideas and encouragement have played a significant role in shaping my academic pursuits. Reflecting on my time at Concordia, I am deeply appreciative of all my instructors, who have contributed to my growth and learning in profound ways. Their dedication to education has been truly inspiring. Finally, I owe immeasurable gratitude to my parents and sibling for their unconditional love and unwavering support. Their belief in my potential, endless encouragement, and prayers have been the pillars of strength, especially during the most challenging times. I want to extend a special thank you to my mother and father, whose selfless sacrifices and prayers have been an endless source of motivation and hope. The support I have received from all these remarkable individuals and institutions has been transformative, and I am filled with gratitude for their role in my academic and personal development.

Table of Contents

List of Figures

List of Tables

List of Acronyms

DNNs Deep Neural Networks

GCN Graph Convolutional Networks

Fix-GCN Fixed-Point Graph Convolutional Network

ResNets Residual Networks

AUC Area Under the Curve

CHAPTER 1

Introduction

In this chapter, we outline the motivation behind this work, provide a succinct problem statement and objectives, review relevant literature, and highlight the key contributions of the thesis. The literature review section provides a comprehensive overview of graph neural networks against adversarial attacks, graph anomaly detection, and graph encoder-decoder models.

1.1 Motivation and Background

Graph adversarial attacks are sophisticated techniques designed to manipulate or perturb graph data with the intent of misleading graph learning models. These attacks aim to cause incorrect predictions or classifications by exploiting vulnerabilities in graph-based machine learning systems. Common strategies include modifying graph structure by adding or removing edges, perturbing node features, or even altering the node set itself. Adversarial attacks on graphs can be broadly classified into two categories: poisoning attacks, which occur during the training phase, and evasion attacks, which target already trained models at test time [1]. Several well-known attacking methods have been developed to demonstrate the potential vulnerabilities of graph learning models. For instance, Nettack [2] aims to misclassify specific nodes in the graph by strategically perturbing the graph structure during the training phase. Another notable method is Mettack [3], which takes a broader approach by aiming to degrade the overall performance of Graph Neural Networks (GNNs) through structural perturbations. This untargeted strategy emphasizes the potential for widespread disruption of graph-based models. However, various approaches have been proposed to address the graph adversarial problem. In the realm of defense against adversarial at-

tacks, Robust GCN (RGCN) [4] employs Gaussian distributions to represent node embeddings and incorporates a variance-based attention mechanism, enhancing the model's resilience. Another defensive strategy, GNN-Jaccard [5], focuses on pre-processing by removing edges with low Jaccard similarity, operating under the assumption that clean graphs adhere to homophily principles.

Another major issue with graphs is anomalies which are unusual patterns, behaviors, or structures within graph-structured data that significantly deviate from the expected norm. These anomalies can manifest as unusual individual nodes, unexpected connections between nodes, atypical subgraphs, or entire graphs that differ from typical structures. Detecting such anomalies is crucial in various domains, including network security, fraud detection in financial networks, social media spam identification, and e-commerce interaction analysis. The importance of graph anomaly detection lies in its ability to reveal critical insights about system behaviors, potential security threats, or fraudulent activities that might otherwise go unnoticed. However, this field faces several challenges due to the complexity of graph data [6], which encompasses both topological and attribute information, the rarity of anomalies, and the need to consider both local and global graph structures. Approaches to tackle these challenges range from traditional statistical and clustering-based methods to modern techniques leveraging Graph Neural Networks (GNNs), which have gained popularity for their ability to learn complex graph representations. As graph-structured data becomes increasingly prevalent and complex, the field of graph anomaly detection continues to evolve, with ongoing research aimed at developing more accurate and efficient methods to uncover these critical patterns and behaviors in various network structures. However, several methods have been developed to improve accuracy. ResGCN [7], for instance, leverages GCNs to capture sparsity and nonlinearity in attributed graphs, learning residual information and employing a residual-based attention mechanism to mitigate the impact of anomalous nodes. AMEN [8] takes a different approach by considering ego-network information for each node to discover anomalous neighborhoods in attributed networks. Additionally, a separate family of methods focuses on identifying abnormal nodes within node feature subspaces. These diverse approaches collectively demonstrate the ongoing efforts to enhance both the robustness of graph neural networks against adversarial attacks and the accuracy of anomaly detection in graph-structured data, addressing the unique challenges posed by the complex nature of graph data.

Encoder-Decoder Networks Graph anomaly detection using encoder-decoder architectures has emerged as a promising approach in recent years, leveraging the power of graph representation learning to identify unusual patterns or behaviors in graph-structured data. This method has gained traction in various domains, including network security, fraud detection, and social media analysis, where traditional approaches often struggle with the complex topological relationships inherent in

graph data. The encoder-decoder framework typically consists of an encoder, often implemented using graph convolutional networks, which maps the input graph attribute data matrix $\mathbf{X}$ into a low-dimensional latent representation $\mathbf{Z}$, and a decoder that attempts to reconstruct the original graph attributes from this latent representation. This process can essentially be represented as $\mathbf{Z} = \text{Encoder}(\mathbf{X})$ and $\hat{\mathbf{X}} = \text{Decoder}(\mathbf{Z})$, where $\hat{\mathbf{X}}$ is the reconstructed graph attribute data matrix. The reconstruction error, typically measured as the difference between $\mathbf{X}$ and $\hat{\mathbf{X}}$, is then used as an anomaly score, based on the assumption that anomalies will be harder to reconstruct accurately and thus have higher reconstruction errors. Recent developments in this field include unsupervised approaches that learn anomaly scoring functions without requiring labeled training data, and reconstruction-based methods that focus on rebuilding both graph structure and nodal features.

Residual Connections. Residual connections are a crucial architectural innovation in deep learning models (DNNs, especially in deep neural networks with numerous layers. They were first introduced in the context of residual networks (ResNets) and have since become a fundamental building block in various state-of-the-art architectures. In light of this insight, incorporating residual connections into our architecture also proves beneficial. Traditional deep neural networks can encounter difficulties in training as the number of layers increases. By introducing residual connections, deeper networks become easier to optimize, as they enable the model to retain essential information from earlier layers and build upon it in subsequent layers. This facilitates the training process and allows the network to explore more complex and meaningful representations. In addition, residual connections enable the reuse of learned features from previous layers in subsequent ones. Moreover, residual connections effectively allow a neural network to learn identity mapping, i.e., passing the input directly to the output. This is crucial when the optimal mapping between input and output is close to an identity mapping. Without residual connections, the network would need to learn this mapping from scratch, which can be inefficient and challenging, especially in very deep architectures.

Dropout. Dropout is a regularization technique used in neural networks to mitigate the problem of overfitting. Overfitting occurs when a neural network becomes too specialized in learning from the training data and fails to generalize well to unseen or new data. In a typical neural network, each neuron in a layer connects to neurons in the previous and subsequent layers, forming a dense network of connections. During training, dropout randomly "deactivates" or "turns off" a certain fraction of neurons in the network with a probability p. This means that those neurons and their connections are temporarily ignored for the current forward and backward pass. Therefore, we also apply dropout layers to our models.

1.2 Problem Statement

In this section, we briefly describe the main problems addressed in this thesis: graph adversarial attacks and graph anomaly detection. Graphs serve as powerful tools for representing relationships between entities in various domains, including social networks, e-commerce platforms, citation networks, shape classification and retrieval [9–13], geometry processing [14], and mesh watermarking [15]. Formally, a graph is denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \ldots, N\}$ is the set of N nodes or vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges or links connecting pairs of vertices. The ability to detect and mitigate adversarial attacks, as well as identify anomalies, is critical for the integrity and reliability of graph-based systems.

1.2.1 Graph Adversarial Attacks

Graph adversarial attacks involve the intentional manipulation of graph data to deceive graph-based models, such as graph neural networks, into making incorrect predictions. These attacks can be categorized into:

- **Evasion Attacks:** Perturbations are made during the inference phase to alter the predictions of a pre-trained graph neural network. Attackers may add, delete, or modify edges and node features to deceive the model into making incorrect classifications or recommendations.
- **Poisoning Attacks:** These involve tampering with the training data. Attackers introduce small but strategically crafted changes to the graph structure or node features during the training phase, causing the graph neural network to learn incorrect patterns, which degrade its performance on clean test data.
- **Targeted Attacks:** Specific nodes or edges are targeted to achieve desired misclassifications. For example, in a social network, an attacker might aim to misclassify a particular user as belonging to a different community.
- **Non-Targeted Attacks:** The goal is to cause a general degradation in the model's performance, without focusing on specific nodes or edges.

Adversarial attacks on graphs pose significant challenges, as graph perturbations can be subtle and hard to detect. Also, graph neural networks trained on adversarially perturbed graphs often fail to generalize, leading to poor performance. In addition, the susceptibility of graph neural networks to adversarial attacks undermines their reliability, limiting their adoption in sensitive domains.

1.2.2 Graph Anomaly Detection

Graph anomaly detection aims to identify nodes or edges that exhibit unusual or unexpected behavior, deviating from the normal patterns within the graph. This task is particularly challenging due to the rarity of anomalies, as only a small fraction of nodes or edges are typically anomalous. Effective graph anomaly detection is essential for applications such as fraud detection in financial networks, intrusion detection in communication networks, and identifying anomalous activities in social networks. For instance, identifying fraudulent transactions or malicious accounts in banking and online payment systems can prevent significant financial losses and protect users. Also, detecting intrusions, unauthorized access, or abnormal behaviors in computer networks and communication systems can help in mitigating cyber threats and attacks. In addition, detecting unusual user behavior or interactions in social media can help in identifying bots, fake accounts, or harmful activities.

Despite its importance, anomaly detection on graph-structured data presents several significant challenges, including the complex structures of graphs that can have diverse and complex topologies (e.g., dense or sparse graph structures), the scarcity of labeled anomalies, the high dimensionality of attributes that must be considered alongside the structural information, and the need to balance local and global patterns as some anomalies are localized to small regions of the graph (e.g., a single fraudulent transaction in a financial network), while other anomalies might be spread across the graph (e.g., a widespread network attack affecting multiple nodes). Another key challenge lies in accurately identifying anomalies within complex and often noisy graph data. Traditional methods may fall short in effectively capturing both the structural and attribute information necessary for robust anomaly detection. Also, real-world graph data often contain noise and ambiguous patterns, requiring the model to distinguish between true anomalies and noise-induced outliers.

1.3 Objectives

The aim of this thesis is to develop robust graph convolutional networks for adversarial resilience and anomaly detection. Specifically, the objectives are to:

- Develop a novel model, called fixed-point iterative graph convolutional network (Fix-GCN), which achieves robustness against adversarial perturbations by effectively capturing higher-order node neighborhood information in the graph without additional memory or computational complexity.
- Design an unsupervised graph encoder-decoder model for anomaly detection in attributed

graphs. In the encoding stage, we design a fixed-point GCN encoder that allows for effective aggregation of information from higher-order neighborhoods. In the decoding stage, we employ a structure reconstruction decoder to predict the presence or absence of edges between nodes in the graph based on the latent representation obtained from the encoder, and an attribute reconstruction decoder to recover the original node features based on the graph structure and the learned latent representation.

- Evaluate the proposed methods through extensive experiments on standard benchmark datasets, demonstrating their superiority over existing techniques.

1.4 Literature Review

Graph Convolutional Networks Against Adversarial Attacks. Considerable efforts have recently been dedicated to devising defensive mechanisms that bolster the robustness of GNNs. For instance, Robust GCN (RGCN) [4] is a defense method that utilizes Gaussian distributions to represent node embeddings and incorporates a variance-based attention mechanism. Similar to the graph attention (GAT) model [16] that extends the fundamental aggregation function of GCN by assigning varying importance to each edge using attention coefficients, the attention mechanism of RGCN assigns weights to node neighborhoods based on their variances. Larger variances indicate a higher likelihood of being targeted by attacks. By leveraging these variances, RGCN penalizes the attention scores of adversarial edges, thereby reducing the propagation of adversarial effects through the graph. GNN-Jaccard [5] is a pre-processing defensive mechanism designed to enhance model robustness against adversarial attacks by removing edges from the graph that exhibit low Jaccard similarity based on the assumption that the clean graph adheres to homophily, where nodes with similar attributes are more likely to be connected. GCN-SVD [17] is another defense mechanism that operates at the preprocessing stage, specifically targeting high-rank perturbations like those induced by Nettack [2]. It operates by performing a low-rank approximation of the graph adjacency matrix through truncating its singular values via SVD. By focusing on retaining the low-frequency structural features while eliminating high-frequency perturbations, GCN-SVD aims to enhance the robustness of GNNs against adversarial attacks. Both GNN-Jaccard and GCN-SVD adopt a two-stage preprocessing approach to mitigate the effects of adversarial attacks. In the first stage, these methods preprocess the perturbed graphs to extract clean graph representations. This preprocessing step involves applying specific strategies to identify and remove perturbations introduced by adversarial attacks, such as edges with low Jaccard similarity in GNN-Jaccard or high-frequency components in the adjacency matrix in GCN-SVD. Once the clean graph repre-

sentations are obtained, the second stage involves training the GCN model on these clean graphs. As an effective approach, several defense methods propose to leverage the properties of low rank, sparsity, and feature smoothness in the graph structure [18, 19]. For instance, Pro-GNN [18] employs a joint learning framework to simultaneously learn the clean graph structure from perturbed data and optimize the parameters of the GNN. It imposes constraints on the graph structure, enforcing it to be low-rank and sparse through regularization, aligning it closely with the clean structure. However, the joint optimization process in Pro-GNN requires iterative updates to both the adjacency matrix and the model parameters, leading to increased computational complexity. GNNGuard [20] employs a defense strategy against adversarial attacks by assigning higher weights to edges connecting similar nodes and lower weights to edges connecting dissimilar nodes. The rationale behind this approach lies in the assumption that similar nodes are more likely to interact with each other. Although GNNGuard offers promising strategies for defending against adversarial attacks, it relies on the estimation of neighbor importance for every node and the memory layer for graph coarsening, which can incur significant computational overhead, especially for large graphs. While incorporating defense mechanisms into GNNs has demonstrated efficacy in countering adversarial attacks, a recent and promising line of work involves the design and development of robust GNNs by devising novel graph filters or feature aggregation schemes tailored to preserve information robustness against adversarial manipulations [21–25]. For instance, Simp-GCN [21] employs an adaptive message aggregation mechanism to integrate the graph structure and node features, as well as self-supervised learning to capture intricate relationships between node features, including both similarities and dissimilarities. Mid-GCN [25] employs a mid-pass graph filter to protect against adversarial attacks by leveraging the observation that the eigenvalues of the graph Laplacian within mid-frequency are less affected by such attacks, as mid-frequency signals tend to preserve information from higher-order neighbors.

Graph Anomaly Detection. Recently, graph anomaly detection has attracted increasing attention, leading to a plethora of innovative techniques, which can be classified into two main categories: (i) *Shallow Models*: The Local Outlier Factor (LOF) [26] algorithm can be employed to detect density-based local outliers, flagging nodes that exhibit significantly different density patterns compared to their surrounding regions. Additionally, SCAN [27] can be utilized for anomaly detection based on structural similarity by identifying nodes that deviate from their local neighborhood structure. (ii) *Spectral Models*: BWGNN [28] leverages a beta kernel to handle higher frequency anomalies through multiple flexible, spatial/spectral localized, and band-pass filters. The key idea is to employ filters that can effectively capture anomalous patterns across different frequency bands, both in the spectral and spatial domains. Another notable approach is SpecF [29]

which proposed a community-based anomaly detection algorithm using a spectral graph filter that incorporates the network community structure into the Laplacian matrix adopted as the basis for the Fourier transform. *(iii) Generative graph neural networks*: ALARM [30] proposes a multi-view representation learning framework with multiple graph encoders and a well-designed aggregator. Semi-GNN [31] is a semi-supervised graph neural model which adopts hierarchical attention to model the multi-view graph for fraud detection. SL-GAD [32] performs anomaly detection from both generative and multi-view contrastive perspectives. Sub-CR [33] then proposes a self-supervised method based on multi-view contrastive learning with graph diffusion and attribute reconstruction. *(iv) Enhanced graph neural networks*: Radar [34] proposes a graph anomaly detection framework that leverages residual analysis to identify deviations between predicted and observed attribute values in the graph. It focuses on detecting attribute-based anomalies in attributed graphs by analyzing the residuals, which are the differences between predicted attribute values and the observed attribute values. ResGCN [7] introduces a graph anomaly detection approach that captures the sparsity and nonlinearity present in attributed graphs through the use of GCNs, learns residual information, and employs a residual-based attention mechanism to mitigate the negative impact caused by anomalous nodes. AMEN [8] considers the ego-network information for each node and discovers anomalous neighborhoods on attributed networks. Besides that, another family of methods is focused on spotting abnormal nodes in a node feature subspace. OCGNN (One-Class Graph Neural Network) [35], a hypersphere learning framework which is designed to combine the powerful representation ability of GNNs with the classical hypersphere learning objective to detect anomalies. ComGA [36] tailored a deep graph convolutional network (tGCN) which propagates community-specific representation into its corresponding layers of GCN via multiple gateways. And, the hidden node representations of each layer in the netwok which can fuse local and structure anomalies information. *(v) Graph contrastive learning methods*: Mul-gad [37] & GRADATE [38] present a multi-view, multi-scale contrastive learning framework with subgraph-subgraph contrast for graph anomaly detection by combining various anomalous information and calculating the anomaly score for each node. HCM [39] models both local and global contextual anomalies by using hop counts prediction as a self-supervised task to train model. CoLA [40] designs a contrastive self-supervised learning framework to detect local anomalies within the graph.

Graph Encoder-Decoder Models. In recent years, graph encoder-decoder models have emerged as a prominent approach for graph anomaly detection. These methods leverage graph convolutional networks (GCNs) and their variants as the core components of the encoder-decoder architectures, effectively capturing and integrating both the structural and attribute information present in the

graph data. This unified representation learning capability is crucial for accurate anomaly detection, as anomalies can manifest in various forms, including deviations in node attributes, structural patterns, or a combination of both. Recently, Dominant [41] introduces a deep autoencoder framework that compresses input attributed networks into low-dimensional embeddings using a graph convolutional network as an encoder. It then reconstructs both the topological structure and node attributes with decoder functions, leveraging reconstruction errors to identify anomalous nodes. DeepAE [42] presents a method that utilizes the different order proximities for reconstructing the graph node attributes and structures. Also, in the decode they introduce a technique called Laplacian sharpening which can preserve the differences between normal nodes and anomalies to classify anomalies. GUIDE [43] utilizes a higher-order network structure with graph node attention layer which can learn different weights according to the structural differences between the node and its neighbor, they did this in autoencoder way, where they have used GCN and GNA for attribute and structure encoder respectively. GAAN [44] proposes a generative adversarial anomaly detection model where they generate fake nodes using Gaussian noise and employ a MLP based encoder to get the low-dimensional latent space. This method trains jointly reconstruction loss and discriminator loss to detect anomalies. AnomalyDAE [45] proposes a deep joint representation learning framework for anomaly detection through a dual autoencoder. Specifically, it consists of a structure autoencoder and an attribute autoencoder to learn both node embedding and attribute embedding jointly in latent space. Moreover, an attention mechanism employed in structure encoder to learn the importance between a node and its neighbors for an effective capturing of structure pattern. GAD-NR [46] incorporates neighborhood reconstruction method for graph anomaly detection.

1.5 Overview and Contributions

This thesis is structured as follows:

- Chapter 1 lays the foundation for this research by presenting a thorough examination of the studys motivations and objectives. It begins by articulating the problem statement, clearly defining the research goals, and outlining the study's scope. The chapter then delves into an extensive literature review, providing critical insights into the current state of knowledge in three key areas: deep learning algorithms for graph adversarial attacks, graph anomaly detection techniques, and autonomous vehicle datasets. This comprehensive review establishes the context for the study, highlighting gaps in existing research and setting the stage for the novel contributions presented in subsequent chapters.

- In Chapter 2, we develop a novel spectral modulation filter that selectively attenuates high-frequency components while preserving low-frequency structural information in the graph signal. This allows for better noise reduction and feature extraction. Furthermore, we implement an efficient aggregation mechanism that captures higher-order neighborhood information without increasing computational complexity. By combining these elements, our model effectively balances the preservation of essential graph structure with the incorporation of broader contextual information, leading to improved anomaly detection capabilities in attributed graphs.

- In Chapter 3, we present an encoder-decoder model for anomaly detection in attributed graphs. The encoder uses an efficient aggregation mechanism to spread higher-order neighborhood information among graph nodes. The decoder leverages a structure reconstruction decoder and a graph deconvolutional network. To reduce noise and retain essential graph information, we apply spectral graph wavelet denoising during decoding.

- Chapter 4 concludes the thesis by summarizing its key contributions, acknowledging limitations, and proposing future research directions in the fields of graph adversarial attacks, anomaly detection.

In addition to developing mechanisms that strengthen the robustness of GCNs against various types of adversarial attacks, including structural modifications and feature alterations, we design a GCN-based method to identify anomalous nodes within graph-structured data, leveraging structural and feature information to detect nodes that exhibit unusual or unexpected behavior. We also conduct extensive experiments on benchmark datasets to evaluate the proposed methods, ensuring that the framework delivers significant improvements in anomaly detection accuracy and adversarial robustness compared to existing baselines.

CHAPTER 2

Fixed-Point Graph Convolutional Networks Against Adversarial Attacks

In this chapter, we discuss robust graph convolutional networks against adversarial attacks. Adversarial attacks present a significant risk to the integrity and performance of graph neural networks, particularly in tasks where graph structure and node features are vulnerable to manipulation. In this thesis, we present a novel model, called fixed-point iterative graph convolutional network (Fix-GCN), which achieves robustness against adversarial perturbations by effectively capturing higher-order node neighborhood information in the graph without additional memory or computational complexity. Specifically, we introduce a versatile spectral modulation filter and derive the feature propagation rule of our model using fixed-point iteration. Unlike traditional defense mechanisms that rely on additional design elements to counteract attacks, the proposed graph filter provides a flexible-pass filtering approach, allowing it to selectively attenuate high-frequency components while preserving low-frequency structural information in the graph signal. By iteratively updating node representations, our model offers a flexible and efficient framework for preserving essential graph information while mitigating the impact of adversarial manipulation. We demonstrate the effectiveness of the proposed model through extensive experiments on various benchmark graph datasets, showcasing its resilience against adversarial attacks.

2.1 Introduction

Graph neural networks (GNNs), particularly graph convolutional networks (GCNs) [47] and their variants [48–50], have proven effective at learning representations of graph-structured data, demonstrating state-of-the-art performance in a wide variety of real-world applications, including weather forecasting [51], biomedicine and healthcare [52], traffic forecasting [53], and cybersecurity [54]. The key to the effectiveness of GNNs lies in the neural message passing scheme, which iteratively passes and aggregates feature information from neighboring nodes in the graph. However, despite their effectiveness, GNNs are vulnerable to adversarial attacks [2, 3], which involve intentionally crafted perturbations to the input graph data, such as modifying the graph structure or altering node features, with the goal of causing the model to make incorrect predictions.

Adversarial attacks on graphs can be classified into poisoning and evasion attacks based on the attacker's objectives and the timing of the attack [1–3,55]. Evasion attacks, also known as test-time attacks, aim to deceive the model and compromise its prediction performance at testing time. On the other hand, poisoning attacks, also known as training-time attacks, manipulate the training data to mislead the model during the training phase with the aim of degrading the model performance on downstream tasks. Moreover, poisoning attacks can be broadly categorized into targeted and non-targeted attacks. Targeted attacks, such as Nettack [2], aim to misclassify specific nodes in the graph by perturbing the graph structure during the training phase. Rather than focusing on specific nodes, non-targeted attacks, such as Mettack [3], aim to degrade the overall performance of GNNs by perturbing the graph structure (i.e., adding or modifying edges) during the training phase. These adversarial attacks pose significant challenges to the integrity and performance of GNN-based systems, especially in critical applications, such as healthcare, cybersecurity and autonomous driving, where trustworthiness and robustness are paramount. Hence, it is crucial to incorporate adversarial defense mechanisms into GNN models and/or develop and design robust and resilient model architectures that can resist adversarial attacks effectively.

While integrating defense mechanisms into GNN models has demonstrated effectiveness in thwarting adversarial attacks [4, 5, 17–19], these mechanisms often employ additional components that require extensive computations for a successful defense. For instance, GCN-SVD [17] preprocesses the graph adjacency matrix using singular value decomposition (SVD), retaining only the low-frequency components to defend against adversarial attacks on graph structures. The basic idea is to remove high-frequency perturbations that may have been introduced by adversarial attacks while preserving the essential low-frequency structural features. However, since GCN-SVD, tailored specifically for Nettack [2], focuses on preprocessing the poisoned graph data, it may not offer consistent efficacy against diverse adversarial attacks. A recent line of work focuses

on designing GNN model architectures that are robust against adversarial attacks [21–25, 56–58]. For instance, Mid-GCN [25] introduces a mid-pass graph filter based on the observation that the eigenvalues of the graph Laplacian within the mid-frequency range are less susceptible to adversarial attacks. However, the mid-frequency information captured by this filter may not sufficiently preserve important structural features of the graph, leading to degraded performance. In addition, the training of Mid-GCN exhibits instability, particularly evident as the spectral radius of its propagation matrix increases with network depth.

In this thesis, we introduce a novel and robust model, named fixed-point iterative graph convolutional network (Fix-GCN), designed to withstand diverse adversarial attacks under varying perturbation levels. The core message-passing mechanism of Fix-GCN is derived by solving a graph filtering system using fixed-point iteration [59]. Our proposed network, belonging to the category of resilient architectures, aims to withstand adversarial perturbations by designing a flexible-pass, higher-order filter that selectively attenuates high-frequency components while preserving low-frequency structural information in the graph signal. This selective attenuation of high-frequency components ensures that the model remains robust and resilient, even when faced with adversarial manipulation. Moreover, by capturing information from higher-order neighbors, Fix-GCN can mitigate the risk posed by direct perturbations on 1-hop neighbors of target nodes, which are often more effective than indirect perturbations (i.e., influencer attacks) on multi-hop neighbors. In addition, similar to the GCN-SVD defense method, our approach with the spectral modulation filter operates on the principle of selectively preserving low-frequency components and discarding high-frequency ones in the graph signal to defend against adversarial attacks while preserving essential graph structural information. Our key contributions can be summarized as follows:

- We propose a novel spectral modulation filter that provides a mechanism to selectively attenuate high-frequency components while preserving low-frequency structural information in the graph signal.
- We present a robust graph convolutional network architecture with an aggregation mechanism that captures information from higher-order neighbors of graph nodes, while maintaining computational efficiency.
- Experimental results demonstrate the robustness of the proposed model against adversarial attacks, outperforming competitive baselines across various benchmark datasets.

The remainder of this chapter is organized as follows. Section 2.2 presents the methodology, including the problem formulation, propagation rule, model architecture, and model training and prediction. The experimental setup and results are presented in Section 2.3.

2.2 Proposed Method

2.2.1 Preliminaries and Problem Formulation

Basic Notions. An attributed graph is a type of graph data structure where each node in the graph is associated with attributes or features. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an attributed graph, where $\mathcal{V} = \{1, \ldots, N\}$ is the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)^\intercal$ an $N \times F$ feature matrix of node attributes (i.e., $\mathbf{x}_i$ is an F-dimensional row vector for node i). We denote by $\mathbf{A}$ an $N \times N$ adjacency matrix whose (i, j)-th entry is equal to 1 if i and j are neighboring nodes, and 0 otherwise. We also denote by $\mathbf{L} = \mathbf{I} - \hat{\mathbf{A}}$ the normalized Laplacian matrix, where $\hat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is the normalized adjacency matrix, $\mathbf{D} = \mathrm{diag}(\mathbf{A}\mathbf{1})$ is the diagonal degree matrix, and $\mathbf{1}$ is an N-dimensional vector of all ones. Since the normalized Laplacian matrix is symmetric positive semi-definite, it admits an eigendecomposition given by $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\intercal$, where $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_N)$ is an orthonormal matrix whose columns constitute an orthonormal basis of eigenvectors and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ is a diagonal matrix comprised of the corresponding eigenvalues such that $0 = \lambda_1 \leq \cdots \leq \lambda_N \leq 2$ in increasing order [60].

Problem Statement. Semi-supervised learning in a graph involves predicting the labels of nodes that are not labeled, based on the labels of a small subset of nodes. Specifically, let $\mathcal{V}_l \subset \mathcal{V}$ be the set of N_l labeled nodes in $\mathcal{V}$ with associated ground-truth labels in the label set $\mathcal{Y}_l = \{\mathbf{y}_1, \ldots, \mathbf{y}_{N_l}\}$, where $\mathbf{y}_i \in \{0, 1\}^C$ is the one-hot encoding vector of node i and C is the total number of classes. Let $\mathcal{V}_u \subset \mathcal{V} \setminus \mathcal{V}_l$ be the set of N_u unlabeled nodes, where $N_l + N_u = N$ and $N_l \ll N_u$. The goal of semi-supervised node classification is to learn the parameters $\boldsymbol{\theta}$ of a graph representation learning (GRL) prediction model $f_{\boldsymbol{\theta}} : \mathcal{V}_l \to \mathcal{Y}_l$. This is usually done by minimizing the categorical cross-entropy loss function over the set of labeled nodes

$$\min_{\boldsymbol{\theta}} \sum_{i \in \mathcal{V}_l} \mathcal{C}(\mathbf{y}_i, \hat{\mathbf{y}}_i) = -\sum_{i \in \mathcal{V}_l} \sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}), \tag{2.1}$$

where $\mathbf{y}_i \in \mathcal{Y}_l$ is the one-hot encoding vector of node i, $\hat{\mathbf{y}}_i = f_{\boldsymbol{\theta}}(i)$ is the vector of predicted probabilities of node i, and $\mathcal{C}(\mathbf{y}_i, \hat{\mathbf{y}}_i)$ is the categorical cross-entropy loss for the ith node. Here, y_{ic} is the indicator that the ith node belongs to the cth class, while $\hat{y}_{ic}$ is the predicted probability that the model associates the ith node with class c.

Adversarial attacks on GRL models can be carried out through various means, including perturbing the graph structure, node features, or a combination of both. To undermine the performance of a GRL model, an adversarial attacker manipulates the edges and/or node features in the original graph $\mathcal{G}$, resulting in perturbed graphs $\mathcal{G}' = (\mathcal{V}, \mathcal{E}', \mathbf{X})$ or $\mathcal{G}' = (\mathcal{V}, \mathcal{E}', \mathbf{X}')$, where $\mathbf{A}'$ denotes the

adjacency matrix of the perturbed graph. In the context of these attacks, $\mathbf{X}$ represents the original node features, and $\mathbf{X}'$ represents the manipulated node features. The attacker has the flexibility to target the graph structure (edges) and/or the node features to create perturbations that can deceive the GRL model into making incorrect predictions. Hence, it is crucial to develop a robust GRL model to counter such attacks. Then, the learned robust model is employed to predict the labels of the nodes in the set $\mathcal{V}_u$.

2.2.2 Spectral Modulation Filtering

Spectral graph filtering employs filters defined as functions of the graph normalized Laplacian (or its eigenvalues). The goal of these filters, often referred to as frequency responses or transfer functions, is to reduce or eliminate high-frequency noise in the graph signal. These functions basically describe how a filter affects the input graph signal to produce the output graph signal. We define a spectral modulation filter as follows:

$$h_s(\lambda) = \frac{1}{(1+s)\lambda - s\lambda^2}, \tag{2.2}$$

where $s \in (0, 1)$ is a positive scaling parameter that allows for the modulation or adjustment of the spectral characteristics, indicating its capability to control the filtering effect on different frequency components of the graph signal. The filter h_s is a rational polynomial function of the eigenvalues of the normalized Laplacian matrix. It is a flexible-pass filter in the sense that it exhibits low-pass characteristics, as it allows low-frequency components (corresponding to small eigenvalues) to flexibly pass through with little attenuation, while attenuating high-frequency components (associated with large eigenvalues). As shown in Figure 2.1, the attenuation behavior of the filter is determined by the scalar s, which serves as a tuning or modulation parameter. By adjusting s, we can control the trade-off between preserving low-frequency structural information and reducing high-frequency noise or variations in the graph signal. When s is large, the filter is less selective, allowing a wider range of frequencies to pass through with less attenuation. As s decreases, the filter becomes more selective and significantly attenuates higher frequencies, effectively filtering out more of the high-frequency noise or variations in the graph signal.

Graph Filtering System. Applying the spectral modulation filter on the graph signal $\mathbf{X} \in \mathbb{R}^{N \times F}$ yields a filtered graph signal $\mathbf{H}$ given by

$$\mathbf{H} = h_s(\mathbf{L})\mathbf{X} = ((1+s)\mathbf{L} - s\mathbf{L}^2)^{-1}\mathbf{X}, \tag{2.3}$$

which can be rewritten as

$$((1+s)\mathbf{L} - s\mathbf{L}^2)\mathbf{H} = \mathbf{X}, \tag{2.4}$$

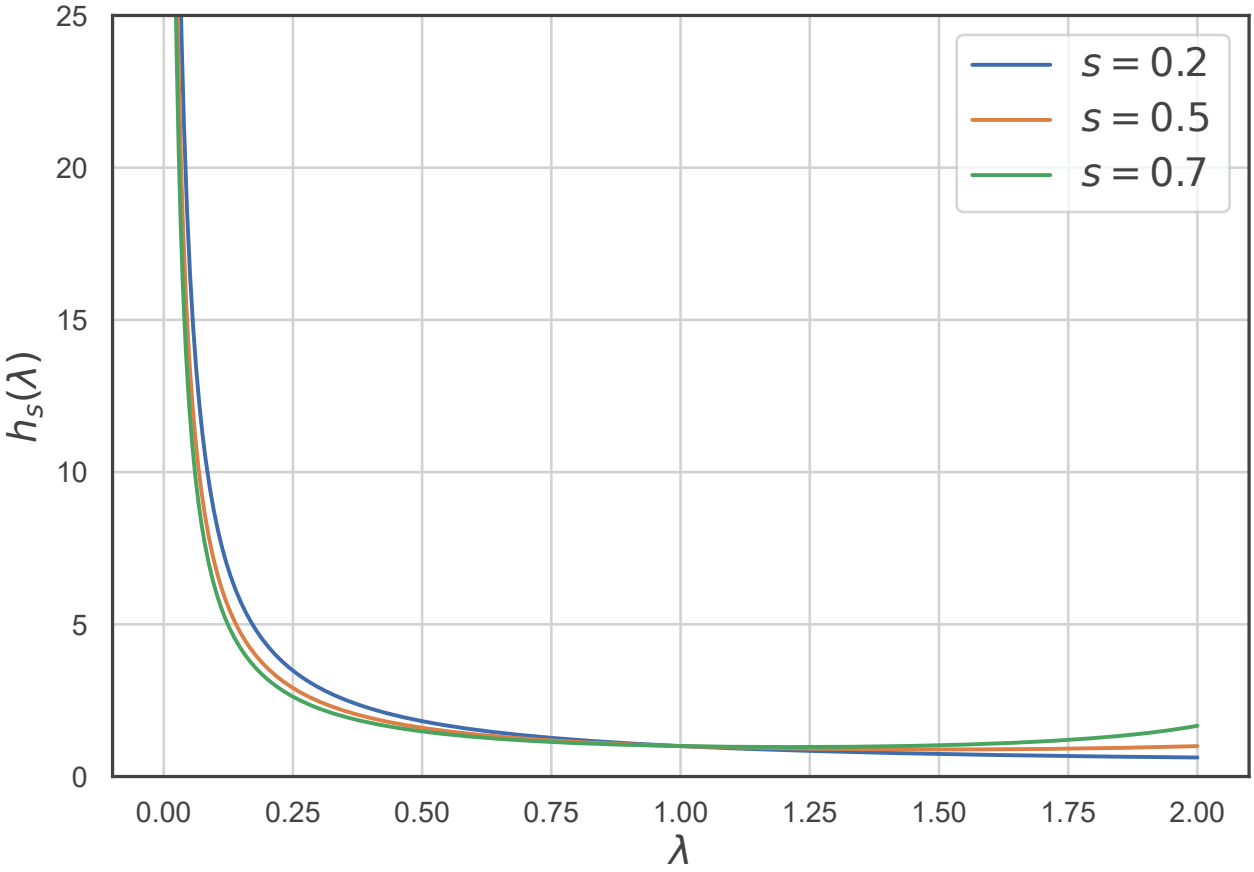


Figure 2.1: Transfer function of the spectral modulation filter. Lower values of the scaling parameter make the filter attenuate high-frequency components more strongly.

or equivalently

$$\begin{aligned} \mathbf{H} &= (\mathbf{I} - (1+s)\mathbf{L} + s\mathbf{L}^2)\mathbf{H} + \mathbf{X} \\ &= (\mathbf{I} - s\mathbf{L})(\mathbf{I} - \mathbf{L})\mathbf{H} + \mathbf{X}. \end{aligned} \tag{2.5}$$

Since $\mathbf{L} = \mathbf{I} - \hat{\mathbf{A}}$, the spectral modulation filter equation becomes

$$\mathbf{H} = ((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}}\mathbf{H} + \mathbf{X}, \tag{2.6}$$

which can be solved using, for instance, the fixed point iteration method [59], where $\mathbf{H} = \varphi(\mathbf{H})$ with the function φ defined by the right-hand side term of Eq. (2.6).

Fixed Point Iterative Solution. Fixed-point iteration is an iterative numerical method used to find a fixed point of a given function [59]. The process involves repeatedly applying the function to an initial guess or estimate and updating this estimate in each iteration until it converges to the fixed point. For the spectral modulation filter equation $\mathbf{H} = \varphi(\mathbf{H})$, The fixed point iterative solution is given by

$$\mathbf{H}^{(t+1)} = ((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}}\mathbf{H}^{(t)} + \mathbf{X}, \tag{2.7}$$

with some initial guess $\mathbf{H}^{(0)}$, and $t \in \mathbb{N}$ denotes the iteration number.

2.2.3 Fixed-Point Iterative Graph Neural Network

At the core of graph neural networks is the concept of feature propagation rule, which determines how information is passed between nodes in a graph. It involves updating the current node features by aggregating information from their immediate and high-order neighboring nodes, followed by a non-linear activation function to produce an updated representation for the node. Inspired by the fixed point iterative solution of the spectral modulation filter equation, we propose a fixed-point graph convolutional network (Fix-GCN) with the following layer-wise update rule for node feature propagation:

$$\mathbf{H}^{(\ell+1)} = \sigma\Big(((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}}\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{X}\widetilde{\mathbf{W}}^{(\ell)}\Big), \tag{2.8}$$

where $\mathbf{W}^{(\ell)}$ and $\widetilde{\mathbf{W}}^{(\ell)}$ are learnable weight matrices, $\sigma(\cdot)$ is an element-wise activation function, $\mathbf{H}^{(\ell)} \in \mathbb{R}^{N\times F_\ell}$ is the input feature matrix of the ℓ-th layer with F_ℓ feature maps for $\ell = 0, \ldots, L-1$. The input of the first layer is the initial feature matrix $\mathbf{H}^{(0)} = \mathbf{X}$. It is worth pointing out that the key difference between Eq. (2.7) and Eq. (3.1) is that the latter defines a representation updating rule for propagating node features layer-wise using trainable weight matrices for learning an efficient representation of the graph, followed by an activation function to introduce non-linearity into the network in a bid to enhance its expressive power.

The update rule of Fix-GCN is essentially comprised of three main components: (i) feature propagation that combines the features of the 1- and 2-hop neighbors of nodes (i.e., it aggregates information from immediate and high-order neighboring nodes), (ii) feature transformation that applies learnable weight matrices to the node representations to learn an efficient representation of the graph, and (iii) residual connection for ensuring that information from the initial feature matrix is preserved. The initial residual connection used in the proposed model allows information from the initial feature matrix to bypass the current layer and be directly added to the output of the current layer. This helps preserve important information that may be lost during the aggregation process, thereby improving the flow of information through the network. In other words, in addition to performing a second-order graph convolution, the update rule of Fix-GCN applies an initial residual connection that reuses the initial node features. Note that the propagation operation or matrix $\mathbf{P} = ((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}}$ of the proposed GNN is a weighted combination of the normalized adjacency matrix and its square. It allows Fix-GCN to capture information from nodes that are not only directly connected (1-hop), but also incorporates information from the neighbors of the neighbors (2-hop). The parameter s helps control the balance between the information from immediate neighbors and the information from nodes that are at most two edges away in the graph. This is

particularly valuable for learning graph representations that capture more global information and dependencies.

Connection to GCN. The hyper-parameter s in Fix-GCN provides control over the network's behavior. When $s = 0$, Fix-GCN reduces to the standard GCN with initial residual connections, which basically decouples the transformations for the self-connections and the 1-hop neighbors, essentially incorporating residual connections at the initial stage of the graph convolution process. This process operates on the normalized adjacency matrix without self-connections. Similarly, when $s = 1$, Fix-GCN corresponds to the second-order GCN with initial residual connections. So, by varying the value of s between 0 and 1, we can smoothly control the behavior of Fix-GCN, allowing it to capture different orders of information from the graph structure. Therefore, Fix-GCN provides a flexible framework for adapting to various graph-based tasks and the level of emphasis on local (first-order) and non-local (second-order) information in the graph structure.

Model Complexity. For simplicity, we assume the feature dimensions are the same across all layers, i.e., $F_\ell = F$ for all ℓ, with $F \ll N$. Multiplying the propagation matrix $((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}}$ with an embedding $\mathbf{H}^{(\ell)}$ costs $\mathcal{O}(\|\hat{\mathbf{A}}\|_0 F)$ in time, where $\|\hat{\mathbf{A}}\|_0$ denotes the number of non-zero entries of the sparse matrix $\hat{\mathbf{A}}$ (i.e., number of edges in the graph). Multiplying an embedding with a weight matrix costs $\mathcal{O}(NF^2)$. Also, multiplying the initial feature matrix by the residual connection weight matrix costs $\mathcal{O}(NF^2)$. Hence, the time complexity of an L-layer Fix-GCN is $\mathcal{O}(L\|\hat{\mathbf{A}}\|_0 F + LNF^2)$.

For memory complexity, an L-layer Fix-GCN requires $\mathcal{O}(LNF + LF^2)$ in memory, where $\mathcal{O}(LNF)$ is for storing all embeddings and $\mathcal{O}(LF^2)$ is for storing all layer-wise weight matrices. Therefore, our proposed Fix-GCN model has the same time and memory complexity as that of GCN, albeit Fix-GCN takes into account both immediate and distant graph nodes for improved learned node representations. It is important to note that there is no need to explicitly compute the square of the normalized adjacency matrix in the Fix-GCN model. Instead, we perform right-to-left multiplication of the normalized adjacency matrix with the embedding. This process avoids the computational cost associated with matrix exponentiation and simplifies the computation, making our model more efficient while achieving its objectives.

2.2.4 Numerical Stability of Fix-GCN

In order to demonstrate the numerical stability of the proposed Fix-GCN model, we start with a useful result in matrix analysis, which states that the spectral radius of the sum of two commuting matrices is bounded by the sum of the individual matrices.

Lemma 1. *If two matrices* M_1 *and* M_2 *commute, i.e.,* $\mathrm{M}_1\mathrm{M}_2 = \mathrm{M}_2\mathrm{M}_1$*, then*

$$\rho(\mathrm{M}_1 + \mathrm{M}_2) \leq \rho(\mathrm{M}_1) + \rho(\mathrm{M}_2),$$

where $\rho(\cdot)$ *denotes matrix spectral radius (i.e., largest absolute value of all eigenvalues).*

Since the eigenvalues of the normalized Laplacian matrix $\mathbf{L} = \mathbf{I} - \hat{\mathbf{A}}$ lie in the interval $[0, 2]$, it follows that $\rho(\hat{\mathbf{A}}) \leq 1$. Hence, we have the following result, which demonstrates the training stability of the proposed model, with information smoothly propagating through the graph layers without amplifying or dampening effects that could lead to instability.

Proposition 2. *The update rule of Fix-GCN is numerically stable.*

Proof. Recall that the propagation matrix of Fix-GCN is given by

$$\mathbf{P} = ((1-s)\mathbf{I} + s\hat{\mathbf{A}})\hat{\mathbf{A}} = (1-s)\hat{\mathbf{A}} + s\hat{\mathbf{A}}^2.$$

Since the matrices $(1-s)\hat{\mathbf{A}}$ and $s\hat{\mathbf{A}}^2$ satisfy the assumptions of Lemma 1, we have

$$\rho((1-s)\hat{\mathbf{A}} + s\hat{\mathbf{A}}^2) \leq \rho((1-s)\hat{\mathbf{A}}) + \rho(s\hat{\mathbf{A}}^2) \leq 1,$$

because both $\rho(\hat{\mathbf{A}})$ and $\rho(\hat{\mathbf{A}}^2)$ are bounded by 1. Hence, the spectral radius of the propagation matrix is bounded by 1. Consequently, repeated layer-wise application of this propagation operator is stable.

2.3 Experiments

In this section, we conduct experimental evaluations of the proposed model, comparing it with state-of-the-art methods. We begin by outlining the experimental setup, followed by providing an overview of datasets, baseline methods and implementation details. Then, we present both quantitative and qualitative results on benchmark datasets using various evaluation metrics. We also conduct parameter sensitivity analysis on the significance of various components in our model in an effort to provide valuable insight into the effectiveness of the model. The source code is available at: https://github.com/Shakib-IO/Fix-GCN

2.3.1 Experimental Setup

Datasets. We assess the performance of our proposed method on five benchmark datasets: GitHub [61], Cora-ML [62], and citation networks (Cora, CiteSeer, PubMed) [63]. Dataset statistics are summarized in Table 2.1, where only the largest connected component is considered.

Table 2.1: Summary statistics of benchmark graph datasets. We only consider the largest connected component in these adversarial graphs.

Datasets	#Nodes	#Edges	#Features	#Classes
Cora	2,485	5,069	1,433	7
CiteSeer	2,110	3,668	3,703	6
Cora-ML	2,995	4,208	2,879	5
GitHub	3,150	71,310	4,005	2
PubMed	19,717	44,338	500	3

Baseline Methods. We evaluate the performance of our model against comparative GNN models and state-of-the-art adversarial defense methods, including GCN [47], GAT [16], GCN-Jaccard [5], GCN-SVD [17], RGCN [4], Pro-GNN [18], and Mid-GCN [25].

Implementation Details. All experiments are conducted on a Linux machine with a single NVIDIA GeForce RTX 3070 GPU featuring 8GB of memory. For fair comparison, we run experiments following the setup and default settings of the baselines, including the data split for semi-supervised learning. For each dataset, we randomly assign 10% of the nodes to the labeled training set, 10% of the nodes to the validation set, and the remaining 80% of the nodes to the test set. We use PyTorch to implement our two-layer model with a hidden dimension of 64, and train it for 200 epochs using the Adam optimizer [64] with a learning rate of 1e-2 and a weight decay rate of 5e-4. The default dropout ratio is 0.6 for all datasets, and the filter hyperparameter $s = 0.2$ is determined via grid search.

2.3.2 Results and Analysis

We evaluate the node classification performance of Fix-GCN against various types of adversarial attacks, including non-targeted attacks, targeted attacks, random attacks, and feature attacks.

Robustness Against Non-targeted Attacks. The goal of non-targeted attacks is to degrade the overall performance of the mode. We adopt Mettack [3] as a non-targeted attack to perturb the graph structure with the aim of compromising the model's performance in node classification. Specifically, we evaluate the robustness of our model against non-targeted adversarial attacks using different perturbation rates, spanning from 0% to 25% in increments of 5%. The node classification results of Fix-GCN and baseline methods are summarized in Table 2.2, where both the average accuracy and standard deviation are reported over 10 runs. The best results are in bold and the second best ones are underlined. The results presented in Table 2.2 show that, for the vast majority of cases, our model consistently surpasses all baselines across all datasets, with notable

performance improvements observed, especially at higher perturbation rates. At 25% perturbation rate, our Fix-GCN model yields relative improvements of 6.4%, 1.98%, 2.97% and 11.73% over Mid-GCN on Cora, CiteSeer, GitHub and PubMed, respectively, with the highest relative improvement of 13.23% achieved on Cora-ML. Interestingly, our model outperforms Mid-GCN under all perturbation rates on the dense GitHub dataset, which has the highest number of edges among the five graph benchmarks.

Table 2.2: Node classification performance of Fix-GCN and baselines under non-targeted attacks (Mettack) with different perturbation rates P(%). We report the average accuracy over 10 runs, along with the corresponding standard deviation. The best results are in **bold** and the second best ones are underlined.

Dataset	P(%)	GCN	GAT	GCN-Jaccard	GCN-SVD	RGCN	Pro-GNN	Mid-GCN	**Fix-GCN**
Cora	0	83.50±0.44	83.97±0.69	82.05±0.51	80.63±0.45	83.09±0.44	**85.39±0.81**	84.61±0.46	84.80±0.33
	5	76.55±0.79	80.44±0.74	79.13±0.59	78.93±0.53	77.42±0.39	82.78±0.39	**82.94±0.46**	82.19±0.27
	10	70.39±1.28	75.61±0.59	75.16±0.76	71.47±0.83	72.22±0.38	79.03±0.59	80.14±0.86	**82.64±0.58**
	15	65.10±0.71	69.78±1.28	71.03±0.64	66.69±1.18	66.82±0.39	76.40±1.27	77.77±0.75	**80.58±0.81**
	20	59.56±0.92	59.54±0.92	65.71±0.89	58.94±1.13	59.27±0.37	73.32±1.56	76.58±0.29	**79.27±0.55**
	25	47.53±1.96	54.78±0.74	60.82±1.08	52.06±1.19	50.51±0.78	69.72±1.69	72.89±0.81	**77.56±0.94**
CiteSeer	0	71.96±0.55	73.26±0.83	72.10±0.63	70.65±0.32	71.20±0.83	73.28±0.69	**74.17±0.28**	73.68±0.31
	5	70.88±0.62	72.89±0.83	70.51±0.97	68.84±0.72	70.50±0.43	73.09±0.34	**74.31±0.42**	74.03±0.22
	10	67.55±0.89	70.63±0.48	69.54±0.56	68.87±0.62	67.71±0.30	72.51±0.75	73.59±0.29	**74.27±0.31**
	15	64.52±1.11	69.02±1.09	65.95±0.94	63.26±0.96	65.69±0.37	72.03±1.11	73.69±0.29	**73.82±1.02**
	20	62.03±3.49	61.04±1.52	59.30±1.40	58.55±1.09	62.49±1.22	70.02±2.28	71.51±0.83	**72.80±0.47**
	25	56.94±2.09	61.85±1.12	59.80±1.47	57.18±1.87	55.35±0.66	68.95±2.78	69.12±0.72	**70.49±0.69**
Cora-ML	0	85.85±0.30	83.45±1.46	80.22±1.54	83.87±1.53	82.39±1.71	85.38±1.14	86.56±0.28	**86.78±0.56**
	5	79.76±1.44	79.11±1.69	79.75±1.78	80.29±1.89	80.13±1.91	**80.38±1.98**	80.20±1.68	80.07±0.66
	10	74.64±0.67	79.36±1.66	74.33±1.79	79.08±1.73	74.55±1.91	**79.48±1.23**	79.30±0.96	79.17±0.82
	15	53.74±1.09	61.22±1.44	57.38±1.09	74.95±1.58	55.42±1.25	53.60±1.18	73.32±0.66	**76.95±1.42**
	20	45.24±1.88	52.72±1.29	47.15±1.09	47.95±1.76	47.68±1.32	47.37±1.34	60.92±1.43	**75.17±1.24**
	25	48.80±1.91	54.26±1.98	49.42±1.56	56.85±1.99	50.62±1.01	50.52±1.12	67.18±1.35	**76.09±0.67**
GitHub	0	72.92±0.13	72.81±0.12	72.93±0.56	73.31±0.15	73.16±0.19	73.34±0.34	79.51±0.67	**81.39±0.23**
	5	72.81±0.07	72.43±1.13	71.85±2.18	72.91±0.12	73.09±0.31	72.89±0.07	81.87±1.46	**82.59±0.42**
	10	72.61±0.57	72.97±0.13	72.63±0.98	72.78±0.09	73.06±0.16	72.75±0.18	81.23±1.67	**82.45±0.35**
	15	72.97±0.11	72.97±0.06	72.79±0.51	72.97±1.13	73.22±0.21	72.98±0.09	80.48±0.25	**82.75±0.26**
	20	72.11±2.27	70.42±2.12	72.24±1.96	72.47±0.14	73.10±0.21	72.98±0.13	81.08±0.96	**82.62±0.28**
	25	72.74±0.41	72.97±1.16	72.32±1.73	72.14±0.07	72.91±0.24	72.56±0.21	80.37±1.51	**82.76±0.30**
PubMed	0	87.19±0.09	83.73±0.40	87.06±0.09	83.44±0.21	86.16±0.18	87.33±0.18	85.67±0.37	**88.26±0.27**
	5	83.09±0.13	78.00±0.44	86.39±0.06	83.41±0.15	81.08±0.20	87.25±0.09	83.48±0.10	**87.33±0.38**
	10	81.21±0.09	74.93±0.38	85.70±0.07	83.27±0.21	77.51±0.27	**87.25±0.09**	81.43±0.43	86.77±0.25
	15	78.66±0.12	71.13±0.51	84.76±0.08	83.10±0.18	73.91±0.25	**87.20±0.09**	79.74±0.14	86.33±0.12
	20	77.35±0.19	68.21±0.96	83.01±0.22	83.88±0.05	71.18±0.31	87.15±0.15	78.69±0.32	**87.28±0.21**
	25	75.50±0.17	65.41±0.77	83.66±0.06	82.72±0.18	67.95±0.15	86.76±0.19	77.81±0.34	**86.94±0.16**

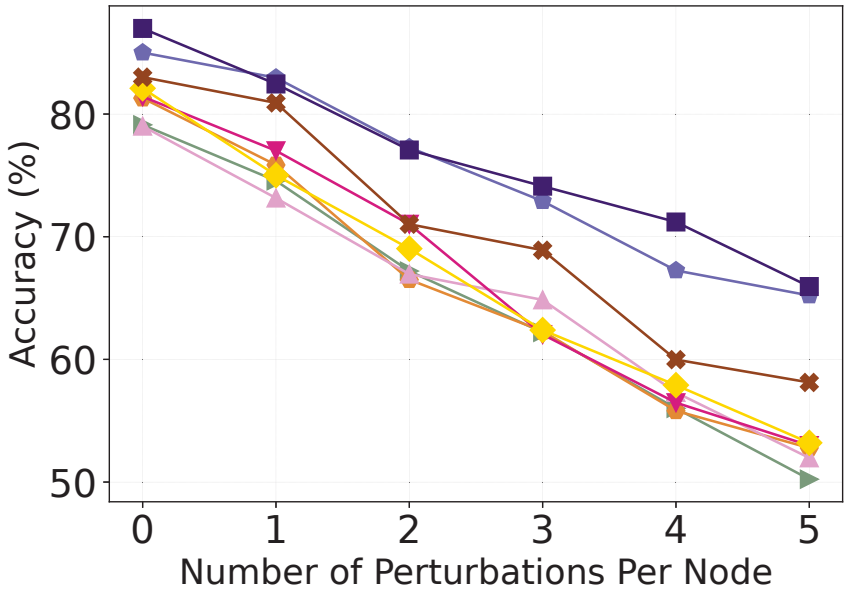
Robustness Against Targeted Adversarial Attacks. Unlike non-targeted attacks, which degrade the overall performance of the model, targeted attacks aim to cause the model to produce incorrect predictions for specific nodes. For instance, a targeted attack might involve perturbing the graph structure in a way that causes the model to misclassify a particular node as belonging to a specific class, even if it does not naturally belong to that class. We adopt Nettack [2] as a

targeted adversarial attack method, which iteratively perturb the graph structure by removing or adding edges in a way that maximally changes the predictions of the GNN model for the target nodes. For this attack, consistent with prior work [4, 18], we change the number of perturbations made on each targeted node from 1 to 5, incrementally increasing by 1. Nodes in the test set with a degree exceeding 10 are designated as target nodes. Since the GitHub dataset exhibits a relatively high level of density, we opt to select only 8% of the nodes for attacks. Similarly, for the PubMed dataset, we only use 10% of the nodes in our analysis to prevent potential lengthy execution times associated with Nettack. The defense performance results (i.e., multi-class classification accuracy) against targeted attacks are depicted in Figure 2.2, 2.3, which shows that our model consistently outperforms the baseline methods under the varying numbers of perturbations on the target nodes across all the datasets. The main findings from Figure 2.2, 2.3 are summarized as follows:

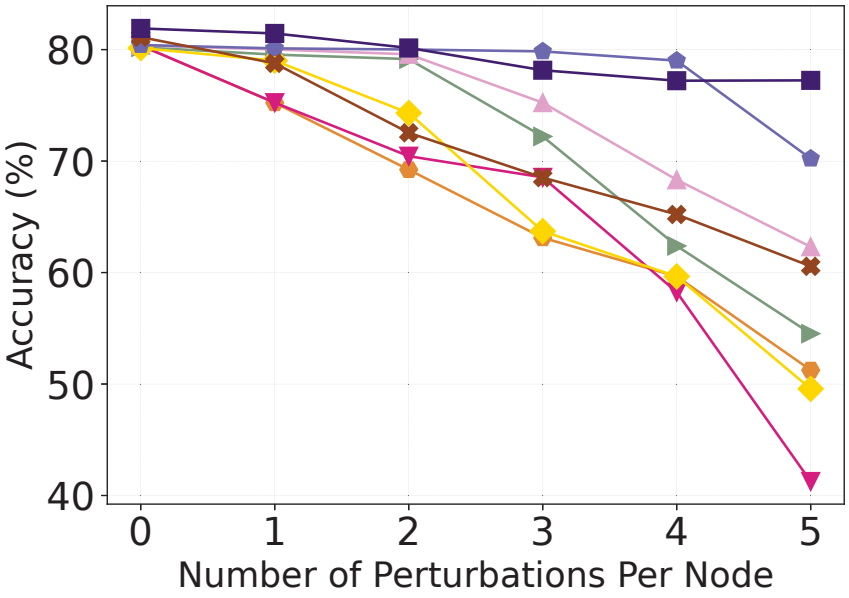
- When compared to the strongest baselines, Mid-GCN and Pro-GNN, our Fix-GCN model consistently achieves superior performance on the Cora dataset across all levels of perturbations. Similarly, on the CiteSeer dataset, Fix-GCN outperforms Mid-GCN in most cases, especially as the number of perturbations increases. Moreover, it is noteworthy that GCN-SVD, despite being tailored for Nettack, exhibits a significant drop in performance, particularly at higher levels of perturbations.

- On the GitHub dataset, known for its denser nature, our model, alongside the strongest baselines, Mid-GCN and Pro-GNN, demonstrates consistent performance across all levels of perturbations. Also, GCN-SVD maintains a stable performance due to its specific design for Nettack. A similar trend is observed on the PubMed dataset, albeit with a slight decline in performance, particularly at higher levels of perturbations. Interestingly, GCN-Jaccard demonstrates superior performance at higher levels of perturbations, showcasing the effectiveness of its two-stage approach. However, it is important to note that this approach involves preprocessing the input graph by dropping dissimilar edges based on the Jaccard similarity metric before training GCN on the processed graph. Notably, without this preprocessing step, the GCN baseline experiences a significant drop in performance as the number of perturbations increases.

- By combining the strengths of a robust aggregation mechanism that captures information from higher-order neighbors of graph nodes and a flexible-pass filtering approach that preserves low-frequency structural information in the graph signal, our Fix-GCN model outperforms all the baselines in most cases.

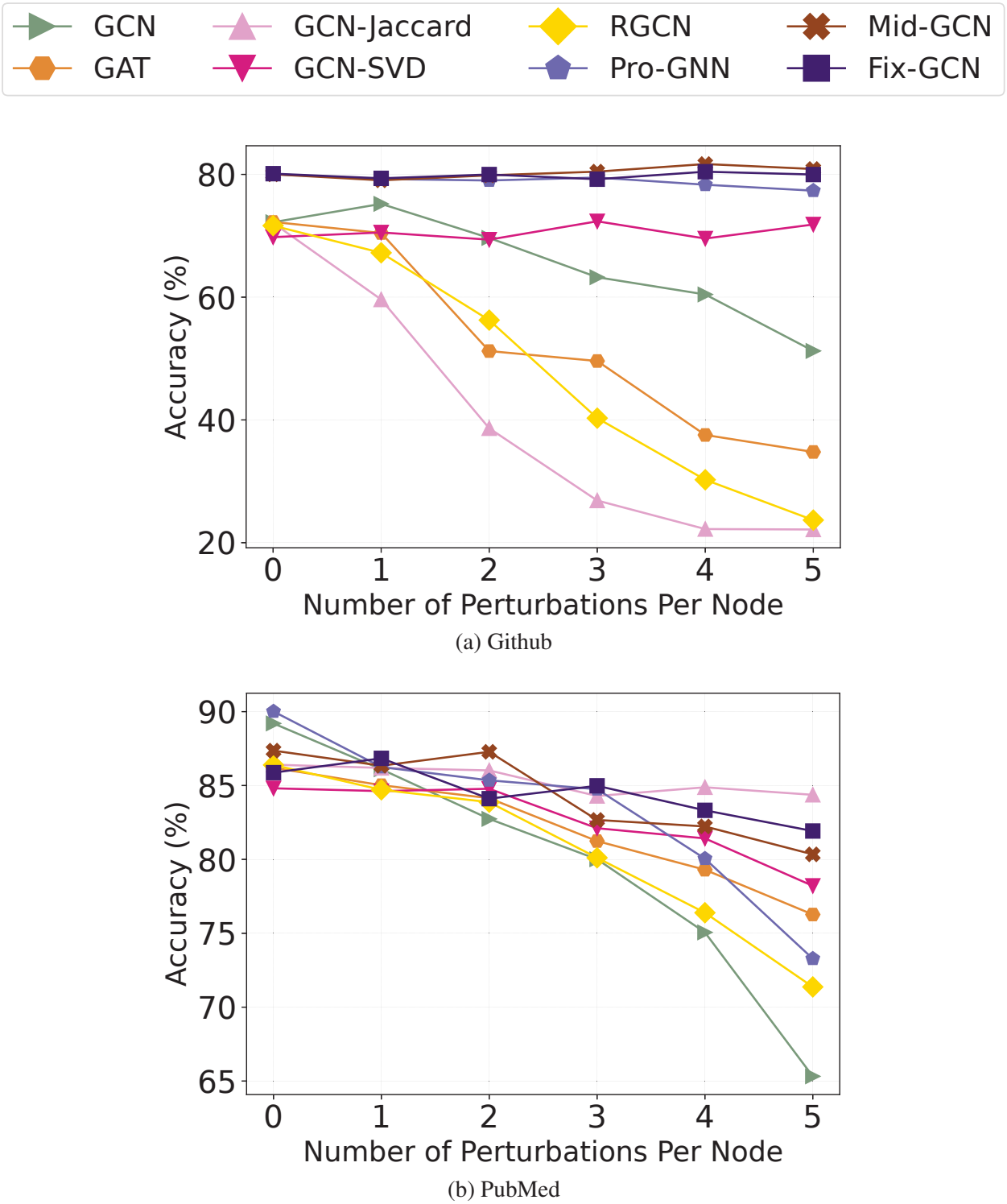


(a) Cora



(b) CiteSeer

Figure 2.2: Node classification accuracy on Cora and CiteSeer dataset under targeted attacks (Nettack) with varying numbers of perturbations on the target nodes $\{1, 2, 3, 4, 5\}$.



(a) Github

(b) PubMed

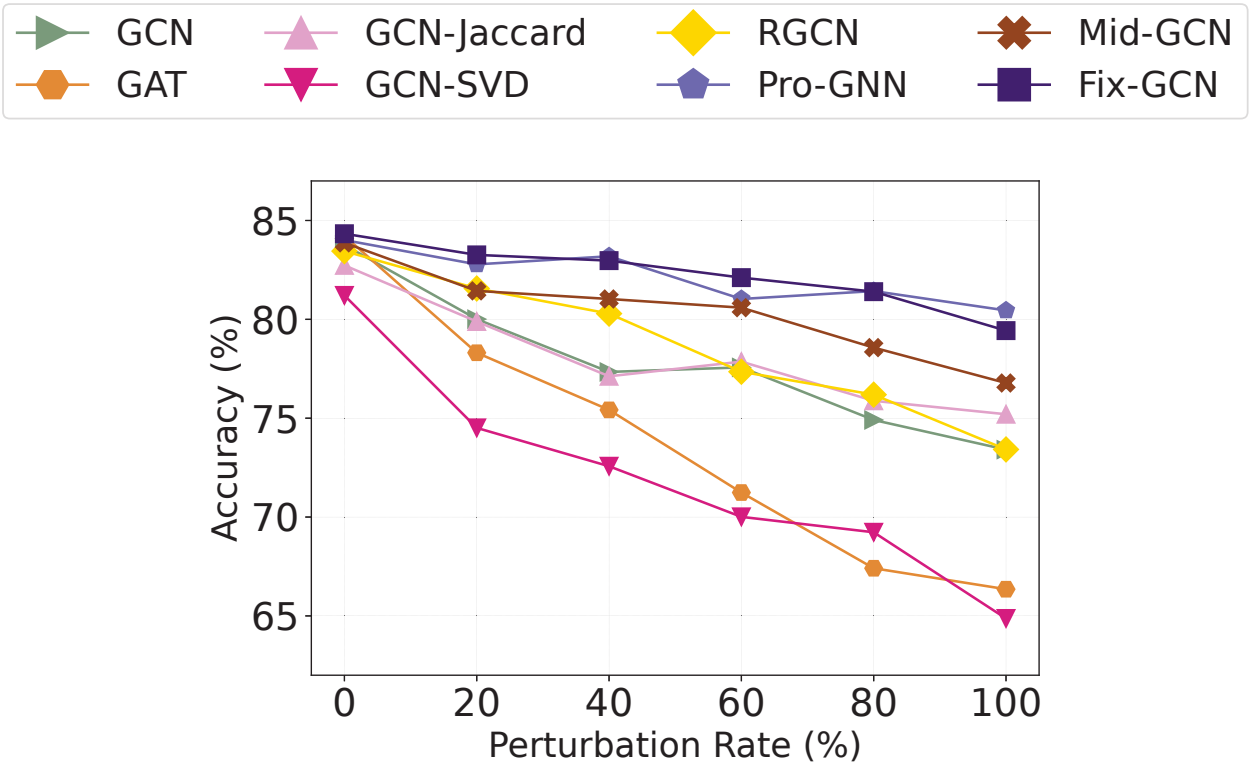
Figure 2.3: Node classification accuracy on Github and PubMed dataset under targeted attacks (Nettack) with varying numbers of perturbations on the target nodes $\{1, 2, 3, 4, 5\}$.

These results underscore the robustness and effectiveness of our Fix-GCN model in defending against adversarial attacks across various settings and datasets.

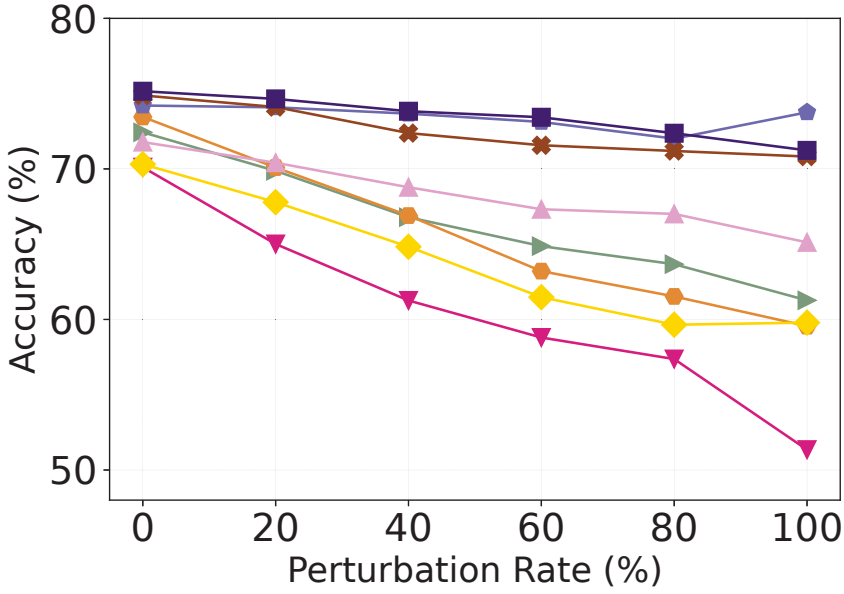
Robustness Against Random Attacks. The objective of random attacks is to introduce randomness by adding edges to the input graph, akin to injecting random noise into the clean graph.

In our experiment, we assess the performance of our Fix-GCN model and baseline methods under varying ratios of random attack, ranging from 0% to 100% of the number of edges in the true adjacency matrix, with increments of 20%. The results, reported in Figure 2.4, show that Fix-GCN, along with Mid-GCN and Pro-GNN, maintains relatively stable performance across all perturbation rates on Cora and CiteSeer datasets. Notably, Fix-GCN exhibits superior results on most perturbation rates for both datasets, except at the highest 100% rate. In contrast, the other comparative methods experience significant performance degradation as the perturbation rate increases.

Robustness Against Feature Attacks. We assess the influence of random attacks on the efficacy of our Fix-GCN model by randomly perturbing the initial node feature matrix. Besides structural alterations to graphs, feature attacks represent a crucial aspect of adversarial attacks since node features are extensively leveraged by GCN-based methods in their message-passing mechanisms. Given that the features of Cora, CiteSeer, and GitHub datasets are exclusively comprised of 0s and 1s, we introduce feature attacks by randomly flipping the 0/1 values [58]. For instance, the Cora and CiteSeer graphs are composed of nodes representing scientific publications and edges representing citation links between these publications. Each node is described by a binary feature vector indicating the presence or absence of words from a dictionary. The results, as illustrated in Figure 2.5, demonstrate that Fix-GCN consistently outperforms all baseline methods across all perturbation rates, particularly at higher rates, on both datasets. Interestingly, both Mid-GCN and Pro-GNN experience significant performance drops at higher levels of perturbation rates. A more pronounced decline in performance is observed for GCN-Jaccard on both datasets. In contrast, our Fix-GCN model demonstrates robust defense against feature attacks. This strong resilience against such attacks is largely attributed to the fact that Fix-GCN integrates an initial residual connection in its feature propagation scheme by design. The initial residual connection in the proposed model serves the crucial function of preserving information from the initial feature matrix throughout the aggregation process. By allowing the initial features to directly contribute to the output of each layer, the residual connection ensures that important information is retained, even in the face of adversarial perturbations targeting the node features. In essence, the initial residual connection acts as a safeguard against feature manipulation, enhancing the model's resilience to adversarial attacks on node features.



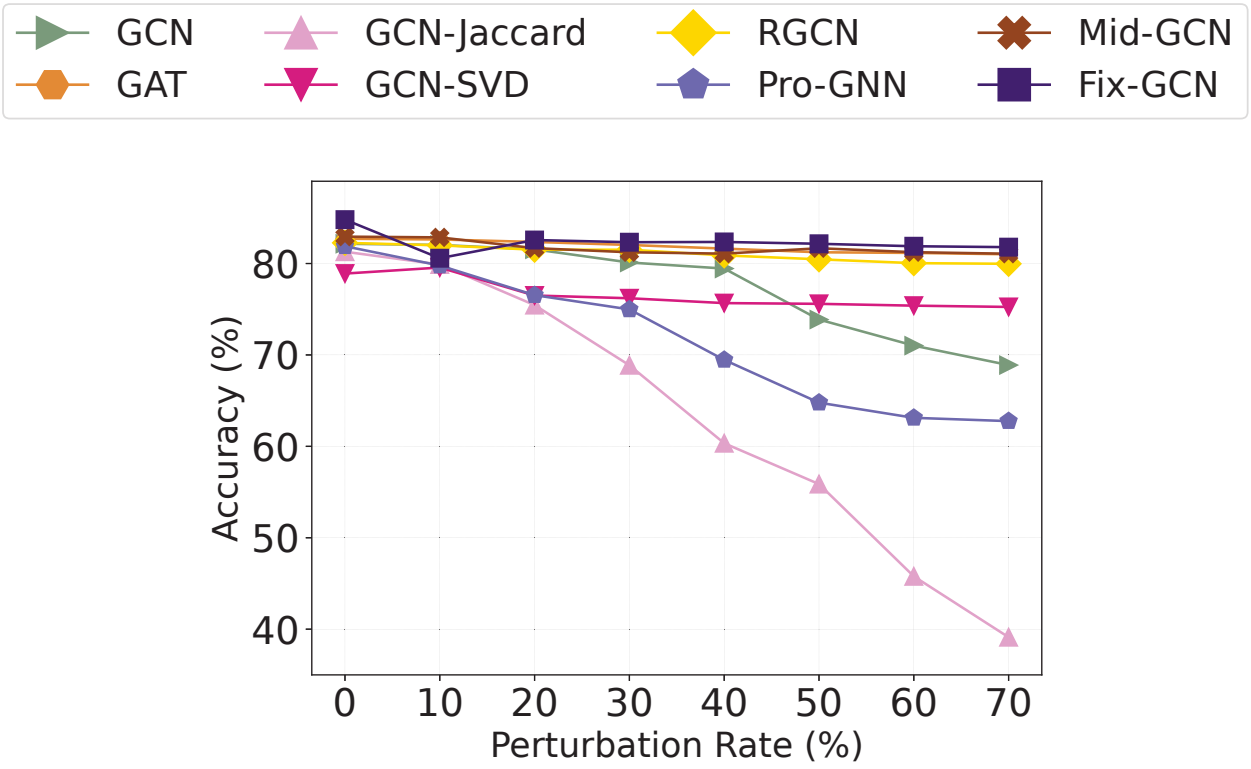
(a) Cora



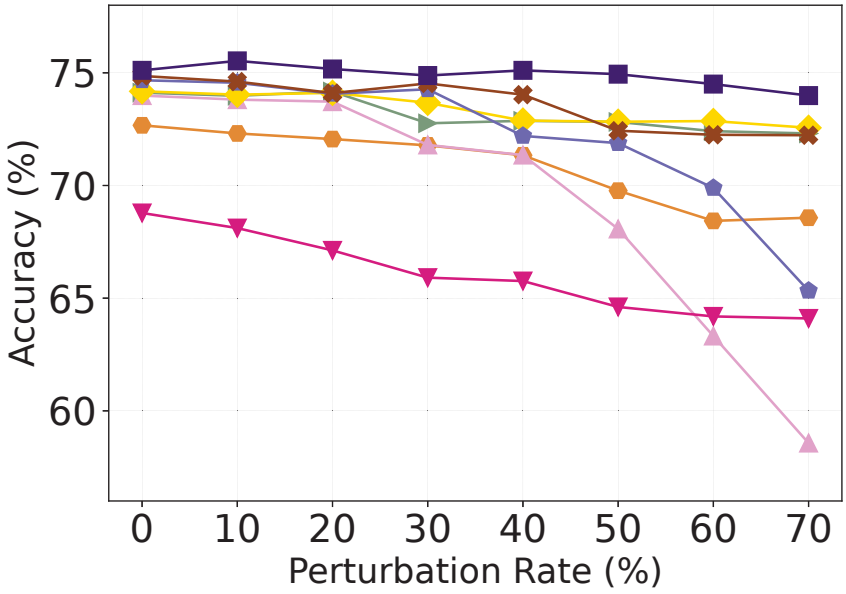
(b) CiteSeer

Figure 2.4: Node classification accuracy under random attacks with varying perturbation rates.

Robustness Against Evasion Attacks. The objective of evasion attacks is to manipulate the model predictions by making small,



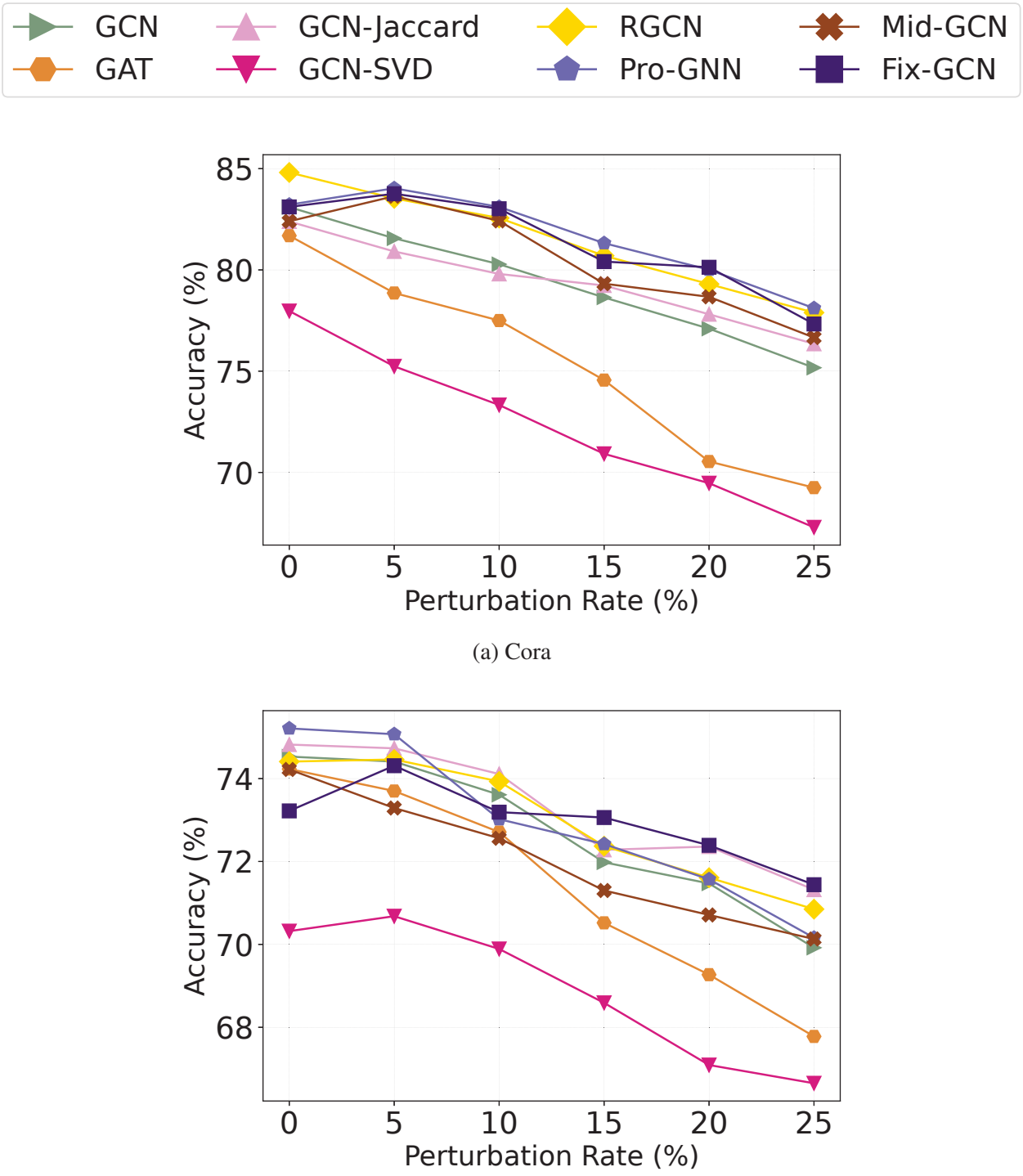
(a) Cora



(b) CiteSeer

Figure 2.5: Node classification accuracy under feature attacks with different perturbation rates.

often imperceptible changes to the graph structure during the testing phase. To assess the vulnerability of our model to evasion attacks, we employ a variant of the disconnect internally, connect



(a) Cora

(b) CiteSeer

Figure 2.6: Node classification accuracy of different models under evasion attacks (DICE) with varying perturbation rates.

externally (DICE) method [24,65], which is a white box attack strategy. This method involves randomly connecting nodes with different labels or dropping edges between nodes that share the same label. In this experiment, we vary the perturbation rate from 0% to 25%, with a step size of 5%, to evaluate the performance of our Fix-GCN model against evasion attacks. The results depicted in Figure 2.6 illustrate that Fix-GCN demonstrates robustness against evasion attacks, particularly at higher perturbation rates, where it outperforms Mid-GCN by a significant margin across both datasets. Furthermore, the performance of GAT and GCN-SVD drops rapidly as the perturbation rate increases, highlighting the effectiveness of Fix-GCN in defending against evasion attacks.

Parameter Sensitivity Analysis. We study the performance variation for our model on the three citation networks with respect to the spectral modulation filtering parameter s. We vary s from 0.1 to 0.9, and the results are presented in Figure 2.7 using Mettack as an adversarial attack with a 5% perturbation rate. It is evident that the accuracy remains relatively stable when s fluctuates between 0.1 and 0.3, which correspond to low-pass filtering. The best performance is achieved when $s = 0.2$, which is the value that we set in our experiments.

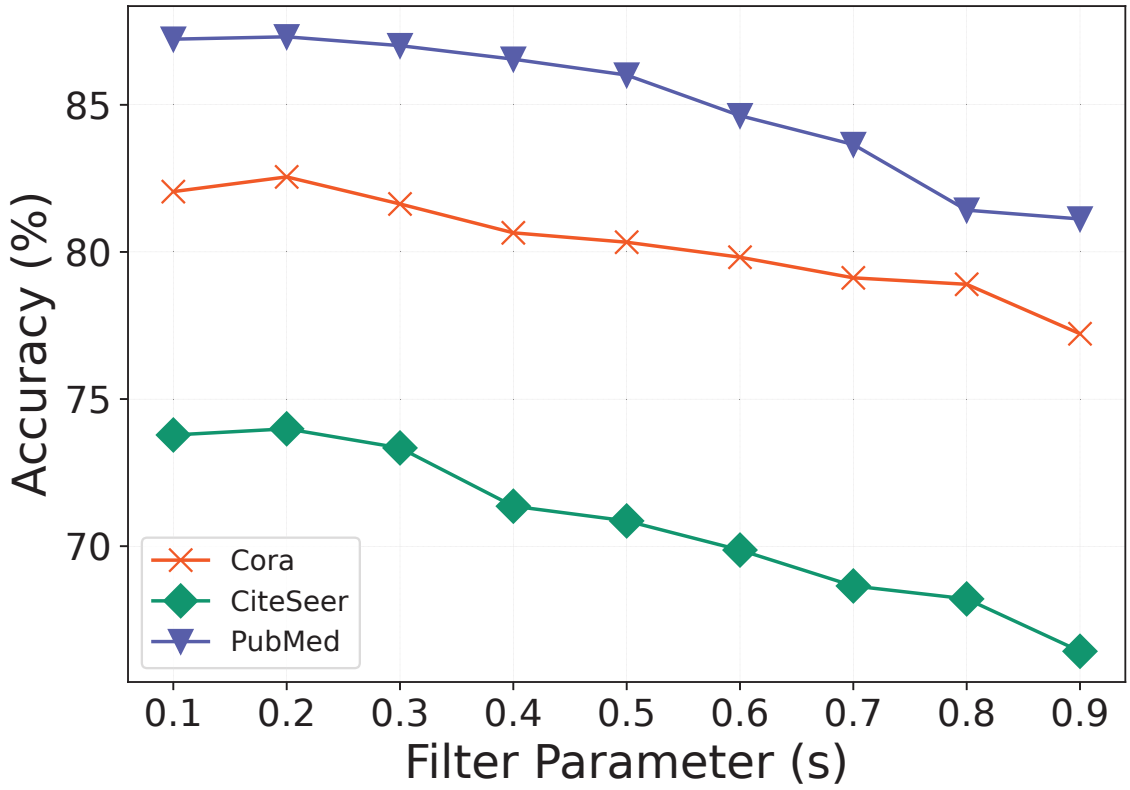


Figure 2.7: Node classification accuracy on citation networks (Cora, CiteSeer, PubMed) under Mettack with a 5% perturbation rate using various values of the spectral modulation filtering parameter s.

2.4 Discussion

In this section, we outline the merits of the proposed Fix-GCN model in three key aspects:

- *Flexibility.* By leveraging a flexible-pass filter that selectively attenuates high-frequency components while preserving low-frequency structural information in the graph signal, our Fix-GCN model is able to effectively mitigate the impact of adversarial perturbations. Moreover, capturing information from higher-order neighbors reduces the susceptibility of our model to direct perturbations on 1-hop neighbors of target nodes. Adversarial attacks that directly manipulate the immediate neighbors of target nodes can significantly impact the model's performance. By incorporating information from higher-order neighbors, Fix-GCN can mitigate the effects of such direct perturbations.

- *Robustness.* Fix-GCN's combination of selective filtering, higher-order information capture, initial residual connection, and stable performance makes it robust against various forms of adversarial attacks. In particular, the ability of our model to capture information from higher-order neighbors adds an additional layer of defense against adversarial attacks, enhancing the robustness of Fix-GCN in real-world scenarios where targeted perturbations on immediate neighbors may occur frequently.

- *Efficiency.* The time and memory complexity of Fix-GCN is on the same order as that of the standard GCN despite considering both immediate and distant graph nodes for improved node representations. This computational efficiency is achieved without the need for explicitly computing the square of the normalized adjacency matrix, as Fix-GCN utilizes right-to-left matrix multiplication to compute its second-order propagation matrix. This ensures that our model maintains practicality and scalability for real-world applications, as it can handle large-scale datasets efficiently without significantly increasing computational overhead.

While our model demonstrates robust performance against various adversarial attacks, particularly at higher perturbation levels, its robustness may be further improved by designing an adaptive filtering mechanism that dynamically adjusts the filter parameters based on the characteristics of the input graph and the severity of adversarial perturbations. This adaptive approach could enhance the model's ability to adapt to different attack strategies and varying levels of perturbations.

CHAPTER 3

Graph Encoder-Decoder Model for Robust Anomaly Detection

In this chapter, we introduce our approach for unsupervised graph anomaly detection. Standard graph convolutional networks (GCNs) mainly focus on immediate neighbors, which might not capture the broader context of a node within the graph, thereby resulting in missed detections of structural or contextual anomalies. In this chapter, we introduce an unsupervised graph encoder-decoder model for anomaly detection in attributed graphs. In the encoding stage, we design a fixed-point GCN encoder that allows for effective aggregation of information from higher-order neighborhoods, improving the encoder's ability to learn from the global structure of the graph while maintaining computational efficiency. Unlike existing GCN-based methods, the proposed encoder provides theoretical guarantees of stability due to its propagation matrix design, ensuring numerically stable feature updates. In the decoding stage, we employ a structure reconstruction decoder to predict the presence or absence of edges between nodes in the graph based on the latent representation obtained from the encoder, and an attribute reconstruction decoder to recover the original node features based on the graph structure and the learned latent representation. We conduct comprehensive empirical evaluations of our proposed encoder-decoder model on six benchmark datasets using several evaluation metrics. The results demonstrate the superiority of our model over competing anomaly detection approaches, highlighting its effectiveness in identifying anomalous nodes in attributed networks.

3.1 Introduction

Graph Anomaly Detection (GAD) is crucial for various real-world applications, including detecting social network spammers [66] [67], [68], [69], identifying intrusions in cybersecurity systems [70], and uncovering financial fraudsters [71], [72], [73]. The objective is to identify graph nodes exhibiting unusual or unexpected behavior based on their structural or feature information, or a combination of both. Detecting these anomalous nodes is challenging because anomalies are rare occurrences, and only a tiny fraction of graph nodes might be anomalous. Moreover, GAD poses unique challenges that set it apart from anomaly detection methods for tabular and time-series data [74], [75]. Unlike these traditional data formats, graph data is often multi-modal, encompassing information from both node/edge attributes and topological structures. This inherent complexity makes it difficult to establish a unified definition of anomalies for graph-structured data and to devise a principled algorithm for detecting them.

Graph-structured data often exhibits multi-modality, which adds complexity to the task of anomaly detection. The most common categories of anomalies are contextual anomalies and structural anomalies. Contextual anomalies refer to nodes whose attributes significantly deviate from those of regular nodes, such as spammers or fake account holders in social media networks. These anomalies are characterized by unusual or unexpected attribute values that differ from the typical patterns observed in the data [6]. On the other hand, structural anomalies refer to nodes with connectivity patterns that are markedly different from other nodes. Examples include a group of malicious sellers exchanging fake reviews with super dense connections or bots retweeting the same tweet, forming a densely connected co-retweet network [76]. These anomalies manifest in the form of abnormal structural patterns, such as unusual edge densities or unexpected connectivity patterns within the graph [77]. Figure 3.1 depicts these two different type anomalies.

In recent years, several approaches have been proposed to tackle the daunting task of anomaly detection. One of the earliest approaches is the clustering method [27,78] [79]. Some of the clustering techniques include distance-based methods like k-Means [80], which groups data points based on their distance from cluster centroids, grid-based [81] techniques that divide the data space into a finite number of cells and perform clustering based on the cell density, and hierarchical structure methods [82] that create a hierarchical decomposition of the data, where clusters are formed by merging or splitting clusters in a tree-like structure. Another approach aims to spot node and edge anomalies based on graph communities, relying on matrix factorization techniques [83, 84]. However, these methods have several drawbacks. Many clustering algorithms require setting various parameters (e.g., number of clusters, distance thresholds), which can significantly impact the quality of the results. Finding optimal parameter values is often non-trivial and may require domain

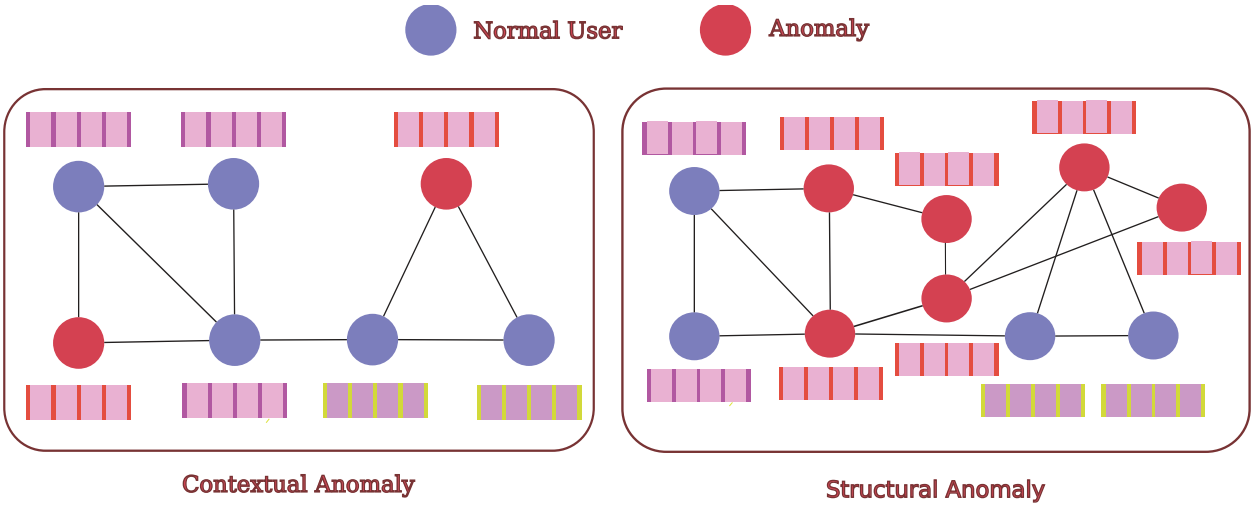


Figure 3.1: Contextual anomalies are identified based on attribute deviations relative to local neighborhoods, while structural anomalies are identified based on deviations in connectivity patterns.

expertise. Additionally, cluster-based methods are better suited for identifying global anomalies but may struggle to detect local anomalies or anomalies within dense clusters.

Another approach to graph anomaly detection is the multi-view method [37], [85], [30], [31], where different views or perspectives of the graph data are fed into the network, enabling it to detect anomalies by leveraging the diverse information sources. Specifically, some works regard the original input graph as the first view and generate a second view through graph augmentation techniques with edge modifications [38]. While innovative in handling complex data from multiple perspectives, multi-view methods face several challenges. One significant issue is that the effectiveness of multi-view anomaly detection heavily depends on the relevance and quality of the individual views. If one or more views contain noisy, irrelevant, or misleading information, it can degrade the performance of the entire model [86].

The lack of labeled data poses a significant challenge in anomaly detection. Identifying anomalies requires a sufficient amount of labeled data, which is a highly labor-intensive and costly task. To address this limitation, researchers have explored various approaches. One approach is to design new Graph Neural Network (GNN) architectures that enable network anomaly detection with limited labeled data [87]. Another technique involves utilizing data augmentation methods to generate pseudo-labels, thereby increasing the quantity of anomalous samples [88]. Furthermore, to handle the label scarcity issue, some works have employed generative-based approaches [89] [44] [90] [91] [32]. The key idea behind these methods is to generate outlier nodes that assimilate anomaly nodes in both local structure and node representations, providing effective negative node samples for training [92]. While these few-shot and generative-based methods

offer innovative solutions, they often focus on local neighborhoods or subgraphs, making it challenging to capture global topological properties and long-range dependencies in graphs. However, capturing these global characteristics is crucial for effective anomaly detection, as anomalies can manifest in both local and global patterns [93].

Another promising approach in graph anomaly detection has emerged from the domain of self-supervised learning [39]. The reason behind the effectiveness of these methods lies in their ability to learn informative knowledge without relying on manual labels. Researchers have applied self-supervised contrastive learning techniques for anomaly detection tasks [94] [40] [95]. Contrastive learning primarily focuses on studying the matching of instance pairs, which offers helpful information for anomaly detection. For normal instances in graphs, there is a potential matching pattern between each node and its neighbours, aligning with the homophily hypothesis. In contrast, anomalies often manifest when there is an inconsistency or mismatch between attributes and structure. However, the performance of contrastive learning heavily relies on the choice of graph augmentation techniques. Improper augmentations may fail to capture meaningful structural information or introduce noise, degrading the learned representations [96].

In recent years, dynamic graphs have gained significant importance in various real-world applications [97]. Consequently, researchers have focused on developing techniques for detecting anomalies in time-changing graphs using transformers [98] [99] [100] [101]. These approaches leverage the powerful self-attention mechanisms of transformers to capture the temporal and structural changes within the graph data. In addition to transformer-based approaches, reinforcement learning techniques have also been recently applied to graph anomaly detection tasks [102] [103] [104] [105] [106]. These algorithms can potentially learn effective strategies for exploring and identifying anomalous patterns within graph data by iteratively interacting with the environment and receiving feedback.

Recent developments in graph neural networks (GNNs) have empowered the unified modeling of attributes and topology, leading to significant advancements in graph anomaly detection. Another popular approach is the autoencoder-based method [107] [41] [108]. This approach employs an encoder and decoder, where the encoder maps the graph into a lower-dimensional embedding space, and the decoder attempts to reconstruct the entire input graph from these embeddings. The loss is calculated based on the reconstruction error of node attributes and structural information. Autoencoder models built upon GNNs can effectively combine node attributes and graph structure, enabling the detection of anomalies by analyzing the reconstruction loss of node attributes. Most encoder-based approaches utilize Graph Convolutional Network (GCN) [109], Graph Attention Network (GAT) [110], or GraphSage [111] as the encoder and decoder components. However, au-

toencoder methods for graph anomaly detection face several challenges and limitations. *(1)* graph autoencoders need to input and reconstruct the full networked data, which becomes infeasible due to the explosive memory requirements when dealing with large networks. *(2)* One limitation of using GCNs as encoders and decoders is the over-smoothing issue, where the representations of graph nodes from different classes become indistinguishable when stacking multiple layers, making it difficult to detect anomalies [112]. *(3)* Most current GCN models are shallow, achieving their best performance with 2-layer models. Such shallow architectures limit their ability to extract information from high-order neighbors, which is crucial for effective anomaly detection [113]. Furthermore, encoder-decoder models often focus on reconstructing local neighborhoods or subgraphs, making it challenging to capture global topological properties and long-range dependencies in the graph, which are crucial for effective anomaly detection. *(4)* The encoding and decoding processes can lead to information loss, especially when dealing with high-dimensional node features or complex graph structures, potentially degrading the anomaly detection performance [114]. While GCN-based approaches have shown promising results, addressing their limitations is crucial for developing more robust and effective graph anomaly detection models. Anomalies in graphs can manifest in various forms, such as unusual nodes whose importance are only apparent when considering the broader context of the graph. For example, a node might appear normal within its immediate neighborhood but exhibit anomalous behavior when considering its role in global graph properties. Since GCNs only consider immediate neighbors, they might fail to detect these anomalies because they do not have enough context to understand the node's global significance or its role in the broader graph structure. With their focus on immediate neighbors, GCNs provide limited context that might miss broader structural or contextual anomalies. For structural anomalies, a node might be part of a community structure that only becomes apparent when looking at the graph from a larger perspective. For contextual anomalies, a node's attributes might seem normal locally but deviate significantly when compared to nodes that are further away but share similar characteristics. For instance, in a social network, a user might have normal interactions with their immediate friends but unusual interactions when considering their position in the overall network.

In this chapter, we introduce a graph encoder-decoder model for unsupervised anomaly detection in attributed graphs. Our model leverages a fixed-point graph convolutional network (Fix-GCN) in the encoding phase to capture intricate structural and attribute information from the graph. The encoder compresses the graph into a latent representation, which succinctly encapsulates the essential features and relationships within the graph data. In the decoding phase, our model employs a structure reconstruction decoder to reconstruct the graph structure from the latent representation, along with a graph deconvolutional network to restore the node attributes from the same latent representa-

tion. This reconstruction process highlights discrepancies between the original and reconstructed graphs, enabling the detection of anomalies. By comparing the original graph's attributes and structure with the reconstructed counterparts, our model can effectively identify anomalous nodes in the graph. In addition, our approach harnesses the power of unsupervised learning, making it particularly suitable for real-world applications where labeled anomalies are scarce or unavailable. The main contributions in this chapter can be summarized as follows:

- We propose an encoder-decoder model for anomaly detection in attributed graphs. The encoder incorporates an efficient aggregation mechanism to propagate higher-order neighborhood information across graph nodes. The decoder leverages a structure reconstruction decoder and a graph deconvolutional network.
- We incorporate spectral graph wavelet denoising to mitigate noise during the decoding process, preserving the essential information of the graph.
- We demonstrate the effectiveness of the proposed model through extensive experiments on six benchmark datasets, showcasing superior performance in comparison with existing graph anomaly detection methods.

The rest of this chapter is organized as follows. Section 3.2, we delve into the methodological details of our proposed approach, outlining the key components employed to effectively detect anomalies in graph-structured data. Section 3.3 presents the experimental evaluation of our encoder-decode model. We perform extensive experiments on various datasets, assessing the effectiveness of our model in comparison with strong baseline methods.

3.2 Proposed Method

In this section, we introduce a graph encoder-decoder network for unsupervised anomaly detection. The encoder transforms the input attributed graph into a low-dimensional latent representation using graph convolutional layers to aggregate information from first- and second-order neighbors. The decoder reconstructs the node features and the graph structure from this latent representation using structure and attribute reconstruction decoders. The quality of this reconstruction is crucial for detecting anomalies, as discrepancies between the original graph and the reconstructed graph can indicate anomalies.

Basic Notions. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an attributed graph, where $\mathcal{V} = \{1, \ldots, N\}$ is the set of N nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)^\intercal$ an $N \times F$ feature matrix of node

attributes (i.e., $\mathbf{x}_i$ is an F-dimensional row vector for node i). We denote by $\mathbf{A}$ an $N \times N$ adjacency matrix whose (i, j)-th entry is equal to 1 if i and j are neighboring nodes, and 0 otherwise. We also denote by $\mathbf{L} = \mathbf{I} - \tilde{\mathbf{A}}$ the normalized Laplacian matrix, where $\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix, $\mathbf{D} = \mathrm{diag}(\mathbf{A}\mathbf{1})$ is the diagonal degree matrix, and $\mathbf{1}$ is an N-dimensional vector of all ones.

Problem Statement. The goal of unsupervised node anomaly detection in an attributed graph is to identify anomalous nodes in a graph without the use of labeled training data. In other words, there is no available ground truth information that indicates which nodes are anomalous and which ones are not. Given an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, the objective of unsupervised node anomaly detection is to learn a scoring function $\varrho : \mathcal{V} \to \mathbb{R}$ that assigns an anomaly score to each node in the graph. Once anomaly scores are computed, the r nodes with the highest anomaly scores are selected based on a user-defined value of r. These selected nodes are then identified as anomalies. In other words, nodes with high anomaly scores are considered anomalous, while nodes with lower scores are deemed normal.

Approach Overview. The overall framework of our proposed anomaly detection approach is illustrated in Figure 3.2. We design an effective graph encoder-decoder model capable of transforming an input attributed graph $\mathcal{G}$ with an adjacency matrix $\mathbf{A}$ and a feature matrix $\mathbf{X}$, into a latent representation $\mathbf{Z}$ using a fixed-point graph convolutional network (Fix-GCN) as an encoder. Then, the graph is reconstructed using a decoder comprised of two primary components: a structure reconstruction decoder and an attribute reconstruction decoder. Each of these components addresses a specific aspect of the graph reconstruction process, ensuring a comprehensive reconstruction of the graph's structural and feature information. By computing the inner products of the latent node representations generated by the encoder and applying a sigmoid activation function, the structure reconstruction decoder yields the reconstructed adjacency matrix $\hat{\mathbf{A}}$, which represents the predicted existence of edges between nodes, thereby approximating the original graph structure. Complementing the structure reconstruction decoder, the attribute reconstruction decoder focuses on reconstructing the node attributes using a graph deconvolutional network (GDN). This component essentially tries to reverse the convolutional transformations applied by the Fix-GCN encoder, yielding the reconstructed node feature matrix $\hat{\mathbf{X}}$ from the latent representation. The GDN decoder employs a deconvolution operation, which can be interpreted as a graph diffusion process. This process spreads the convolutional features back to their original nodes, taking into account the graph's topology. To mitigate the potential introduction of noise during the deconvolution process, the GDN decoder incorporates spectral graph wavelet denoising. This technique leverages graph signal processing principles to remove high-frequency noise, preserving the essen-

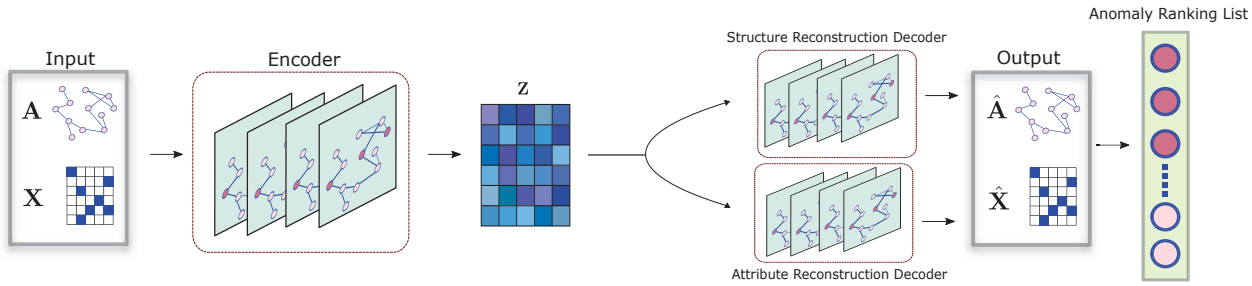


Figure 3.2: Overview of the proposed graph encoder-decoder network architecture for anomaly detection in attributed graphs. In the encoding stage, Fix-GCN is used as an encoder to generate a latent representation. The decoding stage employs two specialized decoders: a structure reconstruction decoder and a graph deconvolutional network decoder to reconstruct the graph structure and the node attributes, respectively, from the latent representation obtained during the encoding phase.

tial low-frequency components that carry the primary information. By combining the latent node representations with the original adjacency matrix and applying the deconvolution operation, the GDN decoder reconstructs the node feature matrix.

3.2.1 Encoder

In the encoding stage, we employ Fix-GCN with normalization as an encoder, which takes as input an adjacency matrix and a node feature matrix, and produces a latent representation of the graph that captures its structural and feature information by performing convolutions on the graph and aggregating information from first and second-order neighboring nodes.

Fix-GCN Encoder. The layer-wise propagation rule of Fix-GCN is given by

$$\mathbf{H}^{(\ell+1)} = \sigma\Big(\mathbf{P}_s\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{X}\widetilde{\mathbf{W}}^{(\ell)}\Big), \tag{3.1}$$

where $\mathbf{P}_s = ((1-s)\mathbf{I} + s\tilde{\mathbf{A}})\tilde{\mathbf{A}}$ is the propagation matrix, $s \in (0,1)$ is a positive scaling parameter, $\mathbf{W}^{(\ell)}$ and $\widetilde{\mathbf{W}}^{(\ell)}$ are learnable weight matrices, $\sigma(\cdot)$ is an element-wise activation function, and $\mathbf{H}^{(\ell)} \in \mathbb{R}^{N\times F_\ell}$ is the input feature matrix of the ℓ-th layer with F_ℓ feature maps for $\ell = 0,\ldots,L-1$. The input of the first layer is the initial feature matrix $\mathbf{H}^{(0)} = \mathbf{X}$.

Since $\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, applying the normalization trick to the matrix $(1-s)\mathbf{I} + s\tilde{\mathbf{A}}$ yields a new propagation matrix $\hat{\mathbf{P}}_s$ given by

$$\hat{\mathbf{P}}_s = \mathbf{D}_s^{-\frac{1}{2}}\mathbf{A}_s\mathbf{D}_s^{-\frac{1}{2}}\tilde{\mathbf{A}}, \tag{3.2}$$

where $\mathbf{A}_s = (1-s)\mathbf{I} + s\mathbf{A}$ and $\mathbf{D}_s = (1-s)\mathbf{I} + s\mathbf{D}$. By normalizing the propagation matrix, Fix-GCN ensures that information from immediate and high-order neighboring nodes is integrated

uniformly in the graph and across layers, enabling the encoder to learn effective representations that capture both local and global graph characteristics. This integration is crucial for anomaly detection tasks, where anomalies often manifest as irregularities or unexpected patterns in how nodes interact within the graph structure. Normalization helps in robustly aggregating features, thereby enhancing the model's ability to detect subtle deviations that might indicate anomalies. Hence, the layer-wise propagation rule of Fix-GCN with normalization is given by

$$\mathbf{H}^{(\ell+1)} = \sigma\Big(\hat{\mathbf{P}}_s\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)} + \mathbf{X}\widetilde{\mathbf{W}}^{(\ell)}\Big). \tag{3.3}$$

The update rule of Fix-GCN is essentially comprised of three main components: (i) feature propagation that combines the features of the 1- and 2-hop neighbors of nodes (i.e., it aggregates information from immediate and high-order neighboring nodes), (ii) feature transformation that applies learnable weight matrices to the node representations to learn an efficient representation of the graph, and (iii) residual connection for ensuring that information from the initial feature matrix is preserved. The initial residual connection used in the proposed model allows information from the initial feature matrix to bypass the current layer and be directly added to the output of the current layer. This helps preserve important information that may be lost during the aggregation process, thereby improving the flow of information through the network. The parameter s helps control the balance between the information from immediate neighbors and the information from nodes that are at most two edges away in the graph. This is particularly valuable for learning graph representations that capture more global information and dependencies. The final output node embeddings are given by an $N \times P$ feature matrix $\mathbf{Z} = \mathbf{H}^{(L)}$, where P is the embedding dimension at the final network layer. This learned low-dimensional latent representation captures the structural and semantic similarities of the graph nodes.

3.2.2 Decoder

In the decoding stage, the decoder attempts to reconstruct the original graph structure and nodal features from the latent representation generated by the encoder. The decoder is composed of two primary components: a structure reconstruction decoder and an attribute reconstruction decoder.

Structure Reconstruction Decoder. The aim of this decoder is to reconstruct the original graph structure using the latent representation $\mathbf{Z}$ obtained from the Fix-GCN encoder. Each row $\mathbf{z}_i$ of $\mathbf{Z}$ corresponds to a latent representation of node i in the graph. We reconstruct the adjacency matrix $\hat{\mathbf{A}}$ as follows:

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^{\mathsf{T}}), \tag{3.4}$$

where $\sigma(\cdot)$ is the sigmoid activation function. The term $\mathbf{Z}\mathbf{Z}^\intercal$ computes the inner product between all pairs of node representations. The element at position (i, j) in the resulting matrix represents the similarity between the latent representations $\mathbf{z}_i$ and $\mathbf{z}_j$ of nodes i and j. This similarity is intended to reflect the likelihood of an edge existing between nodes i and j. The sigmoid function is applied element-wise to the resulting matrix, squashing the values to be between 0 and 1, which can be interpreted as probabilities. In this context, $\sigma(\mathbf{Z}\mathbf{Z}^\intercal)$ represents the probability of the presence of edges between nodes in the graph. In other words, the (i, j)-th element of the reconstructed adjacency matrix $\hat{\mathbf{A}}$ represents the predicted probability that there is an edge between nodes i and j in the original graph.

Attribute Reconstruction Decoder. Given the adjacency matrix $\mathbf{A}$ and the latent representation $\mathbf{Z}$, we employ a graph deconvolutional network (GDN) [108] as a decoder to reconstruct the node feature matrix of the original graph as follows:

$$\hat{\mathbf{X}} = \mathbf{GDN}(\mathbf{A}, \mathbf{Z}). \tag{3.5}$$

The deconvolution process can be thought of as an inverse operation to graph convolution, with the aim of recovering the original node features from the compressed latent representation generated by the encoder. In the context of anomaly detection, GDN helps identify anomalies by reconstructing the feature matrix and comparing it with the original. Discrepancies between the original and reconstructed attributes can indicate the presence of anomalies.

The key idea behind the GDN decoder is to use a learnable deconvolution operation tries to reverse the convolutional transformation applied by the encoder. We apply a deconvolution operation by taking the adjacency matrix $\mathbf{A}$ and latent node representation $\mathbf{Z}$ as input for the GDN decoder, yielding a reconstructed node feature matrix $\mathbf{H} \in \mathbb{R}^{N \times F}$ given by

$$\mathbf{H} = \sigma((\mathbf{I} + \mathbf{L})\mathbf{Z}\mathbf{W}), \tag{3.6}$$

where $\mathbf{L}$ is the normalized Laplacian matrix, and $\mathbf{W} \in \mathbb{R}^{P \times F}$ is a learnable weight matrix. The deconvolution operation can be seen as a graph diffusion process that spreads the convolutional features back to their original locations, while taking into account the underlying graph structure. However, applying the inverse operation of the graph convolution may introduce undesirable noise into the output graph. This issue can be remedied using spectral graph wavelet denoising [115], a graph signal processing technique that aims to remove noise from a graph signal by leveraging a set of wavelet functions, which are usually defined as a set of filters operating on the graph Laplacian eigenvalues. The advantage of using spectral graph wavelets for denoising is that it is possible to remove noise that corresponds to high-frequency components of the signal, while preserving the

low-frequency components that carry the main information of the signal. Moreover, they provide a flexible and adaptive framework for capturing the underlying structure and smoothness of the graph signal.

Specifically, let $\mathbf{L} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$ be an eigendecomposition of the normalized Laplacian matrix, where $\mathbf{\Phi}$ is a matrix whose columns are the eigenvectors (i.e., graph Fourier basis) and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ is a diagonal matrix comprised of the corresponding eigenvalues. Spectral graph wavelets have shown to allow localization of graph signals in both spatial and spectral domains. Let $g_s(\lambda) = e^{-\lambda s}$ be the transfer function (also called frequency response) of the heat kernel with scaling parameter s. The spectral graph wavelet basis $\mathbf{\Psi}_s$ is defined as

$$\mathbf{\Psi}_s = \mathbf{\Phi}\mathbf{G}_s\mathbf{\Phi}^\mathsf{T}, \tag{3.7}$$

where

$$\mathbf{G}_s = g_s(\mathbf{\Lambda}) = \mathrm{diag}(g_s(\lambda_1), \ldots, g_s(\lambda_N)) \tag{3.8}$$

is a diagonal matrix of transformed eigenvalues via the transfer function. Note that $\mathbf{\Psi}_s$ is also referred to as the heat kernel matrix whose inverse $\mathbf{\Psi}_s^{-1}$ is obtained by simply replacing the scale parameter s with its negative value. The spectral graph wavelet basis and its inverse can also be computed efficiently using polynomial approximations via the Maclaurin series, which can be used to approximate the heat kernel on a graph, by expanding it as a polynomial in the normalized Laplacian matrix, and then truncating the series at a finite order [115].

Therefore, using the feature representation matrix $\mathbf{H}$ and both the spectral graph wavelet basis and its inverse, the reconstructed node feature matrix $\hat{\mathbf{X}} = \mathrm{GDN}(\mathbf{A}, \mathbf{Z})$ via the GDN decoder can be obtained as follows:

$$\hat{\mathbf{X}} = \mathbf{\Psi}_t \mathrm{ReLU}(\mathbf{\Psi}_t^{-1}\mathbf{H}\mathbf{W}_1)\mathbf{W}_2, \tag{3.9}$$

where $\mathbf{W}_1 \in \mathbb{R}^{F\times P}$ and $\mathbf{W}_2 \in \mathbb{R}^{P\times F}$ are learnable weight matrices. The reconstructed node feature matrix aims to provide an approximation of the original node feature matrix. By reconstructing the node feature matrix, we can gain insights into the estimated values of node features, understand the patterns within the graph, and utilize this information for further downstream tasks such as anomaly detection. These reconstructions form the basis for anomaly detection, as deviations and anomalies can be identified by comparing the reconstructed graph with the original input data, leveraging the reconstruction errors as indicators of anomalous instances.

3.2.3 Model Training

To detect anomalies, we minimize the joint reconstruction loss of the nodal attributes and topological structure, allowing for computing the reconstruction errors. This loss function [45] is defined as

a weighted combination of the modulated structure reconstruction error and the modulated feature reconstruction error:

$$\mathcal{L} = \alpha\|(\mathbf{A} - \hat{\mathbf{A}}) \odot \mathbf{\Theta}\|_F^2 + (1-\alpha)\|(\mathbf{X} - \hat{\mathbf{X}}) \odot \mathbf{\Omega}\|_F^2, \tag{3.10}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $\odot$ denotes element-wise multiplication, and α is a parameter that controls the trade-off between structure reconstruction and attribute reconstruction. The modulation matrices $\mathbf{\Theta}$ and $\mathbf{\Omega}$ are defined as

$$\mathbf{\Theta}_{i,j} = \begin{cases} 1 & \text{if } \mathbf{A}_{i,j} = 0 \\ \theta & \text{otherwise.} \end{cases}, \ \mathbf{\Omega}_{i,j} = \begin{cases} 1 & \text{if } \mathbf{X}_{i,j} = 0 \\ \omega & \text{otherwise.} \end{cases} \tag{3.11}$$

where $\theta > 1$ and $\omega > 1$ are parameters for imposing more penalty on the reconstruction error of the non-zero elements due to some missing edges or attributes.

Specifically, we aim to minimize the discrepancy between the original input graph data (adjacency matrix and node feature matrix) and their reconstructed counterparts obtained from the structure and attribute reconstruction decoders. By jointly considering the reconstruction errors for both the structural information (adjacency matrix) and the node-level attributes (feature matrix), our encoder-decoder model can capture deviations and anomalies that may manifest in either or both modalities. The final reconstruction errors are then used to compute the anomaly score ϱ_i of node i defined as follows:

$$\varrho_i = \alpha\|(\mathbf{a}_i - \hat{\mathbf{a}}_i) \odot \boldsymbol{\theta}_i\| + (1-\alpha)\|(\mathbf{x}_i - \hat{\mathbf{x}}_i) \odot \boldsymbol{\omega}_i\| \tag{3.12}$$

which is a weighted sum of a structure error term and a feature error term. Here, $\mathbf{a}_i$ and $\mathbf{x}_i$ represent the i-th rows of $\mathbf{A}$ and $\mathbf{X}$, respectively, denoting the structure and feature vectors of node i. On the other hand, $\hat{\mathbf{a}}_i$ and $\hat{\mathbf{x}}_i$ are the i-th rows of $\hat{\mathbf{A}}$ and $\hat{\mathbf{X}}$, representing the recovered structure and feature vectors obtained from the reconstruction process. Moreover, $\boldsymbol{\theta}_i$ and $\boldsymbol{\omega}_i$ represent i-th rows of $\mathbf{\Theta}$ and $\mathbf{\Omega}$, respectively. The nodes are then sorted in descending order according to their anomaly scores, which are derived from the reconstruction errors. Nodes with the highest anomaly scores are identified as potential anomalies. It is important to note that a high value of the structure error term implies that the i-th node in the graph is more likely to be an anomaly based on its structural properties, such as its connectivity patterns or topological characteristics. Conversely, a high value of the feature error term indicates an anomalous node from the perspective of its node-level attributes or features. The sorting of nodes based on their anomaly scores, derived from the reconstruction errors, allows for the identification and ranking of the most anomalous instances within the graph.

3.3 Experiments

In this section, we present our experimental setup and empirical results. Our aim is to assess the effectiveness and performance of the proposed model in comparison with strong baseline methods for anomaly detection in attributed graphs. The source code is available at: https://github.com/Shakib-IO/robust_encoder_decoder.

3.3.1 Experimental Setup

Datasets. We evaluate the model performance on two groups of standard benchmark datasets.

Social networks: BlogCatalog and Flickr are two well-known social network datasets [116], originating from popular online platforms. The BlogCatalog dataset is derived from a blog sharing website of the same name, while the Flickr dataset comes from the renowned image hosting and sharing service. In these datasets, the nodes represent individual users of the respective websites. The links or connections between nodes signify relationships or interactions between users. Typically, users on social networks create personalized content, such as blog posts on BlogCatalog or photo uploads with descriptive tags on Flickr. This user-generated content is considered as attribute data associated with the nodes (users) in the network.

Citation networks: Cora, Citeseer, Pubmed [63], and ACM [117] are well-known citation network datasets that are publicly available and widely used for research purposes. These datasets comprise collections of scientific publications from various domains. In these citation networks, the nodes represent individual published articles or thesiss. The edges or links between nodes signify citation relationships, where one article cites or references another. A key characteristic of these datasets is that each node (article) is associated with a binary feature vector. This vector encodes the presence or absence of specific words from a predefined dictionary or vocabulary. If a word is present in the article's text, the corresponding element in the binary vector is set to 1, otherwise it is 0. This binary representation allows for efficient processing and analysis of the textual content associated with each publication.

Consistent with previous studies, we follow a standardized preprocessing procedure for the attributed network datasets [103] [41] [108]. However, since these datasets do not inherently contain ground-truth labels for anomalous nodes, it is necessary to artificially introduce synthetic anomalies for the purpose of evaluating anomaly detection methods. To this end, a collection of anomalies, encompassing both structural and contextual types, are systematically injected into each dataset. Structural anomalies involve modifications to the network topology or connectivity

patterns, while contextual anomalies pertain to alterations in the attribute or feature representations associated with nodes. Statistics of the datasets are summarized in Table 3.1.

Table 3.1: Summary statistics of datasets.

Dataset	Nodes	Edges	Features	Anomalies
Cora	2708	5429	1433	150
CiteSeer	3327	4732	3703	150
PubMed	19717	44338	500	600
BlogCatalog	5196	171743	8189	300
Flickr	7575	239738	12407	450
ACM	16484	71980	8337	600

Baselines. To comprehensively evaluate the effectiveness of our proposed method, we include a diverse set of strong baselines for anomaly detection in our experimental study. These baselines encompass both traditional and encoder-decoder techniques for anomaly detection in graph-structured data, with a particular focus on attributed graphs. The baselines considered include Local Outlier Factor (LOF) [26], Structural Clustering Algorithm for Networks (SCAN) [27], Anomaly Mining of Entity Neighborhoods (AMEN) [8], Residual Analysis for Anomaly Detection in Attributed Networks (Radar) [34], a joint modeling approach for anomaly detection on attributed networks (ANOMALOUS) [118], Deep Anomaly Detection on Attributed Networks (Dominant) [41], Deep Graph Infomax (DGI) [119], Contrastive Self-Supervised Learning Framework for Anomaly Detection (CoLA), One Class Graph Neural Network (OCGNN) [35], Graph Deviation Networks (GDN) [87], Community-Aware Attributed Graph Anomaly Detection (ComGA) [36], Higher-order structure based anomaly detection on attributed networks (GUIDE) [43], Contrastive attributed network anomaly detection with data augmentation (CONAD) [95], Residual Graph Convolutional Network (ResGCN) [7], and a graph encoder–decoder network for unsupervised anomaly detection (LCPool) [108]. These baselines have been selected based on their relevance to our proposed approach and their recognition as strong graph anomaly detection methods. By comparing our method against these diverse baselines, we can comprehensively assess its performance and identify its strengths and limitations relative to existing approaches.

Evaluation Metrics. In the experiments, various evaluation metrics are used to measure the performance of different anomaly detection algorithms:

ROC–AUC: As a widely used evaluation metric in previous anomaly detection methods [31] [34] the ROC curve is a plot of true positive rate (an anomaly is recognized as an anomaly) against false

positive rate (a normal node is recognized as an anomaly) according to the ground truth and the detection results. (AUC) value is the area under the ROC curve, representing the probability that a randomly chosen abnormal node is ranked higher than a normal node.

Precision@K: As each anomaly detection method outputs a ranking list according to the anomalous scores of different nodes, we use Precision@K to measure the proportion of true anomalies that a specific detection method discovered in its top-K ranked nodes.

Recall@K: This metric measures the proportion of true anomalies that a specific detection method discovered in the total number of ground truth anomalies.

F1@K: F1@K represents the harmonic mean of precision and recall, evaluated at a specific value of K, which denotes the number of top-ranked instances being considered.

Implementation Details. We implemented our model using PyTorch [120] and Pytorch Geometric [121]. We trained our model using the Adam optimizer [122] for 200 epochs on all the datasets. The learning rate was set to 10^{-4} for BlogCatalog, Flickr, and ACM, while for Citeseer and Pubmed, it was set to 10^{-5} and 10^{-3}, respectively. Our Fix-GCN encoder had 3 hidden layers, with the embedding dimension set to 512 for all the datasets. Across all experiments, we set the hyperparameter s to 0.2, which was computed via grid search with cross-validation on the training set. The weighting hyperparameter α in the loss function was chosen between 0.5 and 0.8 for all datasets. The modulation hyperparameters θ and ω were chosen between 20 to 50 and 1 to 5, respectively. We utilized the PyGOD library [123] for the injection of contextual and structural anomalies and for running the baseline anomaly detection models. Other hyperparameters and initialization strategies followed the baseline authors recommendations [41] [45].

3.3.2 Results and Analysis

We evaluate the anomaly detection performance of our approach against strong baseline methods. Table 3.2 reports the AUC scores for our model and baselines on the six benchmark datasets. The AUC scores for the baseline methods on the citation networks are taken from [41] and [108]. The best results are shown in bold, and the second best results are underlined.

To provide a comprehensive evaluation of the anomaly detection performance, we report the results across multiple evaluation metrics, including AUC, Precision@K, Recall@K, and F1@K scores. These metrics are presented in Tables 3.2, 3.3, 3.4, and 3.5, respectively, for all datasets under consideration. Focusing on the AUC scores in Table 3.3, we observe that most shallow methods, such as LOF, SCAN, and AMEN, exhibit relatively low performance in detecting anomalies. In contrast, encoder-decoder methods like Dominant perform relatively better, whereas LCPool performs much better than these shallow approaches. Notably, contrastive-based methods like

CoLA outperform most shallow techniques, indicating their effectiveness in capturing anomalous patterns. However, despite the promising results of these methods, our analysis reveals that they are still not sufficient to comprehensively detect anomalies across all scenarios. Our proposed encoder-decoder model outperforms most of the existing baseline methods across datasets.

Table 3.3 presents the Precision@K metric results. Our model often exceeding 70%, consistently performs better than other algorithms for the majority of datasets and K values. This proves that our model is useful for precisely locating abnormalities in the top-K predictions. LCPool, another GNN-based technique, also performs well, suggesting that encoder-decoder methods effectively utilize graph structure and node attributes to identify irregularities. When compared to other traditional techniques such as LOF, Radar, Anomalous, AMEN, often falling below 50%, typically show lesser precision, underscoring their challenges in capturing anomalies. Moreover, Table 3.4 assesses the efficacy of anomaly detection using Recall@K, emphasizing the models capacity for identifying real anomalies from among the top K predictions. As shown in Table 3.4, our model performs better than other techniques on a variety of datasets and for different values of K, indicating its superiority in anomaly detection. To evaluate the anomaly detection ability, Table 3.5 uses the F1@K score, a balanced statistical metric that combines precision and recall. Our model demonstrates its overall efficacy in precisely and thoroughly finding anomalies by emerging as the best-performing approach for a broad range of datasets and K values. Finally, our model continuously performs better in anomaly identification across all four tables, as seen by its strong AUC, Precision@K, Recall@K, and F1@K scores over several datasets. This demonstrates how our model effectively detects anomalies with a low rate of false positives. In all four evaluation criteria, traditional approaches often perform lower than GNN models, indicating their limits in capturing intricate patterns and relationships in graph-structured data. All of the findings point to how crucial it is to select an assessment metric that is suited for the particular objectives of the anomaly detection activity. Overall, this comprehensive evaluation across multiple datasets and varying K values provides compelling evidence that our proposed model is a highly effective and precise tool for anomaly detection in attributed graphs, outperforming both traditional and other state-of-the-art methods.

3.3.3 Parameter Sensitivity Analysis

In our study, we explored the influence of the controlling parameters s and α on the performance of our proposed encoder-decoder framework. The results of this investigation, illustrating the variance in performance across different values of s and α are presented in Figures 3.3 and 3.4, respectively.

Table 3.2: Test AUC (%) scores on four citation networks and two social networks. Boldface numbers indicate the best performance, whereas the underlined numbers indicate the second best performance.

Method	Cora	CiteSeer	PubMed	BlogCatalog	Flickr	ACM
LOF [26]	-	-	-	49.15	48.81	47.38
SCAN [27]	-	-	-	27.27	26.86	35.99
AMEN [8]	62.66	61.54	77.13	66.48	60.47	53.37
Radar [34]	65.87	67.09	62.33	71.04	72.86	69.36
Anomalous [118]	57.70	63.07	73.16	72.81	71.59	71.85
Dominant [41]	81.55	82.51	80.81	78.13	74.90	74.94
DGI [119]	75.11	82.93	69.62	58.27	62.37	62.40
CoLA [40]	87.79	89.68	**95.12**	78.54	75.13	82.37
OCGNN [35]	86.97	85.62	74.72	55.50	48.91	50.00
GDN [87]	75.77	78.89	69.15	54.24	52.40	69.15
ComGA [36]	88.40	91.67	90.20	81.40	79.90	84.96
GUIDE [43]	64.06	83.94	83.92	70.98	72.42	83.05
CONAD [95]	76.74	78.95	76.43	79.72	65.12	69.80
ResGCN [7]	84.79	76.47	80.79	78.50	78.00	76.80
LCPool [108]	88.46	91.34	92.34	82.85	83.12	84.69
Ours	**91.67**	**92.31**	90.33	**88.43**	**87.18**	**87.88**

Table 3.3: Test Precision@K (%) scores of our approach and baselines on four citation networks and two social networks. Boldface numbers indicate the best performance, whereas the underlined numbers indicate the second best performance.

	BlogCatalog				Flicker				ACM				Cora				Citeseer				Pubmed			
K	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300
LOF [26]	30.0	22.0	18.0	18.3	42.0	38.0	27.0	23.7	6.0	6.0	4.5	3.7	-	-	-	-	-	-	-	-	-	-	-	-
Radar [34]	66.0	67.0	55.0	41.6	74.0	70.0	63.5	50.3	5.6	5.8	5.2	4.3	-	-	-	-	-	-	-	-	-	-	-	-
Anomalous [118]	64.0	65.0	51.5	41.7	79.0	71.0	65.0	51.0	60.0	57.0	51.0	41.0	-	-	-	-	-	-	-	-	-	-	-	-
AMEN [8]	60.0	58.0	49.6	38.3	67.0	64.0	55.0	46.1	52.0	49.0	43.2	36.0	54.6	47.2	29.0	23.0	64.0	44.0	23.0	21.6	56.0	54.1	49.0	45.7
Dominant [41]	76.0	71.0	59.0	**47.0**	77.0	73.0	68.5	59.3	62.0	59.0	54.0	49.7	68.0	55.0	36.0	27.0	76.0	51.0	32.0	25.3	70.0	66.0	63.0	56.0
DGI [87]	52.0	51.0	43.6	32.3	59.0	57.7	46.0	45.4	46.0	41.4	38.0	35.4	47.1	39.0	25.3	19.1	54.0	36.3	21.0	17.9	49.0	48.0	44.0	39.5
CoLA [40]	62.0	58.0	39.5	31.0	60.0	51.0	31.5	26.7	**88.0**	**71.0**	57.5	46.8	66.0	54.0	**41.5**	**34.3**	58.0	47.0	**39.0**	**31.7**	76.0	69.0	58.5	55.7
LCPool [108]	80.0	75.0	60.0	31.7	84.0	79.0	71.0	**63.3**	**88.0**	69.1	**58.0**	47.1	74.0	55.0	31.0	26.0	78.0	59.0	32.5	27.3	75.8	69.2	**64.3**	57.0
Ours	**83.1**	**79.3**	**64.7**	45.3	**84.5**	**80.1**	**73.3**	62.2	86.3	68.3	55.4	**50.3**	**77.9**	**58.4**	34.6	27.5	**81.5**	**60.4**	33.5	30.3	**78.4**	**71.2**	62.3	**58.9**

Regarding the hyperparameter s, we explored a range of values from 0.1 to 0.9. To determine the optimal value, we performed a grid search across all the datasets. Our observations indicate that when s falls between 0.1 and 0.3, our model exhibits superior performance, with the highest results achieved when $s = 0.2$. Figure 3.3 illustrates the AUC values in percentage form, where we can observe a declining trend in our models performance as the value of s increases beyond the optimal range for all the datasets.

We further analyzed the effect of the trade-off hyperparameter α on the models performance, and the results are presented in Figure 3.4. When $\alpha = 0$, our loss function reduces to the structure

Table 3.4: Test Recall@K (%) scores of our approach and baselines on four citation networks and two social networks. Boldface numbers indicate the best performance, whereas the underlined numbers indicate the second best performance.

	BlogCatalog				Flicker				ACM				Cora				Citeseer				Pubmed			
K	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300
LOF [26]	5.0	7.3	12.0	18.3	4.7	8.4	12.0	15.8	0.5	1.0	1.5	1.8	-	-	-	-	-	-	-	-	-	-	-	-
Radar [34]	11.0	22.3	36.7	41.6	8.2	15.6	28.2	33.6	4.7	9.7	17.3	21.5	-	-	-	-	-	-	-	-	-	-	-	-
Anomalous [118]	10.7	21.7	34.3	41.7	8.7	15.8	28.9	34.0	5.0	9.5	17.0	20.5	-	-	-	-	-	-	-	-	-	-	-	-
AMEN [8]	9.7	19.6	32.4	38.9	7.2	14.3	26.9	32.1	4.5	7.9	15.9	17.1	20.9	31.4	42.4	47.6	21.6	30.1	37.7	44.7	4.4	8.1	15.5	22.2
Dominant [41]	12.7	23.7	39.3	47.0	8.4	16.2	30.4	39.6	5.2	9.8	18.0	24.8	23.7	36.7	48.0	54.0	25.3	34.0	42.7	50.7	**28.3**	11.0	21.0	28.0
DGI [87]	8.4	17.1	28.2	34.1	7.3	13.0	24.4	29.3	4.3	8.4	13.7	16.5	18.1	27.1	31.9	35.8	17.2	24.1	30.3	35.8	3.5	6.6	12.1	17.7
CoLA [40]	10.4	19.5	26.5	31.2	6.7	11.5	14.1	18.0	7.3	11.9	19.3	23.4	22.0	36.0	55.3	68.7	19.3	31.3	52.0	63.3	6.3	11.5	19.5	27.9
LCPool [108]	24.8	25.2	**40.3**	**60.9**	18.2	20.6	41.3	48.8	**9.8**	**18.5**	21.0	28.3	**42.5**	51.7	71.3	**88.5**	31.0	**52.2**	57.5	70.8	11.8	15.8	27.8	30.8
Fix-GCN	**30.4**	**26.2**	39.4	50.7	**23.7**	**26.5**	**43.9**	**50.7**	9.1	18.0	**28.2**	**30.4**	40.3	**54.1**	**74.3**	70.3	**35.7**	50.1	**59.6**	**71.6**	22.7	**24.1**	**29.3**	**31.9**

Table 3.5: Test F1@K (%) scores of our approach and baselines on four citation networks and two social networks. Boldface numbers indicate the best performance, whereas the underlined numbers indicate the second best performance.

	BlogCatalog				Flicker				ACM				Cora				Citeseer				Pubmed			
K	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300	50	100	200	300
LOF [26]	8.6	10.9	14.4	18.3	8.4	13.7	16.6	18.9	0.9	1.7	2.2	2.4	-	-	-	-	-	-	-	-	-	-	-	-
Radar [34]	18.8	33.4	44.0	41.6	14.7	25.5	39.0	40.3	5.1	7.2	7.9	7.2	-	-	-	-	-	-	-	-	-	-	-	-
Anomalous [118]	18.3	32.5	41.2	41.7	15.7	25.8	40.0	40.8	9.2	16.3	25.5	27.3	-	-	-	-	-	-	-	-	-	-	-	-
AMEN [8]	16.7	29.3	39.2	38.6	13.0	23.4	36.1	37.8	8.3	13.6	23.2	23.2	30.2	37.7	34.4	31.0	32.3	35.7	28.6	29.1	8.2	14.1	23.6	29.9
Dominant [41]	21.8	35.5	47.2	47.0	15.1	26.5	42.1	47.5	9.6	16.8	27.0	33.1	35.1	44.0	41.1	36.0	38.0	40.8	36.6	33.8	40.3	18.9	31.5	37.3
DGI [87]	14.5	25.6	34.2	33.2	13.0	21.2	31.9	35.6	7.9	14.0	20.1	22.5	26.2	32.0	28.2	24.9	26.1	29.0	24.8	23.9	6.5	11.6	19.0	24.4
CoLA [40]	17.8	29.2	31.7	31.1	12.1	18.8	19.5	21.5	13.5	20.4	28.9	31.2	33.0	43.2	47.4	**45.8**	29.0	37.6	44.6	42.2	11.6	19.7	29.3	37.2
LCPool [108]	37.9	37.7	48.2	41.7	**29.9**	32.7	52.2	55.1	17.6	29.2	30.8	**35.4**	**54.0**	53.3	43.2	40.2	44.4	55.4	41.5	39.4	20.4	25.7	**38.8**	**40.0**
Fix-GCN	**42.3**	**41.3**	**50.9**	**48.2**	28.4	**35.6**	**56.3**	**60.2**	**22.5**	**33.7**	**33.2**	32.4	50.3	**56.0**	**48.6**	45.2	**47.3**	**57.3**	**50.2**	**46.3**	**22.3**	**29.8**	36.2	37.0

reconstruction loss, while when $\alpha = 1$, it reduces to the feature reconstruction loss. The structure reconstruction loss evaluates the extent to which our model accurately represents the original adjacency matrix, whereas the feature reconstruction loss assesses the reconstructed node feature representation. As can be observed in Figure 3.4, our model generally yields higher AUC values when α is between 0.5 and 0.8. This suggests that the best detection performance is typically achieved by simultaneously considering the reconstruction errors of both graph structure and node features. By striking a balance between these two components through the trade-off hyperparameter α, our model can effectively capture and leverage the complementary information present in the graph topology and node attributes, leading to improved anomaly detection capabilities.

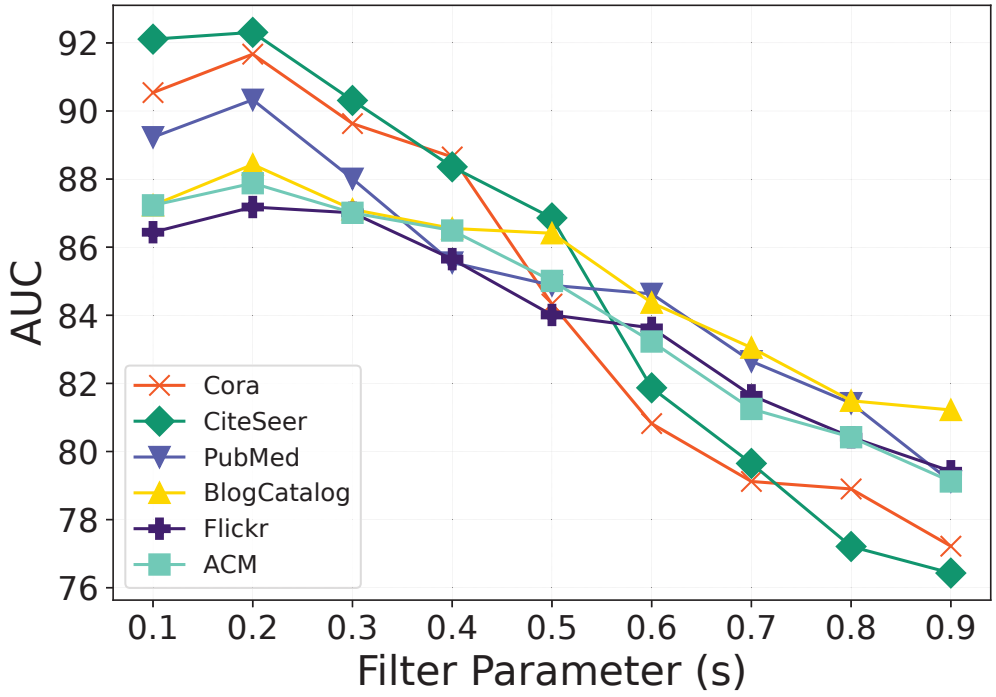


Figure 3.3: Effect of hyperparameter s on anomaly detection performance of our model using AUC as evaluation metric

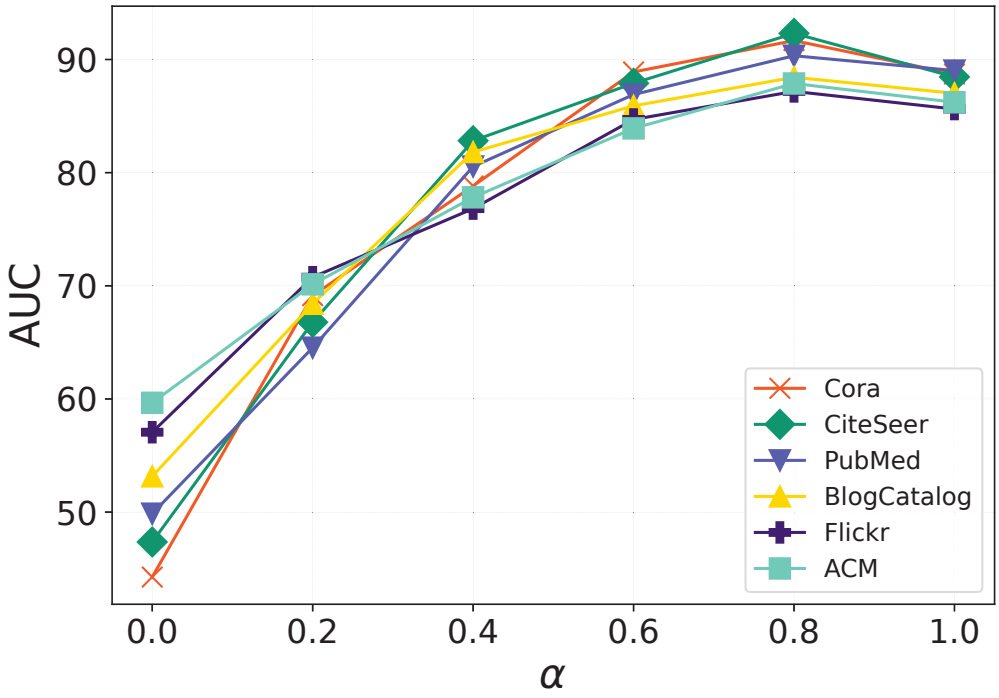


Figure 3.4: Effect of hyperparameter α on anomaly detection performance of our model using AUC as evaluation metric

3.4 Discussion

In this section, we highlight the merits of the proposed encoder-decoder model across three key aspects:

- *Improved Representation Learning.* The proposed encoder-decoder model excels in capturing the intricate relationships within attributed graphs by simultaneously encoding both structural and attribute information. This dual focus enables it to generate a unified latent representation that integrates graph topology and node feature details.

- *Enhanced Reconstruction Capability.* Featuring a dual decoder architecture, the model achieves superior reconstruction of both the graph structure and node attributes. The structure reconstruction decoder accurately restores the adjacency matrix, crucial for maintaining graph integrity, while the attribute reconstruction decoder employs a graph deconvolutional network optimized for recovering node features. This dual-decoder setup not only ensures fidelity in representation but also facilitates robust anomaly detection by providing comprehensive insight into anomalous nodes in attributed graphs.

- *Robustness through Noise Reduction.* To mitigate noise inherent in attributed graphs, the model integrates spectral graph wavelet denoising during the attribute reconstruction phase. This method selectively filters high-frequency noise while preserving essential low-frequency components, thereby enhancing the fidelity of reconstructed node attributes. By effectively reducing noise, the model improves overall robustness and reliability in anomaly detection tasks, ensuring that detected anomalies are meaningful deviations rather than artifacts of noise or irrelevant variations.

These strengths collectively position the proposed encoder-decoder model as a robust framework for unsupervised anomaly detection in attributed graphs, capable of handling complex graph structures while maintaining high fidelity in reconstruction and interpretation of both structural and attribute information.

CHAPTER 4

Conclusions and Future Work

This thesis introduces a robust and efficient framework for graph convolutional networks (GCNs) with the aim of enhancing their resilience to adversarial attacks and improving their capability to accurately detect anomalies in graph-structured data. This is achieved by leveraging spectral modulation filtering and resilient aggregation mechanisms to address the inherent limitations of standard GCNs, particularly their vulnerability to adversarial manipulations and their limited ability to consider higher-order neighbor information. At the core of our approach is a novel GCN-based model that draws inspiration from the fixed-point iterative method, incorporating a layer-wise propagation rule derived from iteratively solving graph filtering. The proposed framework is comprised of several key components that work in tandem to improve the model's effectiveness and efficiency. First, we introduce a spectral modulation filter that selectively attenuates high-frequency components while preserving low-frequency structural information, allowing for more nuanced feature extraction from the graph signal. Second, we implement an aggregation mechanism that efficiently captures information from higher-order node neighbors, expanding the network's receptive field without increasing computational complexity. We also adopt a spectral graph wavelet denoising method to mitigate noise while retaining essential graph information. The proposed framework aims to pave the way for more resilient and effective GCNs, ensuring reliable performance in various real-world graph-based applications. In Section 4.1, we discuss the concluding outcomes of the associated research work in each of the previous chapters, along with the contributions made. Furthermore, we address the limitations of the proposed approach in Section 4.2, and present suggestions for potential research directions related to this thesis in Section 4.3.

4.1 Contributions of the Thesis

4.1.1 Fixed-Points Graph Convolutional Networks Against Adversarial Attacks

In Chapter 2, we introduced Fix-GCN, a robust model against adversarial attacks. The core concept behind Fix-GCN lies in its message-passing mechanism, which is derived from solving a spectral graph modulation filtering system using fixed-point iteration. One of the key strengths of our model is its ability to capture information from higher-order connections, thereby improving its resilience against direct perturbations on immediate neighbors, which are often more vulnerable to adversarial attacks. By considering information from higher-order neighbors, Fix-GCN effectively captures the global context of the graph, making it more resilient to direct perturbations on 1-hop neighbors. Moreover, Fix-GCN maintains computational efficiency comparable to the standard GCN without sacrificing performance or requiring additional defense mechanisms. Our comparative experiments showed that our model yields improved performance compared to strong baselines across different graph datasets and under a variety of adversarial attacks.

4.1.2 Graph Encoder-Decoder Model for Robust Anomaly Detection

In Chapter 3, we introduced a graph encoder-decoder model for unsupervised anomaly detection in attributed graphs. The encoder in our model is designed to learn a compressed, latent representation of the input graph by aggregating information from first- and second-order neighboring nodes. The decoder, which consists of a structure reconstruction decoder and an attribute reconstruction decoder, reconstructs the original graph structure and node attributes from the latent representation learned by the encoder. The structure reconstruction decoder reconstruct the adjacency matrix of the original graph by computing the inner product between the latent representations of all node pairs, while the attribute reconstruction decoder employs a graph deconvolutional network as a decoder to recover the original node features by spreading the latent features back to their respective nodes, considering the graph structure. To address potential noise introduced by the deconvolution operation, we incorporated spectral graph wavelet denoising, which removes high-frequency noise while preserving the low-frequency components that carry significant information. The experimental results demonstrate the superiority of the proposed model over competitive baseline methods.

4.2 Limitations

While our proposed framework demonstrates significant advancements in robustness and accuracy, achieving state-of-the-art performance in many scenarios, it also reveals certain limitations that warrant consideration. One primary constraint is the model's difficulty in capturing long-range dependencies in larger graphs, despite its efficient handling of higher-order neighborhood information. This scalability issue becomes particularly pronounced when dealing with very large graph structures. Additionally, like many deep learning models, Fix-GCN may be susceptible to overfitting, especially when trained on smaller graph datasets. The model's performance is also sensitive to hyperparameter selection, necessitating careful tuning for optimal results. Interpretability remains a challenge, as the decision-making process of Fix-GCN can be difficult to decipher, particularly for complex graph structures. Furthermore, the model's effectiveness may vary depending on the underlying graph structure, potentially leading to inconsistent performance across different types of graphs. While Fix-GCN aims to mitigate the over-smoothing issue common in traditional GCNs, this problem may still persist to some extent, especially in deeper architectures. The model's primary design for static graphs might limit its adaptability to dynamic or temporal graph structures. Lastly, the robustness of Fix-GCN to noise and incomplete data, which are common in real-world scenarios, remains an area for potential improvement. These limitations highlight areas for future research and refinement in graph neural network architectures.

4.3 Future Work

Several interesting research directions, motivated by this thesis, are discussed below:

4.3.1 Dynamic Graphs

In recent years, dynamic graphs have gained significant prominence across various domains, reflecting the evolving nature of real-world networks and relationships. Recognizing the growing importance of these time-varying graph structures, we aim to extend the application of our proposed Fix-GNN method to dynamic graph data. This expansion of our research focus will allow us to explore the methods capabilities in detecting graph anomalies and enhancing robustness in dynamic graph environments.

4.3.2 Kolmogorov-Arnold Networks

Kolmogorov-Arnold Networks (KANs) have recently been introduced as promising alternatives to multi-layer perceptrons (MLPs), offering new possibilities for data processing and representation. KANs are based on a theoretical result known as the Kolmogorov-Arnold representation theorem, which states that any multivariate continuous function can be represented as a composition of a finite number of univariate continuous functions and the addition operation. This theorem is similar in spirit to the universal approximation theorem, which states that an MLP with single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of the n-dimensional Euclidean space. While MLPs employ fixed activation functions on nodes (i.e., neurons), KANs take a different approach by placing learnable activation functions on edges (i.e., weights). This fundamental difference means that KANs do not use linear weight matrices at all; instead, each weight parameter is substituted with a learnable one-dimensional function, typically parameterized as a spline. In KANs, nodes simply sum incoming signals without applying any non-linearities, which contrasts with MLPs where non-linear activations are integral to node operations. This unique design characteristic of KANs enables them to adaptively learn complex mappings while potentially offering advantages in tasks requiring flexible and data-driven adjustments to edge weights.

The adaptation of KAN principles to graph-structured data represents a significant development in the field of graph analysis and machine learning. Recognizing the potential of this emerging approach, we intend to integrate our Fix-GCN method with KAN principles. This integration aims to combine the strengths of our fixed-point graph convolutional network with the approximation capabilities of KAN. By merging these two powerful concepts, we anticipate potential improvements in various downstream tasks related to graph analysis.

4.3.3 Applications

In our future work, we aim to expand the application and capabilities of our model across diverse domains. A primary focus will be on exploring its potential in biomedical networks, particularly protein-protein interaction networks, where detecting anomalies could lead to the discovery of new interactions or disease biomarkers. This application has significant implications for advancing our understanding of biological systems and potentially accelerating drug discovery processes. Beyond biomedical applications, we intend to investigate the models efficacy in security datasets, where anomaly detection is crucial for identifying potential threats or unusual activities. Furthermore, we are keen on exploring the models potential in computer vision tasks, with a particular interest in 3D human pose estimation. This interdisciplinary approach aims to leverage our graph-based tech-

niques to improve accuracy and robustness in pose estimation, potentially opening new avenues in fields such as motion capture, augmented reality, and human-computer interaction. Through these diverse applications, we hope to demonstrate the broad applicability and adaptability of our graph-based model across various scientific and technological domains.

References

[1] W. Jin, Y. Li, H. Xu, Y. Wang, S. Ji, C. Aggarwal, and J. Tang, "Adversarial attacks and defenses on graphs," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 19–34, 2021.

[2] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2847–2856, 2018.

[3] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," in *International Conference on Learning Representations*, 2019.

[4] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1399–1407, 2019.

[5] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu, "Adversarial examples on graph data: Deep insights into attack and defense," in *Proc. International Joint Conference on Artificial Intelligence*, 2019.

[6] X. Hu, J. Tang, and H. Liu, "Online social spammer detection," in *Proc. AAAI Conference on Artificial Intelligence*, 2014.

[7] Y. Pei, T. Huang, W. van Ipenburg, and M. Pechenizkiy, "ResGCN: Attention-based deep residual modeling for anomaly detection on attributed networks," *Machine Learning*, vol. 111, pp. 519–541, 2022.

[8] B. Perozzi and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods," in *Proc. SIAM International Conference on Data Mining*, pp. 207–215, 2016.

[9] S. Biasotti, A. Cerri, M. Abdelrahman, and et al., "SHREC'14 track: Retrieval and classification on textured 3D models," in *Proc. Eurographics Workshop on 3D Object Retrieval*, pp. 111–120, 2014.

[10] S. Biasotti, A. Cerri, M. Aono, and et al., "Retrieval and classification methods for textured 3D models: a comparative study," *The Visual Computer*, vol. 32, pp. 217–241, 2016.

[11] M. Masoumi and A. Ben Hamza, "Spectral shape classification: A deep learning approach," *Journal of Visual Communication and Image Representation*, vol. 43, pp. 198–211, 2017.

[12] E. Rodola, L. Cosmo, O. Litany, and et al., "SHREC'17: Deformable shape retrieval with missing parts," in *Proc. Eurographics Workshop on 3D Object Retrieval*, 2017.

[13] M. Masoumi and A. Ben Hamza, "Shape classification using spectral graph wavelets," *Applied Intelligence*, vol. 47, pp. 1256–1269, 2017.

[14] H. Krim and A. Ben Hamza, *Geometric methods in signal and image analysis*. Cambridge University Press, 2015.

[15] E. E. Abdallah, A. Ben Hamza, and P. Bhattacharya, "Spectral graph-theoretic approach to 3D mesh watermarking," in *Proceedings of Graphics Interface*, pp. 327–334, 2007.

[16] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[17] N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis, "All you need is low (rank) defending against adversarial attacks on graphs," in *Proc. International Conference on Web Search and Data Mining*, pp. 169–177, 2020.

[18] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 66–74, 2020.

[19] A. Alchihabi, Q. En, and Y. Guo, "Efficient low-rank gnn defense against structural attacks," in *Proc. IEEE International Conference on Knowledge Graph*, 2023.

[20] X. Zhang and M. Zitnik, "GNNGUARD: Defending graph neural networks against adversarial attacks," in *Advances in Neural Information Processing Systems*, pp. 9263–9275, 2020.

[21] W. Jin, T. Derr, Y. Wang, Y. Ma, Z. Liu, and J. Tang, "Node similarity preserving graph convolutional networks," in *Proc. ACM International Conference on Web Search and Data Mining*, pp. 148–156, 2021.

[22] X. Liu, W. Jin, Y. Ma, Y. Li, H. Liu, Y. Wang, M. Yan, and J. Tang, "Elastic graph neural networks," in *Proc. International Conference on Machine Learning*, 2021.

[23] H. Chang, Y. Rong, T. Xu, Y. Bian, S. Zhou, X. Wang, J. Huang, and W. Zhu, "Not all low-pass filters are robust in graph convolutional networks," in *Advances in Neural Information Processing Systems*, 2021.

[24] R. Lei, Z. Wang, Y. Li, B. Ding, and Z. Wei, "EvenNet: Ignoring odd-hop neighbors improves robustness of graph neural networks," in *Advances in Neural Information Processing Systems*, 2022.

[25] J. Huang, L. Du, X. Chen, Q. Fu, S. Han, and D. Zhang, "Robust mid-pass filtering graph convolutional networks," in *Proc. ACM Web Conference*, pp. 328–338, 2023.

[26] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proc. ACM SIGMOD International Conference on Management of Data*, pp. 93–104, 2000.

[27] X. Xu, N. Yuruk, Z. Feng, and T. A. Schweiger, "SCAN: a structural clustering algorithm for networks," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 824–833, 2007.

[28] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking graph neural networks for anomaly detection," in *Proc. International Conference on Machine Learning*, pp. 21076–21089, 2022.

[29] R. Francisquini, A. C. Lorena, and M. C. Nascimento, "Community-based anomaly detection using spectral graph filtering," *Applied Soft Computing*, vol. 118, p. 108489, 2022.

[30] Z. Peng, M. Luo, J. Li, L. Xue, and Q. Zheng, "A deep multi-view framework for anomaly detection on attributed networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, pp. 2539–2552, 2020.

[31] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, and Y. Qi, "A semi-supervised graph attentive network for financial fraud detection," in *Proc. IEEE International Conference on Data Mining*, pp. 598–607, 2019.

[32] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y.-P. P. Chen, "Generative and contrastive self-supervised learning for graph anomaly detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 12220–12233, 2021.

[33] J. Zhang, S. Wang, and S. Chen, "Reconstruction enhanced multi-view contrastive learning for anomaly detection on attributed networks," in *Proc. International Joint Conference on Artificial Intelligence*, pp. 2376–2382, 2022.

[34] J. Li, H. Dani, X. Hu, and H. Liu, "Radar: Residual analysis for anomaly detection in attributed networks," in *Proc. International Joint Conference on Artificial Intelligence*, pp. 2152–2158, 2017.

[35] X. Wang, B. Jin, Y. Du, P. Cui, Y. Tan, and Y. Yang, "One-class graph neural networks for anomaly detection in attributed networks," *Neural Computing and Applications*, vol. 33, pp. 12073–12085, 2021.

[36] X. Luo, J. Wu, A. Beheshti, J. Yang, X. Zhang, Y. Wang, and S. Xue, "Comga: Community-aware attributed graph anomaly detection," in *Proc. ACM International Conference on Web Search and Data Mining*, pp. 657–665, 2022.

[37] Z. Liu, C. Cao, and J. Sun, "Mul-GAD: a semi-supervised graph anomaly detection framework via aggregating multi-view information," *arXiv preprint arXiv:2212.05478*, 2022.

[38] J. Duan, S. Wang, P. Zhang, E. Zhu, J. Hu, H. Jin, Y. Liu, and Z. Dong, "Graph anomaly detection via multi-scale contrastive learning networks with augmented view," in *Proc. AAAI Conference on Artificial Intelligence*, pp. 7459–7467, 2023.

[39] T. Huang, Y. Pei, V. Menkovski, and M. Pechenizkiy, "Hop-count based self-supervised anomaly detection on attributed networks," in *Proc. Joint European Conference on Machine Learning & Knowledge Discovery in Databases*, pp. 225–241, 2022.

[40] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, pp. 2378–2392, 2021.

[41] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep anomaly detection on attributed networks," in *Proc. SIAM International Conference on Data Mining*, pp. 594–602, 2019.

[42] D. Zhu, Y. Ma, and Y. Liu, "Anomaly detection with deep graph autoencoders on attributed networks," in *Proc. IEEE Symposium on Computers and Communication*, pp. 1–6, 2020.

[43] X. Yuan, N. Zhou, S. Yu, H. Huang, Z. Chen, and F. Xia, "Higher-order structure based anomaly detection on attributed networks," in *Proc. IEEE International Conference on Big Data*, pp. 2691–2700, 2021.

[44] Z. Chen, B. Liu, M. Wang, P. Dai, J. Lv, and L. Bo, "Generative adversarial attributed network anomaly detection," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 1989–1992, 2020.

[45] H. Fan, F. Zhang, and Z. Li, "AnomalyDAE: Dual autoencoder for anomaly detection on attributed networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5685–5689, 2020.

[46] A. Roy, J. Shu, J. Li, C. Yang, O. Elshocht, J. Smeets, and P. Li, "GAD-NR: Graph anomaly detection via neighborhood reconstruction," in *Proc. ACM International Conference on Web Search and Data Mining*, pp. 576–585, 2024.

[47] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[48] F. Wu, T. Zhang, A. H. de Souza Jr., C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proc. International Conference on Machine Learning*, 2019.

[49] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, G. Ver Steeg, and A. Galstyan, "MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proc. International Conference on Machine Learning*, pp. 21–29, 2019.

[50] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. International Conference on Machine Learning*, pp. 1725–1735, 2020.

[51] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, F. Alet, S. Ravuri, T. Ewalds, Z. Eaton-Rosen, W. Hu, A. Merose, S. Hoyer, G. Holland, O. Vinyals, J. Stott, A. Pritzel, S. Mohamed, and P. Battaglia, "GraphCast: Learning skillful medium-range global weather forecasting," *Science*, pp. 1416–1421, 2023.

[52] M. M. Li, K. Huang, and M. Zitnik, "Graph representation learning in biomedicine and healthcare," *Nature Biomedical Engineering*, pp. 1353–1369, 2022.

[53] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[54] H. He, Y. Ji, and H. H. Huang, "ILLUMINATI: Towards explaining graph neural networks for cybersecurity analysis," in *Proc. IEEE European Symposium on Security and Privacy*, pp. 74–89, 2022.

[55] L. Sun, Y. Dou, C. Yang, K. Zhang, J. Wang, S. Y. Philip, L. He, and B. Li, "Adversarial attack and defense on graph data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[56] S. Geisler, D. Zügner, and S. Günnemann, "Reliable graph neural networks via robust aggregation," in *Advances in Neural Information Processing Systems*, pp. 13272–13284, 2020.

[57] Y. Wang, S. Liu, M. Yoon, H. Lamba, W. Wang, C. Faloutsos, and B. Hooi, "Provably robust node classification via low-pass message passing," in *Proc. IEEE International Conference on Data Mining*, pp. 621–630, 2020.

[58] L. Zhang and H. Lu, "A feature-importance-aware and robust aggregator for GCN," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 1813–1822, 2020.

[59] R. L. Burden, J. D. Faires, and A. M. Burden, *Numerical Analysis*. Cengage Learning, 2015.

[60] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[61] B. Rozemberczki, C. Allen, and R. Sarkar, "Multi-scale attributed node embedding," *Journal of Complex Networks*, vol. 9, no. 2, pp. 1–22, 2021.

[62] A. Bojchevski and S. Günnemann, "Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking," in *International Conference on Learning Representations*, 2018.

[63] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.

[64] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[65] M. Waniek, T. P. Michalak, M. J. Wooldridge, and T. Rahwan, "Hiding individuals and communities in a social network," *Nature Human Behaviour*, vol. 2, no. 2, pp. 139–147, 2018.

[66] J. Ye and L. Akoglu, "Discovering opinion spammer groups by network footprints," in *Proc. Joint European Conference on Machine Learning & Knowledge Discovery in Databases*, pp. 267–282, 2015.

[67] V. H. Nguyen, K. Sugiyama, P. Nakov, and M. Y. Kan, "FANG: Leveraging social context for fake news detection using graph representation," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 1165–1174, 2020.

[68] D. Savage, X. Zhang, X. Yu, P. Chou, and Q. Wang, "Anomaly detection in online social networks," *Social Networks*, vol. 39, pp. 62–70, 2014.

[69] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li, "Spam review detection with graph convolutional networks," in *Proc. ACM International Conference on Information and Knowledge Management*, pp. 2703–2711, 2019.

[70] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, pp. 1–58, 2009.

[71] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 315–324, 2020.

[72] B. Branco, P. Abreu, A. S. Gomes, M. S. Almeida, J. T. Ascensão, and P. Bizarro, "Interleaved sequence rnns for fraud detection," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3101–3109, 2020.

[73] C. Liu, Q. Zhong, X. Ao, L. Sun, W. Lin, J. Feng, Q. He, and J. Tang, "Fraud transactions detection via behavior tree with local intention calibration," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3035–3043, 2020.

[74] J. Tang, F. Hua, Z. Gao, P. Zhao, and J. Li, "GADBench: Revisiting and benchmarking supervised graph anomaly detection," in *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[75] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, and L. Akoglu, "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 12012–12038, 2021.

[76] A. Z. Wang, R. Ying, P. Li, N. Rao, K. Subbian, and J. Leskovec, "Bipartite dynamic representations for abuse detection," in *Proc. ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 3638–3648, 2021.

[77] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "FRAUDAR: Bounding graph fraud in the face of camouflage," in *Proc. ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 895–904, 2016.

[78] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proc. Australasian Conference on Computer Science*, pp. 333–342, 2005.

[79] Z. Mingqiang, H. Hui, and W. Qian, "A graph-based clustering algorithm for anomaly intrusion detection," in *Proc. International Conference on Computer Science & Education*, pp. 1311–1314, 2012.

[80] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Berkeley Symposium on Mathematical Statistics & Probability*, vol. 1, pp. 281–297, 1967.

[81] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1641–1650, 2003.

[82] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 68–86, 1971.

[83] H. Tong and C.-Y. Lin, "Non-negative residual matrix factorization with application to graph anomaly detection," in *Proc. SIAM International Conference on Data Mining*, pp. 143–153, 2011.

[84] T. Idé and H. Kashima, "Eigenspace-based anomaly detection in computer systems," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 440–449, 2004.

[85] J. Wu, S. Pan, X. Zhu, and Z. Cai, "Boosting for multi-graph classification," *IEEE Transactions on Cybernetics*, vol. 45, pp. 416–429, 2014.

[86] X.-R. Sheng, D.-C. Zhan, S. Lu, and Y. Jiang, "Multi-view anomaly detection: Neighborhood in locality matters," in *Proc. AAAI Conference on Artificial Intelligence*, pp. 4894–4901, 2019.

[87] K. Ding, Q. Zhou, H. Tong, and H. Liu, "Few-shot network anomaly detection via cross-network meta-learning," in *Proc. ACM Web Conference*, pp. 2448–2456, 2021.

[88] S. Zhou, X. Huang, N. Liu, H. Zhou, F.-L. Chung, and L.-K. Huang, "Improving generalizability of graph anomaly detection models via data augmentation," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[89] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "GraphGAN: Graph representation learning with generative adversarial nets," in *Proc. AAAI Conference on Artificial Intelligence*, 2018.

[90] L. Meng, H. Mostafa, M. Nassar, X. Zhang, and J. Zhang, "Generative graph augmentation for minority class in fraud detection," in *Proc. ACM International Conference on Information and Knowledge Management*, pp. 4200–4204, 2023.

[91] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient GAN-based anomaly detection," *arXiv preprint arXiv:1802.06222*, 2018.

[92] H. Qiao, Q. Wen, X. Li, E.-P. Lim, and G. Pang, "Generative semi-supervised graph anomaly detection," *arXiv preprint arXiv:2402.11887*, 2024.

[93] Z. Gong, G. Wang, Y. Sun, Q. Liu, Y. Ning, H. Xiong, and J. Peng, "Beyond homophily: Robust graph anomaly detection via neural sparsification," in *Proc. International Joint Conference on Artificial Intelligence*, pp. 2104–2113, 2023.

[94] M. Jin, Y. Liu, Y. Zheng, L. Chi, Y.-F. Li, and S. Pan, "ANEMONE: Graph anomaly detection with multi-scale contrastive learning," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 3122–3126, 2021.

[95] Z. Xu, X. Huang, Y. Zhao, Y. Dong, and J. Li, "Contrastive attributed network anomaly detection with data augmentation," in *Proc. Pacific Asia Conference on Knowledge Discovery & Data Mining*, pp. 444–457, 2022.

[96] Y. Hu, C. Chen, B. Deng, Y. Lai, H. Lin, Z. Zheng, and J. Bian, "Decoupling anomaly discrimination and representation learning: self-supervised learning for anomaly detection on attributed graph," *Data Science and Engineering*, pp. 1–14, 2024.

[97] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart, "Representation learning for dynamic graphs: A survey," *Journal of Machine Learning Research*, vol. 21, no. 70, pp. 1–73, 2020.

[98] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN," in *Proc. International Joint Conference on Artificial Intelligence*, 2019.

[99] H. He, X. Li, P. Chen, J. Chen, W. Song, and Q. Xi, "DGFormer: An effective dynamic graph transformer based anomaly detection model for IoT time series," in *Proc. International Conference on Collaborative Computing*, pp. 173–188, 2023.

[100] J. Li, Q. Xing, Q. Wang, and Y. Chang, "CVTGAD: Simplified transformer with cross-view attention for unsupervised graph-level anomaly detection," in *Proc. Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 185–200, 2023.

[101] Y. Liu, S. Pan, Y. G. Wang, F. Xiong, L. Wang, Q. Chen, and V. C. Lee, "Anomaly detection in dynamic graphs via transformer," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 12081–12094, 2021.

[102] J. Langford and T. Zhang, "The epoch-greedy algorithm for multi-armed bandits with side information," in *Advances in Neural Information Processing Systems*, 2007.

[103] K. Ding, J. Li, and H. Liu, "Interactive anomaly detection on attributed networks," in *Proc. ACM International Conference on Web Search & Data Mining*, pp. 357–365, 2019.

[104] P. Morales, R. S. Caceres, and T. Eliassi-Rad, "Selective network discovery via deep reinforcement learning on embedded spaces," *Applied Network Science*, vol. 6, pp. 1–20, 2021.

[105] K. Ding, X. Shan, and H. Liu, "Towards anomaly-resistant graph neural networks via reinforcement learning," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 2979–2983, 2021.

[106] Y. Bei, S. Zhou, Q. Tan, H. Xu, H. Chen, Z. Li, and J. Bu, "Reinforcement neighborhood selection for unsupervised graph anomaly detection," in *Proc. IEEE International Conference on Data Mining*, pp. 11–20, 2023.

[107] Y. Li, X. Huang, J. Li, M. Du, and N. Zou, "SpecAE: Spectral autoencoder for anomaly detection in attributed networks," in *Proc. ACM International Conference on Information & Knowledge Management*, pp. 2233–2236, 2019.

[108] M. Mesgaran and A. B. Hamza, "A graph encoder–decoder network for unsupervised anomaly detection," *Neural Computing and Applications*, vol. 35, no. 32, pp. 23521–23535, 2023.

[109] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[110] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[111] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017.

[112] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 34, pp. 3438–3445, 2020.

[113] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *International conference on machine learning*, pp. 1725–1735, 2020.

[114] L. Xie, D. Pi, X. Zhang, J. Chen, Y. Luo, and W. Yu, "Graph neural network approach for anomaly detection," *Measurement*, vol. 180, p. 109546, 2021.

[115] J. Li, J. Li, Y. Liu, J. Yu, Y. Li, and H. Cheng, "Deconvolutional networks on graph data," in *Advances in Neural Information Processing Systems*, pp. 21019–21030, 2021.

[116] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 817–826, 2009.

[117] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: extraction and mining of academic social networks," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 990–998, 2008.

[118] Z. Peng, M. Luo, J. Li, H. Liu, Q. Zheng, *et al.*, "Anomalous: A joint modeling approach for anomaly detection on attributed networks," in *Proc. International Joint Conference on Artificial Intelligence*, pp. 3513–3519, 2018.

[119] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *International Conference on Learning Representations*, 2019.

[120] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019.

[121] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," *arXiv preprint arXiv:1903.02428*, 2019.

[122] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[123] K. Liu, Y. Dou, Y. Zhao, X. Ding, X. Hu, R. Zhang, K. Ding, C. Chen, H. Peng, K. Shu, *et al.*, "PyGOD: A python library for graph outlier detection," *arXiv preprint arXiv:2204.12095*, 2022.