Data-Efficient Graph Learning

by

Kaize Ding

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved February 2023 by the
Graduate Supervisory Committee:

Huan Liu, Chair
Guoliang Xue
Yezhou Yang
James Caverlee

ARIZONA STATE UNIVERSITY

May 2023

ABSTRACT

Graph-structured data, ranging from social networks to financial transaction networks, from citation networks to gene regulatory networks, have been widely used for modeling a myriad of real-world systems. As a prevailing model architecture to model graph-structured data, graph neural networks (GNNs) has drawn much attention in both academic and industrial communities in the past decades. Despite their success in different graph learning tasks, existing methods usually rely on learning from "big" data, requiring a large amount of labeled data for model training. However, it is common that real-world graphs are associated with "small" labeled data as data annotation and labeling on graphs is always time and resource-consuming. Therefore, it is imperative to investigate graph machine learning (GML) with minimal human supervision for the low-resource settings where limited or even no labeled data is available. In this proposal, we investigate a new research field – Data-Efficient Graph Learning, which aims to push forward the performance boundary of graph machine learning (GML) models with different kinds of low-cost supervision signals. To achieve this goal, we conduct a series of research for solving different graph minimally-supervised learning problems, including graph few-shot learning, graph weakly-supervised learning, and graph self-supervised learning.

ACKNOWLEDGMENTS

Nearing the completion of my Ph.D. program, I reflect on how quickly time has passed. Under the guidance of my advisor Prof. Huan Liu, I have grown from a new graduate to an experienced researcher in the field of machine learning and data mining. The experience in the last five years has not only shaped my career, but also greatly impacted the way I think, see, and approach my work as a researcher. I am grateful for not only the knowledge and skills I have gained, but also the memories of my research experiences and the friendships I have made. I am deeply thankful for the support and guidance of my professors, lab mates, and friends throughout my Ph.D. journey. Without them, my accomplishments would not have been possible.

Firstly, I would like to express my heartfelt gratitude to my advisor, Prof. Huan Liu, for his guidance and mentorship throughout my Ph.D. journey. The knowledge and skills I have gained from his advising is only a small part of what I have learned from him. His insights on high-level research directions, ideas about the intersection of multiple disciplines and fields, and strategies for producing impactful research have been great inspirations for me. Working in DMML led by Prof. Liu has been an amazing experience and as I have grown more senior in the lab, I have had the opportunity to teach with him, contribute to research grant proposals, and advise junior students. Through his guidance, he has imparted the necessary skills for me to become a successful researcher, and prepared me for the next step in my career.

I would also like to thank my committee members, Prof. James Caverlee, Prof. Guoliang Xue, and Prof. Yezhou Yang. Their expertise and insights have greatly contributed to the success of my dissertation and I am truly grateful for the time and effort you have invested in me. I wanna give a special thanks to Prof. James Caverlee, who provided invaluable help to me. I am honored to have had the opportunity to

work with each of you and I am incredibly grateful for the knowledge and experience I have gained through this process.

The journey of research is not without its challenges and difficulties. I faced many setbacks in my experiments and paper writing, resulting in countless sleepless nights. I know that I would not have been able to overcome these obstacles without the support and encouragement of my brilliant mentors and collaborators. For the first two years of my Ph.D., I was also honored to be mentored by Dr. Jundong Li, who is one of the greatest researchers and professors in data mining. He provided me with extensive assistance and helped me navigate many aspects of conducting research and writing papers. I was also grateful to Dr. Kai Shu, Ruochen Guo, Liang Wu, and all those successful previous DMMLers, who provided guidance during my Ph.D. and motivated me to work hard and smart as them.

I would also like to express my gratitude to all the amazing people I met during my internships at Amazon Alexa AI, Microsoft Research, and Google Brain. I would like to thank them for the opportunity to work with them and I am thankful for their mentorship, which has helped me to grow both personally and professionally.

Last but not least, I would like to thank my parents, who always give me unconditionally support and freedom to chase my dream. Despite being on opposite sides of the world, my parents have consistently demonstrated their love for me. Our weekly video chats have always been a source of happiness and comfort, erasing any negative feelings I may have had.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Motivation and Contributions

Recent years have witnessed a rapid growth in our ability to generate and gather data from numerous platforms in the online world and various sensors in the physical world. Graphs serve as a common language for modeling a plethora of structured and relational systems, such as social networks, knowledge graphs, and academic graphs, where entities are denoted as nodes while their relations are denoted as edges. More recently, graph learning algorithms, especially those based on graph neural networks (GNNs) (Shchur et al. 2018; Z. Wu et al. 2020) have received much research attention due to their significant impacts in addressing real-world problems. To harness the inherent structure among data, significant methodological advances have been made in graph learning, which have produced promising results in applications from diverse domains, ranging from cybersecurity (Zügner, Akbarnejad, and Günnemann 2018) to natural language processing (Ding, Wang, Li, Li, et al. 2020).

In general, existing graph learning algorithms focus on the setting where abundant human-annotated examples can be accessed during training. This assumption is often infeasible since collecting such auxiliary knowledge is laborious and requires intensive domain-knowledge, especially when considering the heterogeneity of graph-structured data (Yao et al. 2020a; Ding, Wang, Li, Shu, et al. 2020). As such, it is challenging yet imperative to study graph learning under different low-resource settings with limited or no labeled training data. In particular, three fundamental problems have drawn increasing research attention in the field of data-efficient graph learning: (1) *Graph Weakly-Supervised Learning* (Graph WSL), which focuses on learning effective GNNs for a specific down-stream task using either incomplete, or indirect, or inaccurate supervision signals; (2) *Graph Few-Shot Learning* (Graph FSL), whose goal is to handle unseen tasks (from novel label space) when only few labeled instances are available; and (3) *Graph Self-Supervised Learning* (Graph SSL), which aims to either pre-train task-agnostic GNNs or enhance GNNs on specific down-stream tasks without any semantic annotations. To address each of the aforementioned fundamental problems, we conduct a series of research to *push forward the performance boundary of graph machine learning (GML) models with different kinds of low-cost supervision signals*. In the meantime, we also investigate how to effectively adopt data-efficient GML algorithms to advance applications in different domains, such as cybersecurity, natural language processing, and recommendation.

In this dissertation, we attempt to learn effective GML models with minimal human supervision under different low-resource scenarios where only scarce labeled data is accessible. The main contributions of my research can be summarized as:

- Studying novel research problems in GML and define the formal formulation for those problems.
- Proposing novel frameworks to push forward the performance boundaries of GML models under the low-resource settings.
- Conducting extensive experiments on real-world datasets to demonstrate the effectiveness of the proposed frameworks.

1.2 Thesis Outline

This thesis mainly consists of three parts — (1) Graph Few-Shot Learning, (2) Graph Weakly-Supervised Learning, and (3) Graph Self-Supervised Learning.

Chapter 2 focuses on graph few-shot learning. For the problem of few-shot node classification, we propose Graph Prototypical Networks (GPN), which are designed to handle never-before-seen node classes in real-world graphs using only a handful of labeled data samples. GPN extrapolates meta-knowledge from many-shot seen node classes to few-shot unseen node classes using graph meta-learning. Next, I introduce Meta Graph Deviation Networks (Meta-GDN) which perform few-shot anomaly detection on a never-before-seen graph by learning from other auxiliary graphs within the same domain. Based on the proposed Graph Deviation Networks and Cross-network Meta Learning algorithm, Meta-GDN can transfer the knowledge from auxiliary graphs and quickly adapt to the target graph with few labeled anomalies.

Chapter 3 centers around my research in graph weakly-supervised learning. Specifically, we propose Meta Propagation Networks (Meta-PN) for semi-supervised learning with few labels. Meta-PN is able to adjust its label propagation strategy and leverage large receptive fields for inferring accurate pseudo labels on unlabeled nodes. By augmenting the scarce training data with the generated pseudo labels, one can learn a label-efficient GNN with only few labels per class. We also investigate how to leverage indirect supervision to solve the problem of graph anomaly detection. The proposed model Commander first compresses the two attributed graphs from different domains to low-dimensional space via a graph attentive encoder. In addition, we utilize a domain discriminator and an anomaly classifier to detect anomalies that appear across graphs from different domains. In order to further detect the anomalies that merely appear in the target graph, we develop an attribute decoder to provide additional signals for assessing node abnormality.

Chapter 4 focuses on graph self-supervised learning. I first investigate the bottleneck of unsupervised graph contrastive learning methods for learning expressive node representations without using any semantic labels. The proposed model S^3-CL, goes beyond the existing unsupervised graph contrastive methods and capture global graph knowledge from both structural and semantic perspectives. For the problem of graph anomaly detection, I developed a graph generative framework DOMINANT, which models the input attributed graph by reconstructing the structure and attribute information from the learned node representations. DOMINANT is based on the intuition that anomalies usually cannot be well reconstructed due to their deviations from the majority, thus the reconstruction errors are used to measure the abnormality of each node.

We will finally conclude this thesis in Chapter 5.

Chapter 2

GRAPH FEW-SHOT LEARNING

Given the rapidly evolving nature of real-world graphs, many practical graph applications require a Graph ML model to possess the capability of dealing with *never-before-seen* node classes (e.g., newly created user interest group), relations (e.g., newly extracted relations in knowledge graph) or even graphs (e.g., a protein-protein interaction graph from a new organism), using only a handful of labeled data samples. Despite that humans are capable of learning new tasks rapidly by utilizing what they learned in the past, current AI techniques cannot rapidly generalize from a few examples.

To bridge this gap between Graph ML models and humans, I have developed a series of graph few-shot learning algorithms, which show promising results and can shed light on following research. In this chapter, I will discuss my efforts on (i) the solution for handling the never-before-seen node classes in graphs. Specifically, I proposed *Graph Prototypical Networks (GPN)* to extrapolate the meta-knowledge from those many-shot seen node classes to the few-shot unseen node classes using graph meta-learning. Different from the few-shot learning endeavors in the image or text domain that simply assume all the labeled examples are of equal importance, GPN supports estimating the informativeness of each labeled node, by leveraging the graph property information (i.e., node centrality). In this way, GPN derives highly robust and representative class prototypes for classifying those never-before-seen node classes with even few labeled nodes; and (ii) the initial investigation on performing few-shot anomaly detection on a never-before-seen graph by learning from other similar graphs within the same domain. Specifically, I first designed a new family of graph neural networks – *Graph Deviation Networks (GDN)* that can leverage a small number of labeled anomalies for enforcing statistically significant deviations between abnormal and normal nodes on an arbitrary graph. By equipping it with a new cross-network meta-learning algorithm, the model is able to extract comprehensive meta-knowledge of anomalies from multiple auxiliary graphs and quickly to a new graph with only few labeled anomalies.

2.1 Few-Shot Node Classification

2.1.1 Introduction

Prevailing approaches for the node classification problem on attributed graph usually follow a supervised or semi-supervised paradigm, which typically relies upon the availability of sufficient labeled nodes for all the node classes (F. Zhou et al. 2019). Nonetheless, in many real-world attributed networks, a large portion of node classes only contain a limited number of labeled instances, rendering a long-tail distribution of node class labels. As shown in Figure 1, DBLP (J. Tang et al. 2008) is a dataset where nodes represent publications and node labels denote venues. Among all the node classes, more than 30% of them have less than 10 labeled instances. In the meantime, many practical applications require the learning models to possess the capability of dealing with such *few-shot* classes. A typical example is the intrusion detection problem (Northcutt and Novak 2002; Garcia-Teodoro et al. 2009) on traffic networks, where new attacks and threats are continuously being developed by adversaries. Due to the intensive labeling cost, for a specific type of attack, only a few examples can

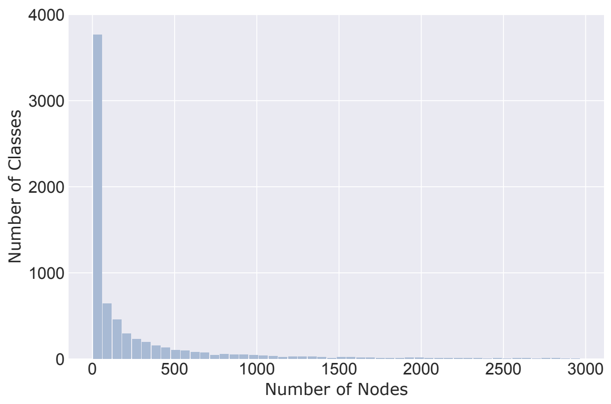


Figure 1. The Distribution of Labeled Nodes in the Real-world DBLP Dataset.

be accessed. Thus, understanding those attacks by type with limited labeled data is crucial for providing effective countermeasures. The shortage of labeled training data hinders existing node classification algorithms from learning an effective model with those *few-shot* node classes (F. Zhou et al. 2019; Qiao et al. 2019; Yao et al. 2020b). As such, it is challenging yet imperative to investigate the problem of node classification on attributed networks under the *few-shot* setting.

Recently, much research progress has been made in few-shot learning (FSL), for solving tasks (e.g., classification) with only a handful of labeled examples. In general, an FSL model learns across diverse *meta-training* tasks sampled from those classes with a large quantity of labeled data, and can be naturally generalized to a new task (i.e., *meta-test* task) from unseen classes during training. Such a *meta-learning* procedure enables the model to adapt knowledge from previous experiences, and has led to significant progress in FSL problems. Specifically, a major line of research such as siamese networks (Koch, Zemel, and Salakhutdinov 2015), matching networks (Vinyals et al. 2016), and relation networks (Sung et al. 2018) attempts to make the prediction by comparing the query instances and labeled examples in a shared metric space.

These *learning-to-compare* approaches have come into fashion due to their simplicity and effectiveness.

Despite their fruitful success, few-shot learning on attributed networks remains largely unexplored, mainly because of the following two challenges: **(i)** The process of constructing those *meta-training* tasks depends on the assumption that data is independent and identically distributed (*i.i.d.*), which is invalid on attributed networks. Apart from conventional text or image data, attributed networks lie in non-Euclidean space and encode the inherent dependency between nodes. Directly grafting existing methods is infeasible to capture the underlying data structure, making the embedded node representations less expressive. Thus how to exert the power of *meta-learning* on attributed networks is indispensable for extracting the meta-knowledge from data; **(ii)** Most of the existing FSL approaches simply assume that all the labeled examples are of equal importance for characterizing their belonged classes. However, neglecting the individual informativeness of labeled nodes will inevitably restrict the model performance on real-world attributed networks: On the one hand, it makes the FSL model highly vulnerable to noises or outliers since labeled data is severely limited (Ren, Triantafillou, et al. 2018; J. Zhang et al. 2019); on the other hand, it runs counter to the fact that the significance of a node could largely deviate from another. Intuitively, those central (core) nodes in a community are supposed to be more representative (Zhang and Wu 2012). Hence, how to capture the informativeness of each labeled node is the other challenge for building an effective few-shot classification model on attributed networks.

To address the aforementioned challenges, we present Graph Prototypical Networks (GPN), a graph meta-learning framework for solving the problem of few-shot node classification on attributed networks. Instead of classifying nodes directly, GPN tries to learn a transferable metric space in which the label of a node is predicted by finding the nearest class prototype. The proposed framework consists of two essential components that seamlessly work together for learning the prototype representation of each class. Specifically, the *network encoder* in GPN first compresses the input network to expressive node representations via graph neural networks (GNNs), in order to capture the data heterogeneity of an attributed network. Concurrently, another GNN-based *node valuator* is developed to estimate the informativeness of each labeled instance, by leveraging additional information encoded in the network. In this way, GPN derives highly robust and representative class prototypes. Moreover, by performing *meta-learning* across a pool of semi-supervised node classification tasks, GPN gradually extracts the meta-knowledge on an attributed network and further achieves better generalization ability on the target few-shot classification task. In summary, the main contributions of our work are as follows:

- ***Problem***: We investigate the novel problem of few-shot node classification on attributed networks. In particular, we emphasize its importance in real-world applications and further provide a formal problem definition.
- ***Algorithm***: We propose a principled framework GPN for the problem, which exploits graph neural networks and meta-learning to learn a powerful few-shot node classification model on attributed networks.
- ***Evaluation***: We perform extensive experiments on various real-world datasets to corroborate the effectiveness of our approach. The experimental results demonstrate the superior performance of GPN for few-shot node classification on attributed networks.

2.1.2 Related Work

Few-shot learning (FSL) aims to solve new tasks with a limited number of examples, based on the knowledge obtained from previous experiences. Generally, existing FSL models fall into two broad categories: (1) *optimization-based approaches*, which focus on learning the optimization of model parameters given the gradients on few-shot examples (Ravi and Larochelle 2017; Finn, Abbeel, and Levine 2017; Zhenguo Li et al. 2017; Mishra et al. 2018). One example is the LSTM-based meta-learner (Ravi and Larochelle 2017), which aims to learn efficient parameter updating rules for training a neural classifier. MAML (Finn, Abbeel, and Levine 2017) learns the parameter initialization that is suitable for different FSL tasks and is compatible with any model trained with gradient descent. Meta-SGD (Zhenguo Li et al. 2017) goes further in meta-learning by arguing to learn the weights initialization, gradient update direction and learning rate within a single step. SNAIL (Mishra et al. 2018) is another model which combines temporal convolution and soft attention to learn an optimal learning strategy. However, this line of work usually suffers from the computational cost of fine-tuning. (2) *metric-based approaches*, which try to learn generalizable matching metrics between query and support set across different tasks (Vinyals et al. 2016; Snell, Swersky, and Zemel 2017; Ren, Triantafillou, et al. 2018; Sung et al. 2018; Liu, Zhou, Long, Jiang, and Zhang 2019). For instance, Matching Networks (Vinyals et al. 2016) learn a weighted nearest-neighbor classifier with attention networks. Prototypical Network (Snell, Swersky, and Zemel 2017) computes the prototype of each class by taking the mean vector of support examples and classifies query instances by calculating their Euclidean distances. An extension of Prototypical Networks proposed by Ren et al. (Ren, Triantafillou, et al. 2018) considers both labeled and unlabeled data for few-shot learning. Relation Network (Sung et al. 2018) trains an auxiliary network to learn a non-linear metric between each query and the support set. It is worth mentioning that our approach also follows this paradigm due to its simplicity and effectiveness. Recently, few-shot learning on graphs has received increasing research attention (F. Zhou et al. 2019; Bose et al. 2019). However, those methods treat support examples equally, rendering the model unstable to noises or outliers (Deng et al. 2020). In this work, we learn a robust and powerful few-shot learning model by considering the individual importance of labeled support examples.

2.1.3 Problem Definition

Following the commonly used notations, we use calligraphic fonts to denote sets (e.g., $\mathcal{V}$), bold lowercase letters (e.g., $\mathbf{x}$) to denote vectors and bold uppercase letters for matrices (e.g., $\mathbf{X}$). The i^{th} row of a matrix $\mathbf{X}$ is denoted by $\mathbf{x}_i$ and the $(i,j)^{th}$ element of matrix $\mathbf{X}$ is denoted by $\mathbf{X}_{i,j}$. Besides, we represent the identity matrix as $\mathbf{I}$, and the transpose of a matrix $\mathbf{X}$ is represented as $\mathbf{X}^{\mathrm{T}}$. The ℓ_2-norm of a vector is denoted by $||\cdot||_2$. The Frobenius norm of a matrix is represented by $||\cdot||_F$. Accordingly, we define the attributed network as follows:

Definition 1 ***Attributed Networks****: An attributed network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ consists of: (1) the set of nodes $\mathcal{V} = \{v_1, v_2, ..., v_n\}$, where $|\mathcal{V}| = n$; (2) the set of edges $\mathcal{E}$, where $|\mathcal{E}| = m$; and (3) the node attributes $\mathbf{X} \in \mathbb{R}^{n \times d}$, where the i^{th} row vector $\mathbf{x}_i \in \mathbb{R}^d$ $(i = 1, ..., n)$ is the attribute[1] information for the i^{th} node. The topological structure of*

[1]Attribute and feature are used interchangeably.

attributed network $\mathcal{G}$ can be represented by an adjacency matrix $\mathbf{A}$, where $\mathbf{A}_{i,j} = 1$ if there is a link between node v_i and node v_j. Otherwise, $\mathbf{A}_{i,j} = 0$.

Note that the aforementioned notations and definition are also used throughout the dissertation. And the studied problem in this section can be formulated as follows:

Problem Definition 1 ***Few-shot Node Classification on Attributed Networks****: Given an attributed network $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$, suppose we have substantial labeled nodes for a set of node classes C_{train}. After training on the labeled data from C_{train}, the model is tasked to predict labels for the nodes (i.e., query set $\mathcal{Q}$) from a disjoint set of node classes C_{test}, for which only a few labeled nodes of each class (i.e., support set $\mathcal{S}$) are available.*

Following the common setting in FSL, if C_{test} consists of N classes and the support set $\mathcal{S}$ includes K labeled nodes per class, this problem is named N-way K-shot node classification problem. In essence, the objective of this problem is to learn a meta-classifier that can be adapted to new classes with only a few labeled nodes. Therefore, how to extract transferable meta-knowledge from C_{train} is the key for solving the studied problem.

2.1.4 Methodology and Model Design

As existing FSL models are not tailored for graph-structured data, it is infeasible to apply them to solve the studied problem directly. In this section, we present the details about the proposed Graph Prototypical Networks (GPN) for few-shot node classification on attributed networks. Specifically, our framework is designed and built to address three challenging research questions: (1) How to perform *meta-learning* on

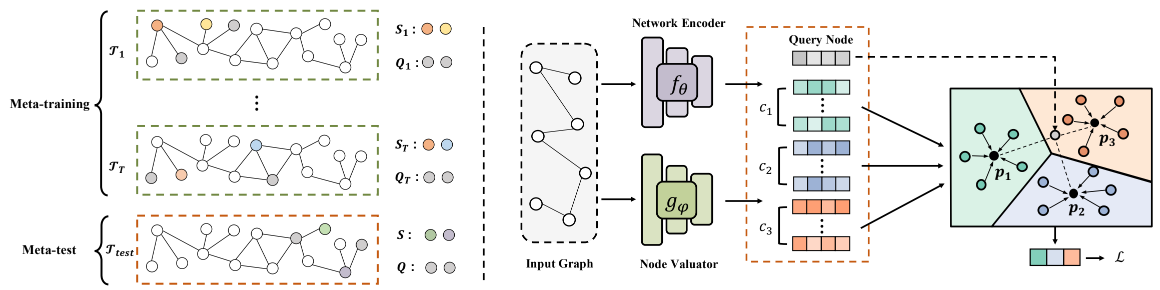


Figure 2. (Left) Episodic Training on Attributed Networks; (Right) The Architecture of the Proposed Framework Graph Prototypical Networks (GPN).

attributed networks (*non-i.i.d.* data) for extracting the meta-knowledge? (2) How to learn expressive node representations from the input attributed network by considering both the node attributes and topological structure? and (3) How to identify the informativeness of each labeled node for learning robust and discriminative class representations?

An overview of the proposed Graph Prototypical Networks (GPN) is provided in Figure 2. In Section 2.1.4.1, we introduce the backbone training mechanism of the proposed model. In Section 2.1.4.2 and 2.1.4.3, we introduce how we design the two essential modules in GPN. Then we discuss how to perform few-shot node classification using the proposed framework in Section 2.1.4.4.

2.1.4.1 Episodic Training on Attributed Networks

Our approach is a meta-learning framework which follows the prevailing episodic training paradigm (Vinyals et al. 2016). Specifically, GPN learns over diverse *meta-training* tasks in a large number of episodes rather than only on the target *meta-test* task. The key idea of episodic training is to mimic the real test environment by sampling nodes from C_{train}. The consistency between training and test environment alleviates the distribution gap and improves model generalization capability. Specifically, in each episode, we construct a N-way K-shot meta-training task:

$$\begin{aligned}
&\mathcal{S}_t = \{(v_1, y_1), (v_2, y_2), ..., (v_{N\times K}, y_{N\times K})\},\\
&\mathcal{Q}_t = \{(v_1^*, y_1^*), (v_2^*, y_2^*), ..., (v_{N\times M}^*, y_{N\times M}^*)\},\\
&\mathcal{T}_t = \{\mathcal{S}_t, \mathcal{Q}_t\},
\end{aligned} \tag{2.1}$$

where both the support set $\mathcal{S}_t$ and query set $\mathcal{Q}_t$ of the meta-training task $\mathcal{T}_t$ are sampled from C_{train}. The support set $\mathcal{S}_t$ contains K nodes from each class, while the query set $\mathcal{Q}_t$ includes M query nodes sampled from the remainder of each of the N classes.

The whole training process is based on a set of T meta-training tasks $\mathcal{T}_{train} = \{\mathcal{T}_t\}_{t=1}^{T}$. The model is trained to minimize the loss of its predictions for the query set $\mathcal{Q}_t$ in each meta-training task $\mathcal{T}_t$, and goes episode by episode until convergence. In this way, the model gradually collects meta-knowledge across those meta-training tasks and then can be naturally generalized to the meta-test task $\mathcal{T}_{test} = \{\mathcal{S}, \mathcal{Q}\}$ with unseen classes C_{test}.

Different from conventional episodic training that constructs a pool of supervised *meta-training* tasks (Garcia and Bruna 2018), in each episode, we sample N-way K-shot labeled nodes and mask the rest as unlabeled nodes. In this way, we can create a semi-supervised *meta-training* task with the partially labeled attributed network. By considering both labeled and unlabeled data and their dependencies, we are able to learn more expressive node representations for few-shot node classification during the *meta-learning* process.

2.1.4.2 Network Representation Learning

In order to learn expressive node representations from an attributed network, we develop a *network encoder* to capture the data heterogeneity. Specifically, the *network encoder* possesses a GNN backbone, which converts each node to a low-dimensional latent representation. In general, GNNs follow the neighborhood aggregation scheme, and compute the node representations by recursively aggregating and compressing node features from local neighborhoods. Briefly, a GNN layer can be defined as:

$$\begin{aligned} \mathbf{h}_i^l &= \textsc{Combine}^l\Big(\mathbf{h}_i^{l-1}, \mathbf{h}_{\mathcal{N}_i}^l\Big), \\ \mathbf{h}_{\mathcal{N}_i}^l &= \textsc{Aggregate}^l\Big(\{\mathbf{h}_j^{l-1} | \forall j \in \mathcal{N}_i \cup v_i\}\Big), \end{aligned} \tag{2.2}$$

where $\mathbf{h}_i^l$ is the node representation of node i at layer l and $\mathcal{N}_i$ is the set of neighboring nodes of v_i. Combine and Aggregate are two key functions of GNNs and have a series of possible implementations (Thomas N. Kipf and Welling 2017b; Hamilton, Ying, and Leskovec 2017; Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018a).

By stacking multiple GNN layers in the *network encoder*, the learned node representations are able to capture the long-range node dependencies in the network:

$$\begin{aligned} &\mathbf{H}^1 = \mathrm{GNN}^1(\mathbf{A}, \mathbf{X}), \\ &\dots \\ &\mathbf{Z} = \mathrm{GNN}^L(\mathbf{A}, \mathbf{H}^{L-1}), \end{aligned} \tag{2.3}$$

where $\mathbf{Z}$ is the learned node representations from the *network encoder*. For simplicity, we will use $f_{\boldsymbol{\theta}}(\cdot)$ to denote the *network encoder* with L GNN layers.

Prototype Computation. With the learned node representations from the *network encoder*, next, we aim to compute the representation of each class with the labeled nodes from the support set. We follow the idea of Prototypical Networks (Snell, Swersky, and Zemel 2017), which encourages nodes of each class cluster around a specific prototype representation. Formally, the class prototypes can be computed by:

$$\mathbf{p}_c = \textsc{Proto}\Big(\{\mathbf{z}_i | \forall i \in \mathcal{S}_c\}\Big), \tag{2.4}$$

where $\mathcal{S}_c$ denotes the set of labeled examples from class c and Proto is the prototype computation function. For instance, in the vanilla Prototypical Networks (Snell, Swersky, and Zemel 2017), the prototype of each class is computed by taking the average of all embedded nodes belonging to that class:

$$\mathbf{p}_c = \frac{1}{|\mathcal{S}_c|} \sum_{i \in \mathcal{S}_c} \mathbf{z}_i. \tag{2.5}$$

2.1.4.3 Node Importance Valuation

Despite its simpleness, directly taking the mean vectors of the embedded support instances as prototypes may not provide promising results for our problem. It not only neglects the fact that each node has a different significance in a network, but also makes the FSL model highly noise-sensitive since labeled data is severely limited (J. Zhang et al. 2019). Therefore, refining those class prototypes becomes especially essential for building a robust and effective FSL model.

To identify the informativeness of each labeled node, we adopt a view that the importance of a node is highly correlated with its neighbors' importance (N. Park et al. 2019). Accordingly, we design a GNN-based *node valuator* $g_\phi(\cdot)$ (as shown in Figure 3) to estimate node importance scores through a *score aggregation layer*, which can be defined as follows:

$$s_i^l = \sum_{j \in \mathcal{N}_i \cup v_i} \alpha_{ij}^l s_j^{l-1}, \tag{2.6}$$

where s_i^l is the importance score of node v_i in the l-th layer ($l = 1, \dots, L$). α_{ij}^l is the attention weight between nodes v_i and v_j, we compute it via a shared attention mechanism:

$$\alpha_{ij}^l = \frac{\exp\big(\mathrm{LeakyReLU}\big(\mathbf{a}^{\mathrm{T}}[s_i^{l-1} || s_j^{l-1}]\big)\big)}{\sum_{k \in \mathcal{N}_i \cup v_i} \exp\big(\mathrm{LeakyReLU}\big(\mathbf{a}^{\mathrm{T}}[s_j^{l-1} || s_k^{l-1}]\big)\big)}, \tag{2.7}$$

where $||$ is a concatenation operator and $\mathbf{a}$ is a weight vector.

To compute the initial importance score s_i^0, we employ a *scoring layer* to compress the node features. Our *scoring layer* is a feed-forward layer with tanh non-linearity. Specifically, the initial score of node v_i is computed by:

$$s_i^0 = \tanh(\mathbf{w}_s^{\mathrm{T}} \mathbf{x}_i + b_s) \tag{2.8}$$

where $\mathbf{w}_s \in \mathbb{R}^d$ is a learnable weight vector and $b_s \in \mathbb{R}^1$ is the bias.

Centrality Adjustment. As suggested in previous research on node importance estimation (Page et al. 1999; N. Park et al. 2019) , the importance of a node positively correlates with its centrality in the graph. Given that the in-degree $\deg(i)$ of node v_i is a common proxy for its centrality and popularity, we define the initial centrality $C(i)$ of node v_i as:

$$C(i) = \log(\deg(i) + \epsilon), \tag{2.9}$$

where ϵ is a small constant. To compute the final importance score, we apply centrality adjustment to the estimated score s_i^L from the last layer, and apply a sigmoid non-linearity as follows:

$$\tilde{s}_i = \mathrm{sigmoid}(C(i) \cdot s_i^L). \tag{2.10}$$

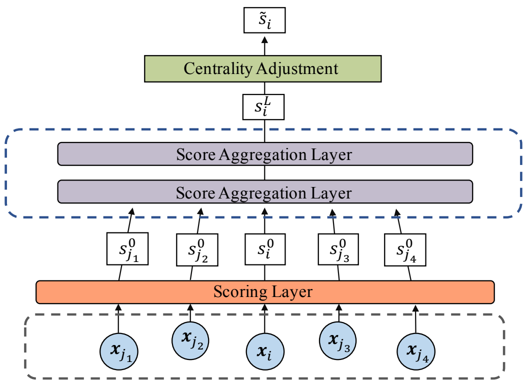

$\tilde{s}_i$
Centrality Adjustment
s_i^L
Score Aggregation Layer
Score Aggregation Layer
$s_{j_1}^0$ $s_{j_2}^0$ s_i^0 $s_{j_3}^0$ $s_{j_4}^0$
Scoring Layer
$\boldsymbol{x}_{j_1}$ $\boldsymbol{x}_{j_2}$ $\boldsymbol{x}_i$ $\boldsymbol{x}_{j_3}$ $\boldsymbol{x}_{j_4}$

Figure 3. Architecture of the Node Valuator.

2.1.4.4 Few-shot Node Classification

After we compute the importance score of each support node, we first normalize those scores using the softmax function:

$$\beta_i = \frac{\exp(\tilde{s}_i)}{\sum_{k \in \mathcal{S}_c} \exp(\tilde{s}_k)}, \tag{2.11}$$

where β_i represents the normalized weight of each support node v_i, then the refined prototypes can be directly computed by:

$$\mathbf{p}_c = \sum_{i \in \mathcal{S}_c} \beta_i \mathbf{z}_i. \tag{2.12}$$

As such, our model can adjust the cluster locations to better represent the examples in both the support and unlabeled sets. These learned prototypes define a predictor for the class label of a query node v_i^*, which assigns a probability over each class c based on the distances between the query node v_i^* and each prototype:

$$p(c|v_i^*) = \frac{\exp(-d(\mathbf{z}_i^*, \mathbf{p}_c))}{\sum_{c'} \exp(-d(\mathbf{z}_i^*, \mathbf{p}_{c'}))}, \tag{2.13}$$

where $d(\cdot)$ is a distance metric function. Commonly, squared Euclidean distance is a simple and effective choice (Snell, Swersky, and Zemel 2017).

Under the episodic training framework, the objective of each meta-training task is to minimize the classification loss between the predictions of the query set and the ground-truth. Specifically, the training loss can be defined as the average negative log-likelihood probability of assigning correct class labels:

$$\mathcal{L} = -\frac{1}{N \times M} \sum_{i=1}^{N \times M} \log p(y_i^* | v_i^*). \tag{2.14}$$

By minimizing the above loss function, GPN is able to learn a generic classifier for a specific meta-training task. Training episodes are formed by randomly selecting a subset of classes from the auxiliary class set C_{train}, then choosing a subset of nodes within each class to act as the support set and a subset of the remainder to serve as query set. After training on a considerable number of meta-training tasks, its generalization performance will be measured on the test episodes, which contain nodes sampled from $\mathcal{C}_{test}$ instead of $\mathcal{C}_{train}$. For each test episode, we use the predictor produced by our GPN for the provided support set $\mathcal{S}$ to classify each query node in $\mathcal{Q}$ into the most likely class: $\hat{y}_i^* = \text{argmax}_c p(c|v_i^*)$.

2.1.5 Performance Evaluation

2.1.5.1 Evaluation Settings

Evaluation Datasets. Due to the fact that few-shot node classification on graph-structured data remains an under-studied problem, it is worth mentioning that the existing benchmark datasets (e.g., Cora, Pubmed) for conventional node classification problem are not suitable for evaluating FSL models. The main reason is that FSL models usually need to be tested on many different classification tasks, while those datasets only contain limited node classes. To extensively evaluate the model per-

Table 1. Statistics of the Evaluation Datasets.

Datasets	# nodes	# edges	# attributes	# Train/Valid/Test
Amazon-Clothing	24,919	91,680	9,034	40/17/20
Amazon-Electronics	42,318	43,556	8,669	90/37/40
DBLP	40,672	288,270	7,202	80/27/30
Reddit	232,965	11,606,919	602	16/10/15

formance on few-shot node classification, in our experiments, we adopt four public datasets with plenty of node classes and their statistics of can be found in Table 1. **Amazon-Clothing** and **Amazon-Electronics** (McAuley, Pandey, and Leskovec 2015) are two product networks built with the products in "Clothing, Shoes and Jewelry" and "Electronics" on Amazon respectively. In these networks, each product is considered as a node and its description is used to construct the node attributes. We use the substitutable and complementary relationship to create links between products. The class label is defined as the low-level product category. **DBLP** (J. Tang et al. 2008) is a citation network where each node represents a paper, and the links are the citation relations among different papers. The paper abstracts are used to construct node attributes. The class label of a node is defined as the paper venue. **Reddit** (Hamilton, Ying, and Leskovec 2017) is a post-to-post graph constructed with data sampled from Reddit, which is used to evaluate the performance of our model on large-scale attributed networks. In this large-scale attributed network, posts are represented by nodes and two posts are connected if they are commented by the same user. Each post is labeled with it a community ID.

Compared Methods. In the experiments, we compare the proposed model GPN with related baseline methods: **DeepWalk** (Perozzi, Al-Rfou, and Skiena 2014) learns node embeddings from a stream of truncated vanilla random walks on the input graph, and **node2vec** (Grover and Leskovec 2016) extends it with biased random walks to explore diverse neighborhoods. **GCN** (Thomas N. Kipf and Welling 2017b) learns latent node representations based on the first-order approximation of spectral graph convolutions. **SGC** (F. Wu et al. 2019) eliminates the non-linearity between GCN layers and folding the convolution functions into a linear transformation. Prototypical Network (**PN**) (Snell, Swersky, and Zemel 2017) is one of the widely used few-shot learning methods for image classification. **MAML** (Finn, Abbeel, and Levine 2017) is an optimization-based meta-learning method, which tries to learn a better model initialization from a series of meta-training tasks. By using a GNN base model, **Meta-GNN** (F. Zhou et al. 2019) extends MAML to graph data.

2.1.5.2 General Comparisons

For each dataset, we evaluate the performance of all the algorithms on four few-shot node classification tasks, i.e., 5-way-3-shot, 5-way-5-shot, 10-way-3-shot, and 10-way-5-shot. We set the query size as same as the support size in our experiments. We adopt two widely used metrics Accuracy (ACC) and Micro-F1 (F1) to evaluate performance. Each model is evaluated on 50 *meta-test* tasks and each *meta-test* task is randomly sampled from test node classes. We repeat the process 10 times and the averaged results are presented in Table 8 and 9. Higher values are better for all metrics. From the comprehensive views, we make the following observations:

- A general observation is that our approach GPN achieves the best performance on all the few-shot tasks. For example, on the Amazon-Clothing dataset, GPN outperforms the best performing baseline Meta-GNN by 5.9% (ACC) under the

Table 2. Averaged Few-shot Node Classification Results on Amazon-Clothing and Amazon-Electronics (%).

Methods	Amazon-Clothing							
	5-way 3-shot		5-way 5-shot		10-way 3-shot		10-way 5-shot	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk	36.7	36.3	46.5	46.6	21.3	19.1	35.3	32.9
node2vec	36.2	35.8	41.9	40.7	17.5	15.1	32.6	30.2
GCN	54.3	51.4	59.3	56.6	41.3	37.5	44.8	40.3
SGC	56.8	55.2	62.2	61.5	43.1	41.6	46.3	44.7
PN	53.7	53.6	63.5	63.7	41.5	41.9	44.8	46.2
MAML	55.2	54.5	66.1	67.8	45.6	43.3	46.8	45.6
Meta-GNN	74.1	73.6	77.3	77.5	61.4	59.7	64.2	62.9
GPN	**75.4**	**74.7**	**78.6**	**79.0**	**65.0**	**66.1**	**67.7**	**68.9**

Methods	Amazon-Electronics							
	5-way 3-shot		5-way 5-shot		10-way 3-shot		10-way 5-shot	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk	23.5	22.2	26.1	25.7	14.7	12.9	16.0	14.7
node2vec	25.5	23.7	27.1	24.3	15.1	13.1	17.7	15.5
GCN	53.8	49.8	59.6	55.3	42.3	38.4	47.4	48.3
SGC	54.6	53.4	60.8	59.4	43.2	41.5	50.0	47.6
PN	53.5	55.6	59.7	61.5	39.9	40.0	45.0	44.8
MAML	53.3	52.1	59.0	58.3	37.4	36.1	43.4	41.3
Meta-GNN	63.2	61.5	67.9	66.8	58.2	55.8	60.8	60.1
GPN	**64.6**	**62.8**	**70.9**	**70.6**	**60.3**	**60.7**	**62.4**	**63.7**

10-way-3-shot task. The improvements are even more substantial on the larger dataset Reddit. This result verifies that GPN is a powerful and reliable model to tackle the problem of few-shot node classification on attributed networks.

- Overall, DeepWalk and node2vec largely fall behind other methods on few-shot node classification tasks. Those random walk-based methods need to train a supervised classifier (e.g., Logistic Regression) with learned node representations, which typically rely on a large number of labeled data for good performance. Similarly, GNN-based methods are unable to obtain competitive results on the few-shot node classification problem. Conventional GNN models are developed for

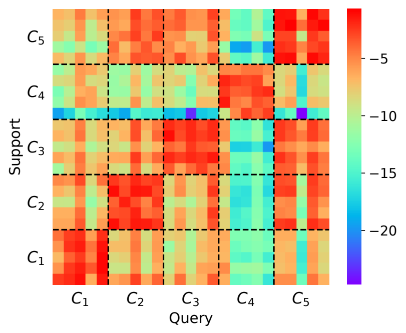
Table 3. Averaged Few-shot Node Classification Results on DBLP and Reddit (%).

Methods	**DBLP**							
	5-way 3-shot		5-way 5-shot		10-way 3-shot		10-way 5-shot	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk	44.7	43.1	62.4	60.4	33.8	30.8	45.1	43.0
node2vec	40.7	38.5	58.6	57.2	31.5	27.8	41.2	39.6
GCN	59.6	54.9	68.3	66.0	43.9	39.0	51.2	47.6
SGC	57.3	54.7	65.0	62.1	40.2	36.8	50.3	46.4
PN	37.2	36.7	43.4	44.3	26.2	26.0	32.6	32.8
MAML	39.7	39.7	45.5	43.7	30.8	25.3	34.7	31.2
Meta-GNN	70.9	70.3	78.2	78.2	60.7	60.4	68.1	67.2
GPN	**74.5**	**73.9**	**80.1**	**79.8**	**62.6**	**62.6**	**69.0**	**69.4**

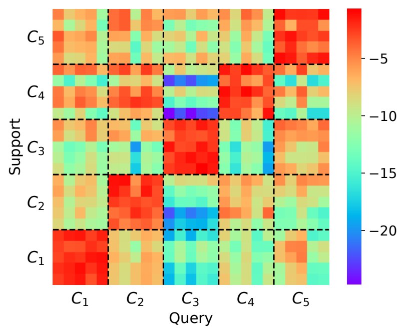
Methods	**Reddit**							
	5-way 3-shot		5-way 5-shot		10-way 3-shot		10-way 5-shot	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk	26.7	26.1	30.1	29.7	17.6	17.1	18.8	18.6
node2vec	27.1	25.6	31.2	29.8	19.8	18.6	23.4	22.6
GCN	38.8	38.1	45.5	44.1	29.0	27.0	35.7	32.4
SGC	44.4	42.1	46.8	42.5	29.7	26.8	31.6	27.7
PN	34.6	33.3	37.6	36.4	19.8	18.0	23.3	21.4
MAML	29.1	26.8	31.1	29.7	15.2	12.2	17.9	15.6
Meta-GNN	60.8	58.3	62.7	61.2	44.9	42.1	51.5	47.1
GPN	**65.5**	**66.2**	**68.4**	**69.0**	**53.4**	**55.8**	**57.7**	**59.2**

semi-supervised node classification, and could be easily overfitted with only a small number of labeled instances.

- Despite the success of MAML and PN on few-shot image classification, however, both of them perform poorly on our tasks. The main reason is that those methods cannot capture the dependency between nodes for learning expressive node representations, rendering unsatisfactory performance on few-shot node classification tasks.
- By integrating the idea of meta-learning into graph neural networks, Meta-GNN is able to achieve considerable improvements over other baseline methods on few-shot node classification in most cases. However, it is worth noting that its performance suffers a catastrophic decline on the Reddit dataset. One reasonable explanation is



(a) **Meta-GNN**



(b) **GPN**

Figure 4. Similarity Matrix on DBLP Dataset (5-way 5-shot).

that optimization-based FSL approaches require extensive fine-tuning efforts for the target task, especially on those large-scale datasets.

2.1.5.3 Case Study

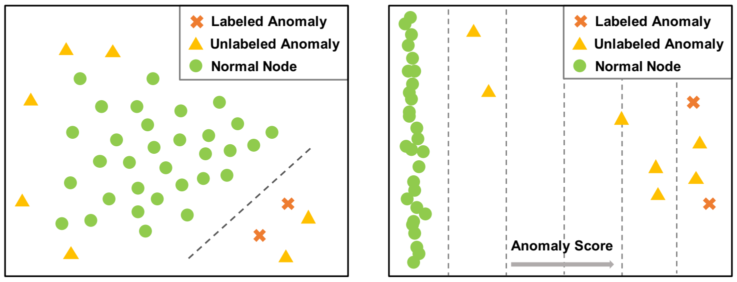
Figure 4 shows the similarity matrix learned by the best performing baseline Meta-GNN and our approach on the DBLP dataset, with the same network encoder in a 5-way 5-shot task. Here we use the negative Euclidean distance as the similarity metric. Specifically, each cell consists of 5×5 grids illustrating the divergence between two classes, as well as the intra-class similarities. To better visualize the results, for GPN, we use the weighted embedding of each support node instead of computing the class prototype. From the figure, we can observe that GPN can better capture the similarities between the support nodes and query nodes from a same class, which validates the robustness and effectiveness of our approach.

2.2 Few-Shot Graph Anomaly Detection

2.2.1 Introduction

Network-structured data, ranging from social networks (Zafarani, Abbasi, and Liu 2014) to team collaboration networks (Zhou, Li, and Tong 2019), from citation networks (J. Tang et al. 2008) to molecular graphs (J. You et al. 2018), has been widely used in modeling a myriad of real-world systems. Nonetheless, real-world networks are commonly contaminated with a small portion of nodes, namely, anomalies, whose patterns significantly deviate from the vast majority of nodes (Ding, Li, and Liu 2019; Ding, J. Li, et al. 2020; Dawei Zhou et al. 2018). For instance, in a citation network that represents citation relations between papers, there are some research papers with a few spurious references (i.e., edges) which do not comply with the content of the papers (Bandyopadhyay, Lokesh, and Murty 2019); In a social network that represents friendship of users, there may exist camouflaged users who randomly follow different users, rendering properties like homophily not applicable to this type of relationships (Dou et al. 2020). As the existence of even few abnormal instances could cause extremely detrimental effects, the problem of network anomaly detection has received much attention in industry and academy alike.

Due to the fact that labeling anomalies is highly labor-intensive and takes specialized domain-knowledge, existing methods are predominately developed in an unsupervised manner. As a prevailing paradigm, people try to measure the abnormality of nodes with the reconstruction errors of autoencoder-based models (Ding et al. 2019; Y. Li et al. 2019) or the residuals of matrix factorization-based methods (Tong and Lin 2011; Li, Dani, Hu, and Liu 2017; Bandyopadhyay, Lokesh, and



(a) Latent Representation Space (b) Anomaly Score Space

Figure 5. Since anomalies usually have distinct patterns, (a) existing methods may easily fail to distinguish them from normal nodes in the latent representation space with only few labeled anomalies, (b) while they can be well separated in an anomaly score space by enforcing statistically significant deviations between abnormal and normal nodes.

Murty 2019). However, the anomalies they identify may turn out to be data noises or uninteresting data instances due to the lack of prior knowledge on the anomalies of interest. A potential solution to this problem is to leverage limited or few labeled anomalies as the prior knowledge to learn anomaly-informed models, since it is relatively low-cost in real-world scenarios – a small set of labeled anomalies could be either from a deployed detection system or be provided by user feedback. In the meantime, such valuable knowledge is usually scattered among other networks within the same domain of the target one, which could be further exploited for distilling supervised signal. For example, LinkedIn and Indeed have similar social networks that represent user friendship in the job-search domain; ACM and DBLP can be treated as citation networks that share similar citation relations in the computer science domain. According to previous studies (X. Tang et al. 2020; F. Zhou et al. 2020; Q. Zhou et al. 2019), because of the similarity of topological structure and nodal attributes, it is feasible to transfer valuable knowledge from source network(s) to the target network so that the performance on the target one is elevated. As such, in this work we propose to investigate the novel problem of few-shot network anomaly detection under the cross-network setting.

Nonetheless, solving this under-explored problem remains non-trivial, mainly owing to the following reasons: **(1)** From the micro (*intra-network*) view, since we only have limited knowledge of anomalies, it is hard to precisely characterize the abnormal patterns. If we directly adopt existing semi-supervised (D. Wang et al. 2019) or PU (M. Wu et al. 2019) learning techniques, those methods often fall short in achieving satisfactory results as they might still require a relatively large percentage of positive examples (Pang, Shen, and Hengel 2019). To handle such incomplete supervision challenge (Z.-Y. Zhang et al. 2019) as illustrated in Figure 5a, instead of focusing on abnormal nodes, how to leverage labeled anomalies as few as possible to learn a high-level abstraction of normal patterns is necessary to be explored; **(2)** From the macro (*inter-network*) view, though networks in the same domain might share similar characteristics in general, anomalies exist in different networks may be from very different manifolds. Previous studies on cross-network learning (M. Wu et al. 2020; Shen et al. 2020) mostly focus on transferring the knowledge only from a single network, which may cause unstable results and the risk of negative transfer. As learning from multiple networks could provide more comprehensive knowledge about the characteristics of anomalies, a cross-network learning algorithm that is capable of adapting the knowledge is highly desirable.

To address the aforementioned challenges, in this work we first design a new GNN architecture, namely Graph Deviation Networks (GDN), to enable network anomaly detection with limited labeled data. Specifically, given an arbitrary network, GDN first uses a GNN-backboned anomaly score learner to assign each node with an anomaly score, and then defines the mean of the anomaly scores based on a prior probability to serve as a reference score for guiding the subsequent anomaly score learning. By leveraging a deviation loss (Pang, Shen, and Hengel 2019), GDN is able to enforce statistically significant deviations of the anomaly scores of anomalies from that of normal nodes in the anomaly score space (as shown in Figure 5b). To further transfer this ability from multiple networks to the target one, we propose a cross-network meta-learning algorithm to learn a well-generalized initialization of GDN from multiple few-shot network anomaly detection tasks. The seamlessly integrated framework Meta-GDN is capable of extracting comprehensive meta-knowledge for detecting anomalies across multiple networks, which largely alleviates the limitations of transferring from a single network. Subsequently, the initialization can be easily adapted to a target network via fine-tuning with few or even one labeled anomaly, improving the anomaly detection performance on the target network to a large extent. To summarize, our main contributions is three-fold:

- ***Problem***: To the best of knowledge, we are the first to investigate the novel problem of few-shot network anomaly detection. Remarkably, we propose to solve this problem by transferring the knowledge across multiple networks.
- ***Algorithms***: We propose a principled framework Meta-GDN, which integrates a new family of graph neural networks (i.e., GDN) and cross-network meta-learning to detect anomalies with few labeled instances.
- ***Evaluations***: We perform extensive experiments to corroborate the effectiveness of Meta-GNN. The experimental results demonstrate its superior performance over the state-of-the-art methods on network anomaly detection.

2.2.2 Related Work

Network anomaly detection methods have a specific focus on the network structured data. Previous research mostly study the problem of anomaly detection on plain networks. As network structure is the only available information modality in a plain network, this category of anomaly detection methods try to exploit the network structure information to spot anomalies from different perspectives (Akoglu, Tong, and Koutra 2015; X. Xu et al. 2007). For instance, SCAN (X. Xu et al. 2007) is one of the first methods that target to find structural anomalies in networks. In recent days, attributed networks have been widely used to model a wide range of complex systems due to their superior capacity for handling data heterogeneity. In addition to the observed node-to-node interactions, attributed networks also encode a rich set of features for each node. Therefore, anomaly detection on attributed networks has drawn increasing research attention in the community, and various methods have been proposed (Müller et al. 2013; Sánchez et al. 2014). Among them, ConOut (Sánchez et al. 2014) identifies the local context for each node and performs anomaly ranking within the local context. More recently, researchers also propose to solve the problem of network anomaly detection using graph neural networks due to its strong modeling power. DOMINANT (Ding et al. 2019) achieves superior performance over other shallow methods by building a deep autoencoder architecture on top of the graph convolutional networks. Semi-GNN (D. Wang et al. 2019) is a semi-supervised graph neural model which adopts hierarchical attention to model the multi-view graph for fraud detection. GAS (A. Li et al. 2019) is a GCN-based large-scale anti-spam method for detecting spam advertisements. Zhao et al. propose a novel loss function to train GNNs for anomaly-detectable node representations (Zhao et al. 2020). Apart from the aforementioned methods, our approach focus on detecting anomalies on a target network with few labels by learning from multiple auxiliary networks.

2.2.3 Problem Definition

Our notations and definition for attributed network are explained in Section 2.1.3. Regarding few-shot cross-network anomaly detection, it aims to maximally improve the detection performance on the target network through transferring very limited supervised knowledge of ground-truth anomalies from the auxiliary network(s). In addition to the target network $\mathbf{G}^t$, in this work we assume there exist P auxiliary networks $\mathcal{G}^s = \{\mathbf{G}_1^s, \mathbf{G}_2^s, \ldots, \mathbf{G}_P^s\}$ sharing the same or similar domain with $\mathbf{G}^t$. For an attributed network, the set of labeled abnormal nodes is denoted as $\mathcal{V}^L$ and the set of unlabeled nodes is represented as $\mathcal{V}^U$. Note that $\mathcal{V} = \{\mathcal{V}^L, \mathcal{V}^U\}$ and in our problem $|\mathcal{V}^L| \ll |\mathcal{V}^U|$ since only few-shot labeled data is given. As network anomaly detection is commonly formulated as a ranking problem (Akoglu, Tong, and Koutra 2015), we formally define the few-shot cross-network anomaly detection problem as follows:

Problem Definition 2 ***Few-shot Cross-network Anomaly Detection***

Given: *P auxiliary networks, i.e., $\mathcal{G}^s = \{\mathbf{G}_1^s = (\mathbf{A}_1^s, \mathbf{X}_1^s), \mathbf{G}_2^s = (\mathbf{A}_2^s, \mathbf{X}_2^s), \ldots, \mathbf{G}_P^s = (\mathbf{A}_P^s, \mathbf{X}_P^s)\}$ and a target network $\mathbf{G}^t = (\mathbf{A}^t, \mathbf{X}^t)$, each of which contains a set of few-shot labeled anomalies (i.e., $\mathcal{V}_1^L, \mathcal{V}_2^L, \ldots, \mathcal{V}_P^L$ and $\mathcal{V}_t^L$).*

Goal: *to learn an anomaly detection model, which is capable of leveraging the knowledge of ground-truth anomalies from the multiple auxiliary networks, i.e., $\{\mathbf{G}_1^s, \mathbf{G}_2^s, \ldots, \mathbf{G}_P^s\}$, to detect abnormal nodes in the target network $\mathbf{G}^t$. Ideally, anomalies that are detected should have higher ranking scores than that of the normal nodes.*

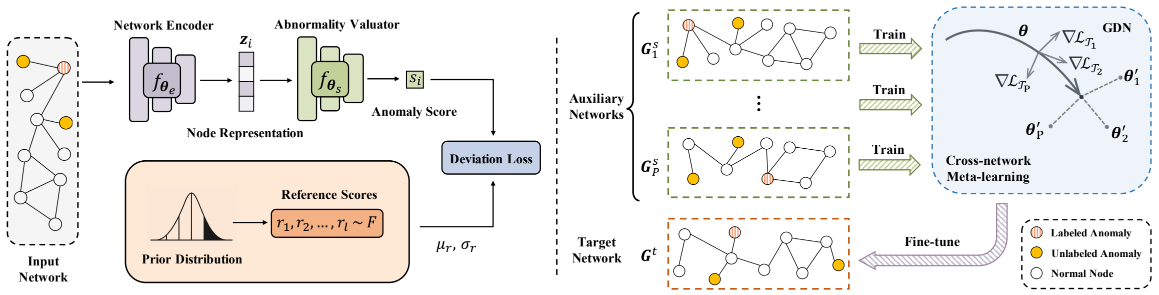


Figure 6. (Left) The model architecture of Graph Deviation Networks (GDN) for network anomaly detection with limited labeled data. (Right) The illustration of the overall framework Meta-GDN. Meta-GDN is trained across multiple auxiliary networks and can be well adapted to the target network with few-shot labeled data. Figure best viewed in color.

2.2.4 Meta-GDN

In this section, we introduce the details of the proposed framework – Meta-GDN for few-shot network anomaly detection. Specifically, Meta-GDN addresses the discussed challenges with the following two key contributions: (1) Graph Deviation Networks (GDN), a new family of graph neural networks that enable anomaly detection on an arbitrary individual network with limited labeled data; and (2) a cross-network meta-learning algorithm, which empowers GDN to transfer meta-knowledge across multiple auxiliary networks to enable few-shot anomaly detection on the target network. An overview of the proposed Meta-GDN is provided in Figure 6.

2.2.4.1 Graph Deviation Networks

To enable anomaly detection on an arbitrary network with few-shot labeled data, we first propose a new family of graph neural networks, called Graph Deviation Network (GDN). In essence, GDN is composed of three key building blocks, including (1) a *network encoder* for learning node representations; (2) an *abnormality valuator* for estimating the anomaly score for each node; and (3) a *deviation loss* for optimizing the model with few-shot labeled anomalies. We adopt the same structure as described in Section 2.1.4.2 for the *network encoder*, which is compatible with arbitrary GNN-based architecture (Thomas N. Kipf and Welling 2017b; Hamilton, Ying, and Leskovec 2017; Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018a; F. Wu et al. 2019). Here we employ Simple Graph Convolution (SGC) (F. Wu et al. 2019) in our implementation. The details for the other building blocks are as follows:

Abnormality Valuator. Afterwards, the learned node representations from the *network encoder* will be passed to the *abnormality valuator* $f_{\boldsymbol{\theta}_s}(\cdot)$ for further estimating the abnormality of each node. Specifically, the *abnormality valuator* is built with two feed-forward layers that transform the intermediate node representations to scalar anomaly scores:

$$\begin{aligned} \mathbf{o}_i &= \mathrm{ReLU}(\mathbf{W}_s \mathbf{z}_i + \mathbf{b}_s), \\ s_i &= \mathbf{u}_s^{\mathrm{T}} \mathbf{o}_i + b_s, \end{aligned} \tag{2.15}$$

where s_i is the anomaly score of node v_i and $\mathbf{o}_i$ is the intermediate output. $\mathbf{W}_s$ and $\mathbf{u}_s$ are the learnable weight matrix and weight vector, respectively. $\mathbf{b}_s$ and b_s are corresponding bias terms.

To be more concrete, the whole GDN model $f_{\boldsymbol{\theta}}(\cdot)$ can be formally represented as:

$$f_{\boldsymbol{\theta}}(\mathbf{A}, \mathbf{X}) = f_{\boldsymbol{\theta}_s}(f_{\boldsymbol{\theta}_e}(\mathbf{A}, \mathbf{X})), \tag{2.16}$$

which directly maps the input network to scalar anomaly scores, and can be trained in an end-to-end fashion.

Deviation Loss. In essence, the objective of GDN is to distinguish normal and abnormal nodes according to the computed anomaly scores with few-shot labels. Here we propose to adopt the deviation loss (Pang, Shen, and Hengel 2019) to enforce the model to assign large anomaly scores to those nodes whose characteristics significantly deviate from normal nodes. To guide the model learning, we first define a *reference score* (i.e., μ_r) as the mean value of the anomaly scores of a set of randomly selected normal nodes. It serves as the reference to quantify how much the scores of anomalies deviate from those of normal nodes.

According to previous studies (Pang, Shen, and Hengel 2019; Kriegel et al. 2011), Gaussian distribution is commonly a robust choice to fit the abnormality scores for a wide range of datasets. Based on this assumption, we first sample a set of k anomaly scores from the Gaussian prior distribution, i.e., $\mathcal{R} = \{r_1, r_2, \dots, r_k\} \sim \mathcal{N}(\mu, \sigma^2)$, each of which denotes the abnormality of a random normal node. The *reference score* is computed as the mean value of all the sampled scores:

$$\mu_r = \frac{1}{k} \sum_{i=1}^{k} r_i. \tag{2.17}$$

With the *reference score* μ_r, the deviation between the anomaly score of node v_i and the *reference score* can be defined in the form of standard score:

$$\mathrm{dev}(v_i) = \frac{s_i - \mu_r}{\sigma_r}, \tag{2.18}$$

where σ_r is the standard deviation of the set of sampled anomaly scores $\mathcal{R} = \{r_1, \dots, r_k\}$. Then the final objective function can be derived from the contrastive loss (Hadsell, Chopra, and LeCun 2006) by replacing the distance function with the deviation in Eq. (2.18):

$$\mathcal{L} = (1 - y_i) \cdot |\mathrm{dev}(v_i)| + y_i \cdot \max(0, m - \mathrm{dev}(v_i)), \tag{2.19}$$

where y_i is the ground-truth label of input node v_i. If node v_i is an abnormal node, $y_i = 1$, otherwise, $y_i = 0$. Note that m is a confidence margin which defines a radius around the deviation.

By minimizing the above loss function, GDN will push the anomaly scores of normal nodes as close as possible to μ_r while enforcing a large positive deviation of at least m between μ_r and the anomaly scores of abnormal nodes. This way GDN is able to learn a high-level abstraction of normal patterns with substantially less labeled anomalies, and empowers the node representation learning to discriminate normal nodes from the rare anomalies. Accordingly, a large anomaly score will be assigned to a node if its pattern significantly deviates from the learned abstraction of normal patterns.

Our preliminary results show that GDN is not sensitive to the choices of μ and σ as long as σ is not too large. Specifically, we set $\mu = 0$ and $\sigma = 1$ in our experiments, which helps GDN to achieve stable detection performance on different datasets. It is also worth mentioning that, as we cannot access the labels of normal nodes, we simply consider the unlabeled node in $\mathcal{V}^U$ as normal. Note that this way the remaining unlabeled anomalies and all the normal nodes will be treated as normal, thus contamination is introduced to the training set (i.e., the ratio of unlabeled anomalies to the total unlabeled training data $\mathcal{V}^U$). Remarkably, GDN performs very well by using this simple strategy and is robust to different contamination levels. The effect of different contamination levels to model performance is evaluated in Sec. 2.2.5.3.

2.2.4.2 Cross-network Meta-learning

Having the proposed Graph Deviation Networks (GDN), we are able to effectively detect anomalies on an arbitrary network with limited labeled data. When auxiliary networks from the same domain of the target network are available, how to transfer such valuable knowledge is the key to enable few-shot anomaly detection on the target network. Despite its feasibility, the performance would be rather limited if we directly borrow the idea of existing cross-network learning methods. The main reason is that those methods merely focus on transferring the knowledge from only a single network (M. Wu et al. 2020; Shen et al. 2020), which may cause negative transfer due to the divergent characteristics of anomalies on different networks. To this end, we turn to exploit multiple auxiliary networks to distill comprehensive knowledge of anomalies.

As an effective paradigm for extracting and transferring knowledge, meta-learning has recently received increasing research attention because of the broad applications in a variety of high-impact domains (Santoro et al. 2016; Vinyals et al. 2016; Ding, Wang, Li, Shu, et al. 2020; N. Wang et al. 2020; Liu, Zhou, Long, Jiang, Yao, et al. 2019; Liu et al. 2021). In essence, the goal of meta-learning is to train a model on a variety of learning tasks, such that the learned model is capable of effectively adapting to new tasks with very few or even one labeled data (Hochreiter, Younger, and Conwell 2001). In particular, Finn et al. (Finn, Abbeel, and Levine 2017) propose a model-agnostic meta-learning algorithm to explicitly learn the model parameters such that the model can achieve good generalization to a new task through a small number of gradient steps with limited labeled data. Inspired by this work, we propose to learn a meta-learner (i.e., Meta-GDN) as the initialization of GDN from multiple auxiliary networks, which possesses the generalization ability to effectively identify anomalous nodes on a new target network. Specifically, Meta-GDN extracts meta-knowledge of ground-truth anomalies from different few-shot network anomaly detection tasks on auxiliary networks during the training phase, and will be further fine-tuned for the new task on the target network, such that the model can make fast and effective adaptation.

We define each learning task as performing few-shot anomaly detection on an individual network, whose objective is to enforce large anomaly scores to be assigned to anomalies as defined in Eq. (2.19). Let $\mathcal{T}_i$ denote the few-shot network anomaly detection task constructed from network $\mathbf{G}_i^s$, then we have P learning tasks in each epoch. We consider a GDN model represented by a parameterized function $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$. Given P tasks, the optimization algorithm first adapts the initial model parameters $\boldsymbol{\theta}$ to $\boldsymbol{\theta}_i'$ for each learning task $\mathcal{T}_i$ independently. Specifically, the updated parameter $\boldsymbol{\theta}_i'$ is computed using $\mathcal{L}_{\mathcal{T}_i}$ on a batch of training data sampled from $\mathcal{V}_i^L$ and $\mathcal{V}_i^U$ in $\mathbf{G}_i^s$. Formally, the parameter update with one gradient step can be expressed as:

$$\boldsymbol{\theta}_i' = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}), \tag{2.20}$$

where α controls the meta-learning rate. Note that Eq. (2.20) only includes one-step gradient update, while it is straightforward to extend to multiple gradient updates (Finn, Abbeel, and Levine 2017).

The model parameters are trained by optimizing for the best performance of $f_{\boldsymbol{\theta}}$ with respect to $\boldsymbol{\theta}$ across all learning tasks. More concretely, the meta-objective function is defined as follows:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{P} \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}_i'}) = \min_{\boldsymbol{\theta}} \sum_{i=1}^{P} \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}})}). \tag{2.21}$$

By optimizing the objective of GDN, the updated model parameter can preserve the capability of detecting anomalies on each network. Since the meta-optimization is performed over parameters $\boldsymbol{\theta}$ with the objective computed using the updated parameters (i.e., $\boldsymbol{\theta}_i'$) for all tasks, correspondingly, the model parameters are optimized such that one or a small number of gradient steps on the target task (network) will produce great effectiveness.

Formally, we leverage stochastic gradient descent (SGD) to update the model parameters $\boldsymbol{\theta}$ across all tasks, such that the model parameters $\boldsymbol{\theta}$ are updated as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{P} \mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}'_i}), \tag{2.22}$$

where β is the meta step size. Specifically, for each batch, we randomly sample the same number of nodes from unlabeled data (i.e., $\mathcal{V}^U$) and labeled anomalies (i.e., $\mathcal{V}^L$) to represent normal and abnormal nodes.

2.2.5 Performance Evaluation

2.2.5.1 Evaluation Setting

Evaluation Datasets. In the experiment, we adopt three real-world datasets, which are publicly available and have been widely used in previous research (Rayana and Akoglu 2015; Sen et al. 2008a; Thomas N. Kipf and Welling 2017b; Hamilton, Ying, and Leskovec 2017). Table 4 summarizes the statistics of each dataset. The detailed description is as follows:

- **Yelp** (Rayana and Akoglu 2015) is collected from Yelp.com and contains reviews for restaurants in several states of the U.S., where the restaurants are organized by ZIP codes. The reviewers are classified into two classes, abnormal (reviewers with only filtered reviews) and normal (reviewers with no filtered reviews) according to the Yelp anti-fraud filtering algorithm. We select restaurants in the same location according to ZIP codes to construct each network, where nodes represent reviewers and there is a link between two reviewers if they have reviewed the same restaurant. We apply the bag-of-words model (Zhang, Jin, and Zhou 2010) on top of the textual contents to obtain the attributes of each node.
- **PubMed** (Sen et al. 2008a) is a citation network where nodes represent scientific

Table 4. Statistics of evaluation datasets. r_1 denotes the ratio of labeled anomalies to the total anomalies and r_2 is the ratio of labeled anomalies to the total number of nodes.

Datasets	Yelp	PubMed	Reddit
# nodes (avg.)	$4,872$	$3,675$	$15,860$
# edges (avg.)	$43,728$	$8,895$	$136,781$
# features	$10,000$	500	602
# anomalies (avg.)	223	201	796
r_1 (avg.)	4.48%	4.97%	1.26%
r_2 (avg.)	0.21%	0.27%	0.063%

articles related to diabetes and edges are citations relations. Node attribute is represented by a TF/IDF weighted word vector from a dictionary which consists of 500 unique words. We randomly partition the large network into non-overlapping sub-networks of similar size.

- **Reddit** (Hamilton, Ying, and Leskovec 2017) is collected from an online discussion forum where nodes represent threads and an edge exits between two threads if they are commented by the same user. The node attributes are constructed using averaged word embedding vectors of the threads. Similarly, we extract non-overlapping sub-networks from the original large network for our experiments.

Note that except the *Yelp* dataset, we are not able to access ground-truth anomalies for *PubMed* and *Reddit*. Thus we refer to two anomaly injection methods (Song et al. 2007; Ding, Li, and Liu 2019) to inject a combined set of anomalies (i.e., structural anomalies and contextual anomalies) by perturbing the topological structure and node attributes of the original network, respectively. To inject structural anomalies, we adopt the approach used by (Ding, Li, and Liu 2019) to generate a set of small cliques since small clique is a typical abnormal substructure in which a small set of nodes are much more closely linked to each other than average (Skillicorn 2007). Accordingly, we randomly select c nodes (i.e., clique size) in the network and then make these nodes fully linked to each other. By repeating this process K times (i.e., K cliques), we can obtain $K \times c$ structural anomalies. In our experiment, we set the clique size c to 15. In addition, we leverage the method introduced by (Song et al. 2007) to generate contextual anomalies. Specifically, we first randomly select a node i and then randomly sample another 50 nodes from the network. We choose the node j whose attributes have the largest Euclidean distance from node i among the 50 nodes. The attributes of node i (i.e., $\mathbf{x}_i$) will then be replaced with the attributes of node j (i.e., $\mathbf{x}_j$). Note that we inject structural and contextual anomalies with the same quantity and the total number of injected anomalies is around 5% of the network size.

Comparison Methods. We compare our proposed Meta-GDN framework and its base model GDN with two categories of anomaly detection methods, including (1) *feature-based* methods (i.e., LOF, Autoencoder and DeepSAD) where only the node attributes are considered, and (2) *network-based* methods (i.e., SCAN, ConOut, Radar, DOMINANT, and SemiGNN) where both topological information and node attributes are involved. Details of these compared baseline methods are as follows:

- **LOF** (Breunig et al. 2000) is a feature-based approach which detects outliers at the contextual level.
- **Autoencoder** (Zhou and Paffenroth 2017) is a feature-based unsupervised deep autoencoder model which introduces an anomaly regularizing penalty based upon L1 or L2 norms.
- **DeepSAD** (Ruff et al. 2020) is a state-of-the-art deep learning approach for general semi-supervised anomaly detection. In our experiment, we leverage the node attribute as the input feature.

Table 5. Performance comparison results (10-shot) w.r.t. AUC-ROC and AUC-PR on three datasets.

	Yelp		PubMed		Reddit	
Methods	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
LOF	0.375	0.042	0.575	0.187	0.518	0.071
Autoencoder	0.365	0.041	0.584	0.236	0.722	0.347
DeepSAD	0.460	0.062	0.528	0.1154	0.503	0.066
SCAN	0.397	0.046	0.421	0.048	0.29	0.048
ConOut	0.4025	0.041	0.511	0.093	0.551	0.085
Radar	0.415	0.045	0.573	0.244	0.721	0.281
DOMINANT	0.578	0.109	0.636	0.337	0.735	0.357
SemiGNN	0.497	0.0583	0.523	0.065	0.610	0.1343
GDN (ours)	0.678	0.132	0.736	0.438	0.811	0.379
Meta-GDN (ours)	**0.724**	**0.175**	**0.761**	**0.485**	**0.842**	**0.395**

- **SCAN** (X. Xu et al. 2007) is an efficient algorithm for detecting network anomalies based on a structural similarity measure.
- **ConOut** (Sánchez et al. 2014) identifies network anomalies according to the corresponding subgraph and the relevant subset of attributes in the local context.
- **Radar** (Li, Dani, Hu, and Liu 2017) is an unsupervised method that detects anomalies on attributed network by characterizing the residuals of attribute information and its coherence with network structure.
- **DOMINANT** (Ding et al. 2019) is a GCN-based autoencoder framework which computes anomaly scores using the reconstruction errors from both network structure and node attributes.
- **SemiGNN** (D. Wang et al. 2019) is a semi-supervised GNN model, which leverages the hierarchical attention mechanism to better correlate different neighbors and different views.

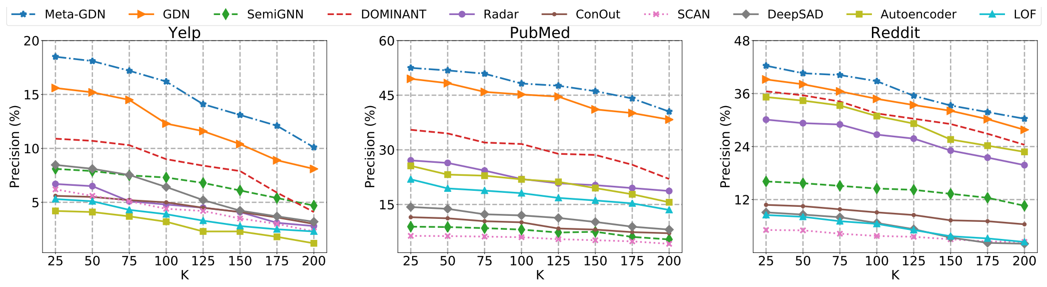


Figure 7. Performance comparison results (10-shot) w.r.t. Precision@K on three datasets. Figure best viewed in color.

Evaluation Metrics. In this work, we use the following metrics to have a comprehensive evaluation of the performance of different anomaly detection methods:

- **AUC-ROC** is widely used in previous anomaly detection research (Ding et al. 2019; Li, Dani, Hu, and Liu 2017). Area under curve (AUC) is interpreted as the probability that a randomly chosen anomaly receives a higher score than a randomly chosen normal object.
- **AUC-PR** is the area under the curve of precision against recall at different thresholds, and it only evaluates the performance on the positive class (i.e., abnormal objects). AUC-PR is computed as the average precision as defined in (Manning, Schütze, and Raghavan 2008) and is used as the evaluation metric in (Pang, Shen, and Hengel 2019).
- **Precision@K** is defined as the proportion of true anomalies in a ranked list of K objects. We obtain the ranking list in descending order according to the anomaly scores that are computed from a specific anomaly detection algorithm.

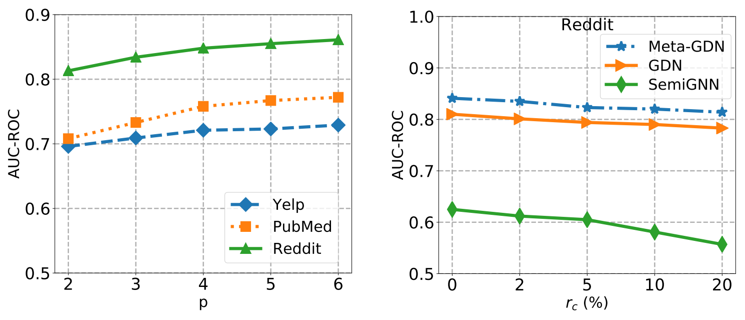
Table 6. Few-shot performance evaluation of Meta-GDN w.r.t. AUC-ROC and AUC-PR.

	Yelp		PubMed		Reddit	
Setting	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
1-shot	0.702	0.159	0.742	0.462	0.821	0.380
3-shot	0.709	0.164	0.748	0.468	0.828	0.386
5-shot	0.717	0.169	0.753	0.474	0.834	0.389
10-shot	0.724	0.175	0.761	0.485	0.842	0.395

2.2.5.2 Effectiveness Results

Overall Comparison. In the experiments, we evaluate the performance of the proposed framework Meta-GDN along with its base model GDN by comparing with the included baseline methods. We first present the evaluation results (10-shot) w.r.t. AUC-ROC and AUC-PR in Table 5 and the results w.r.t. Precision@K are visualized in Figure 7. Accordingly, we have the following observations, including: **(1)** in terms of AUC-ROC and AUC-PR, our approach Meta-GDN outperforms all the other compared methods by a significant margin. Meanwhile, the results w.r.t. Precision@K again demonstrate that Meta-GDN can better rank abnormal nodes on higher positions than other methods by estimating accurate anomaly scores; **(2)** unsupervised methods (e.g., DOMINANT, Radar) are not able to leverage supervised knowledge of labeled anomalies and therefore have limited performance. Semi-supervised methods (e.g., DeepSAD, SemiGNN) also fail to deliver satisfactory results. The possible explanation is that DeepSAD cannot model network information and SemiGNN requires a relatively large number of labeled data and multi-view data, which make them less effective in our evaluation; and **(3)** compared to the base model GDN, Meta-GDN is capable of extracting comprehensive meta-knowledge across multiple auxiliary networks by virtue of the cross-network meta-learning algorithm, which further enhances the detection performance on the target network.

Few-shot Evaluation. In order to verify the effectiveness of Meta-GDN in few-shot as well as one-shot network anomaly detection, we evaluate the performance of Meta-GDN with different numbers of labeled anomalies on the target network (i.e., 1-shot, 3-shot, 5-shot and 10-shot). Note that we respectively set the batch size b to 2, 4, 8, and 16 to ensure that there is no duplication of labeled anomalies exist in a sampled training batch. Also, we keep the number of labeled anomalies on auxiliary networks as 10. Table 6 summarizes the AUC-ROC/AUC-PR performance of Meta-GDN under different few-shot settings. By comparing the results in Table 5 and Table 6, we can see that even with only one labeled anomaly on the target network (i.e., 1-shot), Meta-GDN can still achieve good performance and significantly outperforms all the baseline methods. In the meantime, we can clearly observe that the performance of Meta-GDN increases with the growth of the number of labeled anomalies, which demonstrates that Meta-GDN can be better fine-tuned on the target network with more labeled examples.



(a) (b)

Figure 8. (a) Sensitivity analysis of Meta-GDN w.r.t. different number of auxiliary networks; (b) Model robustness study w.r.t. AUC-ROC with different contamination levels.

2.2.5.3 Sensitivity & Robustness Analysis

In this section, we further analyze the sensitivity and robustness of the proposed framework Meta-GDN. By providing different numbers of auxiliary networks during training, the model sensitivity results w.r.t. AUC-ROC are presented in Figure 8a. Specifically, we can clearly find that **(1)** as the number of auxiliary networks increases, Meta-GDN achieves constantly stronger performance on all the three datasets. It shows that more auxiliary networks can provide better meta-knowledge during the training process, which is consistent with our intuition; **(2)** Meta-GDN can still achieve relatively good performance when training with a small number of auxiliary networks (e.g., $p = 2$), which demonstrates the strong capability of its base model GDN. For example, on *Yelp* dataset, the performance barely drops 0.033 if we change the number of auxiliary networks from $p = 6$ to $p = 2$.

As discussed in Sec. 2.2.4.1, we treat all the sampled nodes from unlabeled data as normal for computing the deviation loss. This simple strategy introduces anomaly contamination in the unlabeled training data. Due to the fact that r_c is a small number in practice, our approach can work very well in a wide range of real-world datasets. To further investigate the robustness of Meta-GDN w.r.t. different contamination levels r_c (i.e., the proportion of anomalies in the unlabeled training data), we report the evaluation results of Meta-GDN, GDN and the semi-supervised baseline method SemiGNN in Figure 8b. As shown in the figure, though the performance of all the methods decreases with increasing contamination levels, both Meta-GDN and GDN are remarkably robust and can consistently outperform SemiGNN to a large extent.

2.3 Conclusion

Developing a powerful graph representation learning model for a downstream task often requires abundant annotated samples. However, in real-world attributed networks, a large portion of node classes only contains limited labeled instances, rendering a long-tail node class distribution. In practice, many current graph representation learning models usually fails to deal with new (i.e., never-seen-before) tasks since they cannot rapidly generalize from a few examples. In this chapter, I discuss my work on solving the challenges in few-shot node classification and few-shot network anomaly detection problems:

To handle the never-before-seen node classes, we propose a graph meta-learning framework – Graph Prototypical Networks (GPN). By constructing a pool of semi-supervised node classification tasks to mimic the real test environment, GPN is able to perform *meta-learning* on an attributed network and derive a highly generalizable model for handling the target classification task. Extensive experiments demonstrate the superior capability of GPN in few-shot node classification.

Meanwhile, we make the first investigation on the problem of few-shot cross-network anomaly detection. To tackle this problem, we first design a novel GNN architecture, GDN, which is capable of leveraging limited labeled anomalies to enforce statistically significant deviations between abnormal and normal nodes on an individual network. To further utilize the knowledge from auxiliary networks and enable few-shot anomaly detection on the target network, we propose a cross-network meta-learning approach, Meta-GDN, which is able to extract comprehensive meta-knowledge from multiple auxiliary networks in the same domain of the target network. Through extensive experimental evaluations, we demonstrate the superiority of Meta-GDN over the state-of-the-art methods.

Chapter 3

GRAPH WEAKLY-SUPERVISED LEARNING

Existing Graph ML algorithms are mainly developed for the supervised or semi-supervised setting where the input graph usually has relatively plenty of labeled instances. However, weakly-supervised Graph ML, in particular where only *incomplete* and *indirect* supervision signals are provided, remains largely understudied in the community. In this chapter, I will present my effort on addressing this challenging research problem, with a focus on the aforementioned two types of weak supervision.

Starting from the most common *incomplete* supervision, we proposed to enlarge receptive fields of GNNs and automatically exploit the knowledge from unlabeled data to deal with incomplete supervision. Specifically, I developed a graph self-training framework, namely *Meta Propagation Networks (Meta-PN)*, which is able to adjust its label propagation strategy and leverage large receptive fields for inferring accurate pseudo labels on unlabeled nodes. By augmenting the scarce training data with the generated pseudo labels, one can learn a label-efficient GNN with only few labels per class are provided.

Then, I discuss my work in leveraging the *indirect* supervision – the invaluable auxiliary information from a labeled attributed graph, to facilitate the anomaly detection in the unlabeled attributed graph is seldom investigated. To solve this problem, we propose a novel framework COMMANDER for cross-domain anomaly detection on attributed graphs. Specifically, COMMANDER first compresses the two attributed graphs from different domains to low-dimensional space via a graph attentive encoder. In addition, we utilize a domain discriminator and an anomaly classifier to detect anomalies that appear across networks from different domains. In order to further detect the anomalies that merely appear in the target network, we develop an attribute decoder to provide additional signals for assessing node abnormality.

3.1 Node Classification with Incomplete Supervision

3.1.1 Introduction

Graphs serve as a common language for modeling a plethora of structured and relational systems, ranging from social networks (Zafarani, Abbasi, and Liu 2014) to citation networks (Namata et al. 2012), to molecular graphs (Klicpera, Groß, and Günnemann 2019). To ingest the rich information encoded in graph-structured data, it is of paramount importance to learn expressive node representations by modeling the information from both node attributes and graph topology. Among numerous endeavors in the graph machine learning (Graph ML) community, graph neural networks (GNNs) have received significant attention due to their effectiveness and scalability (Thomas N. Kipf and Welling 2017b; Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018b; Hamilton, Ying, and Leskovec 2017).

In general, most of the prevailing GNNs adopt the message-passing scheme to learn the representation of a node by iteratively transforming, and propagating/aggregating node features from its local neighborhoods. Along with this idea, different designs of GNN architectures have been proposed, including graph convolutional networks (GCNs) (Thomas N. Kipf and Welling 2017b; Defferrard, Bresson, and Vandergheynst 2016), graph attention networks (GAT) (Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018b; X. Wang et al. 2019) and many others (Hamilton, Ying, and Leskovec 2017;

K. Xu et al. 2019; Klicpera, Bojchevski, and Günnemann 2019; F. Wu et al. 2019; T. Chen et al. 2020). Despite their promising results, existing GNNs developed for *semi-supervised* node classification predominantly assume that the provided gold-labeled nodes are relatively abundant. This assumption is often impractical as data labeling requires intensive domain knowledge, especially when considering the heterogeneity of graph-structured data (Yao et al. 2020a; Ding, Wang, Li, Shu, et al. 2020). When only few labeled nodes per class are available, how to improve the expressive power of Graph ML models for tackling the *few-shot semi-supervised* node classification problem remains understudied and meanwhile requires urgent research efforts.

However, it is a non-trivial and challenging task mainly because of two reasons: (i) ***oversmoothing and overfitting.*** In general, most of the existing GNNs are designed with shallow architecture with restricted receptive fields, thereby restricting the efficient propagation of label information (Li, Han, and Wu 2018). In order to propagate the label signals more broadly, larger receptive fields of GNNs, i.e., the number of layers, are particularly desirable (Klicpera, Bojchevski, and Günnemann 2019). Due to the entanglement of representation transformation and propagation in each layer, GNNs will face the oversmoothing issue when increasing the model depth (Liu, Gao, and Ji 2020), which in turn renders the learned node representations inseparable. In the meantime, when training with few labeled nodes, an over-parametric deep GNN model tends to overfit and goes timber easily; (ii) ***no auxiliary knowledge.*** Though previous works proposed for graph few-shot learning (Ding, Wang, Li, Shu, et al. 2020) or cross-network transfer learning (Yao et al. 2020a) also focus on related low-resource scenarios, their key enabler lies in transferring knowledge from either label-rich node classes or other similar networks. Nonetheless, such auxiliary knowledge is commonly not accessible, making those methods practically infeasible to be applied to few-shot semi-supervised learning. As suggested by previous research, pseudo-labeling (Li, Han, and Wu 2018; Sun, Lin, and Zhu 2020; Ding, Xu, et al. 2022) is commonly beneficial to solve semi-supervised learning, whereas inaccurate pseudo labels may instead lead to abysmal failure. Hence, how to infer accurate pseudo labels on unlabeled nodes plays a pivotal role to solve the studied research problem.

To address the aforementioned challenges, we propose a new graph meta-learning framework, Meta Propagation Networks (Meta-PN), which goes beyond the canonical message-passing scheme of GNNs and learns expressive node representations in a more label-efficient way. Specifically, Meta-PN is built with two simple neural networks, i.e., *adaptive label propagator* and *feature-label transformer*, which inherently decouples the entangled propagation and transformation steps of GNNs, thereby allowing sufficient propagation of label signals without suffering the oversmoothing issue. At its core, the *adaptive label propagator* is meta-learned to adjust its propagation strategy for inferring accurate pseudo labels on unlabeled nodes, according to the feedback (i.e., the performance change on the gold-labeled nodes) from the target model *feature-label transformer*. This way the generated soft pseudo labels not only capture informative local and global structure information, but more importantly, have aligned data usage with the gold-labeled nodes. Optimizing with our proposed meta-learning algorithm, those two decoupled networks are able to reinforce each other synergistically. As a result, the target model assimilates the encoded knowledge of pseudo-labeled nodes and offers excellent performance for the semi-supervised node classification problem even if only few labeled nodes are available. In summary, the contributions of our work are as follows:

- We study the problem of semi-supervised node classification under the few-shot setting, which remains largely under-studied in the Graph ML community.

- We propose a simple yet effective graph meta-learning framework Meta-PN to solve the studied problem. The essential idea is to augment the limited labeled data via a meta-learned label propagation strategy.
- We conduct comprehensive evaluations on different graph benchmark datasets to corroborate the effectiveness of Meta-PN. The results show its superiority over the state-of-the-arts on semi-supervised node classification, especially under the low-resource setting.

3.1.2 Related Works

For real-world graph learning tasks, the amount of gold-labeled samples is usually quite limited due to the expensive labeling cost. To improve the GNN model performance on the node classes with only few labeled nodes, graph few-shot learning (F. Zhou et al. 2019; Ding, Wang, Li, Shu, et al. 2020; N. Wang et al. 2020) and cross-network transfer learning (Yao et al. 2020a; Ding, Zhou, et al. 2021) have been proposed to transfer the knowledge from other auxiliary data source(s). Nonetheless, for the problem of *few-shot semi-supervised* node classification, such auxiliary datasets are commonly not allowed to use. As another line of related work, Li et al. (Li, Han, and Wu 2018) combined GCNs and self-training to expand supervision signals, while M3S (Sun, Lin, and Zhu 2020) advances this idea by utilizing the clustering method to eliminate the inaccurate pseudo labels. However, those methods cannot directly address the oversmoothing issue and may suffer from inaccurate pseudo labels. By conducting meta-learning on top of a decoupled design, our approach Meta-PN achieves superior performance on few-shot semi-supervised node classification.

3.1.3 Architecture Overview

For solving the problem of few-shot semi-supervised node classification, we propose a new framework Meta Label Propagation (Meta-PN), which is built with two simple neural networks, i.e., *adaptive label propagator* and *feature-label transformer*. By decoupling the propagation and transformation steps with two independent networks, such a design inherently allows large receptive fields without suffering performance deterioration. Upon our proposed meta-learning algorithm, the meta learner – *adaptive label propagator* learns to adjust its propagation strategy for inferring accurate pseudo labels on unlabeled nodes, by using the feedback from the target model. Meanwhile, the target model – *feature-label transformer* assimilates both the structure and feature knowledge from pseudo-labeled nodes, therefore addressing the challenges behind few-shot semi-supervised learning. Specifically, we introduce the architecture details as follows:

3.1.3.1 Adaptive Label Propagator (Meta Learner)

In order to enable broader propagation of label signals, we propose to adopt the idea of label propagation (LP) (Zhu and Ghahramani 2002) to encode informative local and global structural information. Similar to the message-passing scheme adopted by many GNNs, label propagation follows the principle of Homophily (McPherson, Smith-Lovin, and Cook 2001) that indicates two connected nodes tend to be similar (share same labels). Specifically, the objective of LP is to find a prediction matrix $\hat{\mathbf{Y}} \in \mathbb{R}^{n \times c}$ that agrees with the label matrix $\mathbf{Y}$ while being smooth on the graph such that nearby vertices have similar soft labels (Dengyong Zhou et al. 2004). Generally, the solution can be approximated via the iteration as follows:

$$\hat{\mathbf{Y}} = \mathbf{Y}^{(K)}, \mathbf{Y}^{(k+1)} = \mathbf{T}\mathbf{Y}^{(k)}, \tag{3.1}$$

where $\mathbf{Y}^{(0)} = \mathbf{Y}$ and K denotes the number of power iteration (propagation) steps. The transition matrix is denoted by $\mathbf{T}$, which can be set as any form of normalized adjacency matrix (e.g., $\tilde{\mathbf{A}}_{sym}$). After K iterations of label propagation, the predicted soft label matrix $\hat{\mathbf{Y}}$ can capture the prior knowledge of neighborhood label distribution up to K hops away.

In practice, various propagation schemes can be adopted for LP, such as the Personalized PageRank (Klicpera, Bojchevski, and Günnemann 2019) where $\mathbf{Y}^{(k+1)} = (1-\alpha)\mathbf{T}\mathbf{Y}^{(k)} + \alpha\mathbf{Y}^{(0)}$. With appropriate teleport probability α, the smoothed labels can avoid losing the focus on local neighborhood even using infinitely many propagation steps (Klicpera, Bojchevski, and Günnemann 2019). However, most of the existing LP algorithms cannot adaptively balance the label information from different neighborhoods for each node, which largely restricts the model expressive power when learning with complex real-world graphs.

To counter this issue, we build an *adaptive label propagator* $g_{\boldsymbol{\phi}}(\cdot)$ parameterized with $\boldsymbol{\phi}$, which is able to adjust the contribution of different propagation steps for computing the smoothed label vector of one node. Specifically, the propagation strategy can be formulated as:

$$\hat{\mathbf{Y}}_{i,:} = \sum_{k=0}^{K} \gamma_{ik}\mathbf{Y}_{i,:}^{(k)}, \mathbf{Y}^{(k+1)} = \mathbf{T}\mathbf{Y}^{(k)}, \tag{3.2}$$

where γ_{ik} denotes the influence from k-hop neighborhood for node v_i and can be computed by the attention mechanism:

$$\gamma_{ik} = \frac{\exp\left(\mathbf{a}^{\mathrm{T}}\mathrm{ReLU}\left(\mathbf{W}\mathbf{Y}_{i,:}^{(k)}\right)\right)}{\sum_{k'=0}^{K}\exp\left(\mathbf{a}^{\mathrm{T}}\mathrm{ReLU}\left(\mathbf{W}\mathbf{Y}_{i,:}^{(k')}\right)\right)}, \tag{3.3}$$

where $\mathbf{a} \in \mathbb{R}^{c}$ is the attention vector and $\mathbf{W} \in \mathbb{R}^{c \times c}$ is a weight matrix. By setting the attention vector and weight matrix as learnable parameters, the *adaptive label propagator* acquire the capability of adjusting its propagation strategy for each node and the final smoothed labels can capture rich structure information of the input graph.

3.1.3.2 Feature-label Transformer (Target Model)

After encoding the structure knowledge into the smoothed label matrix $\hat{\mathbf{Y}}$, we then build a *feature-label transformer* $f_{\boldsymbol{\theta}}(\cdot)$ that transforms node features to node label, in order to further capture feature-based graph information. For each node v_i, the *feature-label transformer* parameterized with $\boldsymbol{\theta}$ takes the node feature vector $\mathbf{X}_{i,:}$ as input and predicts its node label $\mathbf{P}_{i,:}$ by:

$$\mathbf{P}_{i,:} = f_{\boldsymbol{\theta}}(\mathbf{X}_{i,:}), \tag{3.4}$$

where $f_{\boldsymbol{\theta}}(\cdot)$ is a multi-layer perceptron (MLP) followed by a softmax function.

In order to learn the target model , i.e., *feature-label transformer*, we take the soft pseudo labels computed by the *adaptive label propagator* as “ground-truth”. Ideally, if the generated pseudo labels are of high quality, they can be used to augment the insufficient labeled nodes to avoid overfitting and improve the model generalization ability (Li, Han, and Wu 2018). In the meantime, high-quality pseudo-labeled data not only encodes the feature patterns of unlabeled nodes, but also carries informative local and global structure knowledge, which enables the target model to leverage larger receptive fields without suffering from performance degradation. As a result, the *feature-label transformer* can achieve excellent performance on the problem of few-shot semi-supervised node classification.

3.1.3.3 Learning to Propagate

One key challenge of our approach lies in how to learn a better label propagation strategy for generating pseudo labels on unlabeled nodes. If the pseudo labels are inaccurate, the target model may easily overfit to mislabeled nodes and encounter severe performance degradation (Ren, Zeng, et al. 2018). This issue is also known as the problem of confirmation bias in pseudo-labeling (Arazo et al. 2020). While inferring accurate pseudo labels by recursively selecting a subset of samples, re-training the prediction model will be too expensive and unstable. Hence, without linking the two networks in a principled way, it is almost infeasible to enforce the *adaptive label propagator* to efficiently infer meaningful label propagation strategy for improving the performance of the *feature-label transformer*.

In this work, we propose to tackle this problem through a unified meta-learning algorithm, allowing the model to infer accurate pseudo labels for unlabeled nodes and learn a better target model. In a sense, if the generated pseudo labels are of high quality, their data utility should align with the gold-labeled nodes. Accordingly, we can derive the following meta-learning objective: *optimal pseudo labels generated by meta-learner should maximize target model's performance (minimize the classification loss) on the gold-labeled training nodes.* For each *meta label propagation* task, the goal is to generate pseudo labels for a batch of unlabeled nodes using the feedback of the target model (i.e., *feature-label transformer*). By optimizing the *adaptive label propagator* on a meta-level, it can adjust the label propagation strategy to generate informative pseudo-labeled data.

3.1.3.4 Model Learning via Bi-level Optimization

The above meta-learning objective implies a bi-level optimization problem with $\boldsymbol{\phi}$ as the outer-loop parameters and $\boldsymbol{\theta}$ as the inner-loop parameters. This problem shares the same formulation with many meta-learning algorithms that have been proposed for solving different learning tasks such as few-shot learning (Finn, Abbeel, and Levine 2017), hyper-parameter optimization (Baydin et al. 2018), and neural architecture search (Liu, Simonyan, and Yang 2018). Specifically, let $\mathcal{L}$ denote the cross-entropy loss for node classification, and this bi-level optimization problem can be formulated as:

$$\begin{aligned} &\text{Outer loop: } \boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{v_i \in \mathcal{V}^L}[\mathcal{L}(f_{\boldsymbol{\theta}^*(\boldsymbol{\phi})}(\mathbf{X}_{i,:}), \mathbf{Y}_{i,:})], \\ &\text{Inner loop: } \boldsymbol{\theta}^*(\boldsymbol{\phi}) = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{v_i \in \mathcal{V}^U}[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{X}_{i,:}), g_{\boldsymbol{\phi}}(\mathbf{Y}, \mathbf{A})_{i,:})]. \end{aligned} \tag{3.5}$$

The optimal solution of this bi-level optimization problem can potentially train a highly discriminative *feature-label transformer* with abundant pseudo-labeled data and only a small set of gold-labeled data. However, deriving exact solutions for this bi-level problem is indeed analytically intractable and computationally expensive, owing to the fact that it requires solving for the optimal $\boldsymbol{\theta}^*(\boldsymbol{\phi})$ whenever $\boldsymbol{\phi}$ gets updated. To approximate the optimal solution $\boldsymbol{\theta}^*(\boldsymbol{\phi})$, we propose to take one step of gradient descent update for $\boldsymbol{\theta}$, without solving the inner-loop optimization completely by training until convergence. This way allows the optimization algorithm to alternatively update the parameters of *feature-label transformer* in the inner loop and the parameters of *adaptive label propagator* in the outer loop:

Target Model (Inner-loop) Update. Given a batch of unlabeled nodes from $\mathcal{V}^U$, we update the target model parameters $\boldsymbol{\theta}$ by taking their pseudo labels computed by the *adaptive label propagator* as ground-truth. For simplicity, we use $J_{\text{pseudo}}(\boldsymbol{\theta}, \boldsymbol{\phi})$ to denote the inner-loop loss computed on a batch of pseudo-labeled nodes. Assuming that parameter $\boldsymbol{\theta}$ is updated using the computed gradient descent on $J_{\text{pseudo}}(\boldsymbol{\theta}, \boldsymbol{\phi})$, with a learning rate $\eta_{\boldsymbol{\theta}}$, then we have:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} J_{\text{pseudo}}(\boldsymbol{\theta}, \boldsymbol{\phi}). \tag{3.6}$$

Meta Learner (Outer-loop) Update. Note that the dependency between $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ allows us to compute the meta-level (outer-loop) loss using the gold-labeled nodes from $\mathcal{V}^L$. We denote this loss by $J_{\text{gold}}(\boldsymbol{\theta}'(\boldsymbol{\phi}))$ for the purpose of simplicity, and back-propagate this loss to compute the gradient for the *feature-label transformer*. Having the gradient, we can update on the backward parameters $\boldsymbol{\phi}$ with learning rate $\eta_{\boldsymbol{\phi}}$:

$$\boldsymbol{\phi}' = \boldsymbol{\phi} - \eta_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\phi}} J_{\text{gold}}(\boldsymbol{\theta}'(\boldsymbol{\phi})). \tag{3.7}$$

To further compute the gradient of $\boldsymbol{\phi}$, we apply chain rule to differentiate $J_{\text{gold}}(\boldsymbol{\theta}'(\boldsymbol{\phi}))$ with respect to $\boldsymbol{\phi}$ via $\boldsymbol{\theta}'$, where $\boldsymbol{\theta}'(\boldsymbol{\phi}) = \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} J_{\text{pseudo}}(\boldsymbol{\theta}, \boldsymbol{\phi})$. The full derivation is delegated to the Appendix **??**. Here, we directly present the final result:

$$\nabla_{\boldsymbol{\phi}} J_{\text{gold}}(\boldsymbol{\theta}'(\boldsymbol{\phi})) \approx -\frac{\eta_{\boldsymbol{\phi}}}{2\epsilon}[\nabla_{\boldsymbol{\phi}} J_{\text{pseudo}}(\boldsymbol{\theta}^{+}, \boldsymbol{\phi}) - \nabla_{\boldsymbol{\phi}} J_{\text{pseudo}}(\boldsymbol{\theta}^{-}, \boldsymbol{\phi})], \tag{3.8}$$

where $\boldsymbol{\theta}^{\pm} = \boldsymbol{\theta} \pm \epsilon \nabla_{\boldsymbol{\theta}'} J_{\text{gold}}(\boldsymbol{\theta}'(\boldsymbol{\phi}))$, and ϵ is a small scalar for finite difference approximation.

By alternating the update rules in Eq. (3.6) and Eq. (3.7), we are able to progressively learn the two modules. Finally, as the *feature-label transformer* only learns from unlabeled data with pseudo labels generated by the *adaptive label propagator*, we can further fine-tune the *feature-label transformer* on labeled data to improve its accuracy. After the model converges, we use the *feature-label transformer* to make final predictions on unlabeled nodes.

3.1.4 Performance Evaluation

3.1.4.1 Evaluation Settings

Evaluation Datasets. We conduct experiments on five graph benchmark datasets for semi-supervised node classification to demonstrate the effectiveness of the proposed Meta-PN. The detailed statistics of the datasets are summarized in Table 7. Specifically, **Cora-ML**, **CiteSeer** (Sen et al. 2008a) and **PubMed** (Namata et al. 2012) are the three most widely used citation networks. **MS-CS** is a co-authorship network based on the Microsoft Academic Graph (Shchur et al. 2018). For data splitting, we follow the previous work (Klicpera, Bojchevski, and Günnemann 2019) and split each dataset into training set (i.e., K nodes per class for K-shot task), validation set and test set. In addition, to further evaluate the performance of different methods on large-scale graphs, we further include the **ogbn-arxiv** datasets from Open Graph Benchmark (OGB) (W. Hu et al. 2020). For the ogbn-arxiv dataset, we randomly sample 1.0%, 1.5%, 2.0%, 2.5% nodes from its training splits as labeled data while using the same validation and test splits in OGB Benchmark (W. Hu et al. 2020). Note that for all the datasets, we run each experiment 100 times with multiple random splits and different initializations.

Table 7. Summary Statistics of the Evaluation Datasets.

Dataset	# Nodes	# Edges	# Features	# Classes
Cora-ML	2,810	7,981	2,879	7
CiteSeer	2,110	3,668	3,703	6
PubMed	19,717	44,324	500	3
MS-CS	18,333	81,894	6,805	15
ogbn-arxiv	169,343	1,166,243	15	40

Compared Methods. To corroborate the effectiveness of our approach, three categories of baselines are included in our experiments: (i) *Classical Models*. **MLP**, **LP** (Label Propagation) (Dengyong Zhou et al. 2004) are two classical models using only feature and structure information, respectively. **GCN** (Thomas N. Kipf and Welling 2017b) and **SGC** (F. Wu et al. 2019) are two representative GNN models. Due to the space limit, we omit some baselines like GAT, GraphSAGE since similar results can be observed; (ii) *Label-efficient GNNs*. **GLP** (Generalized Label Propagation) and **IGCN** (Improved GCN) (Q. Li et al. 2019) are two models combine label propagation and GCN from a unifying graph filtering perspective. **M3S** (Sun, Lin, and Zhu 2020) is a multi-stage self-training framework, which incorporates self-supervised learning to improve the model performance with few labeled nodes; (iii) *Deep GNNs*. **APPNP** (Klicpera, Bojchevski, and Günnemann 2019) decouples prediction and propagation with performing personalized propagation of neural predictions, while **DAGNN** (Liu, Gao, and Ji 2020) adaptively incorporate information from large receptive fields. **C&S** (Q. Huang et al. 2021) is an effective model that combines label propagation and simple neural networks. **GPR-GNN** (Chien et al. 2021) addresses the limitation of APPNP on different types of graphs with adaptive propagation weights.

3.1.4.2 Performance Evaluation

Performance EvaluationFew-shot Semi-supervised Evaluation. First, we evaluate the proposed approach Meta-PN and all the baseline methods on few-shot semi-supervised node classification, which aims to predict the missing node labels with only a few labeled nodes. The average test accuracies under the few-shot setting (i.e., 3-shot and 5-shot) can be found in Table 8 and Table 9. From the reported results, we can clearly see that Meta-PN significantly outperforms all the baseline methods on each dataset based on paired t-tests with $p < 0.05$. Specifically, we elaborate our in-depth observations and analysis as follows: (i) without abundant labeled data, classical models including vanilla GNNs only obtain very poor classification accuracy under different evaluation entries; (ii) overall the label-efficient GNNs outperform classical GNNs, but still cannot achieve satisfying results. One major reason is that those methods cannot handle the oversmoothing issue since they are incapable of explicitly leveraging the knowledge from large receptive fields; (iii) by enabling better propagation of label signals, deep GNNs have stronger performance than both the classical models and label-efficient GNNs, which again demonstrates the necessity of addressing the oversmoothing issue for solving the few-shot semi-supervised learning problem. However, existing deep GNNs are not specifically developed to tackle the data sparsity issue, thus their performance still falls behind Meta-PN by a noticeable margin on different datasets when only very few labels are available. This observation proves that Meta-PN is able to address the overfitting and oversmoothing issues when labeled data is extremely sparse by combining the power of large receptive fields and pseudo labels.

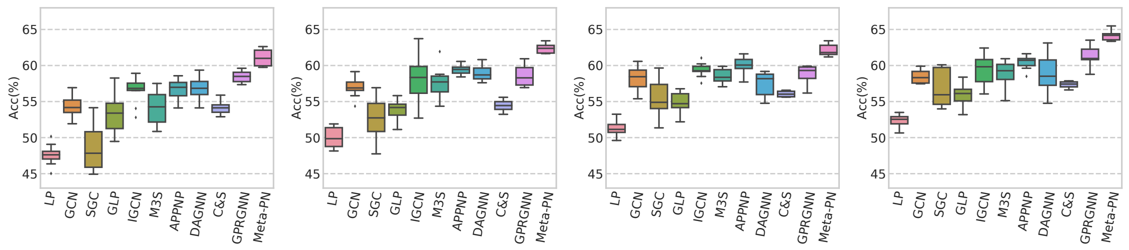
Evaluation on Open Graph Benchmark (OGB). Real-world graphs commonly have a larger size and more node classes than many toy graphs, leading to the collected graphs having noisy structures and complex properties. To further illustrate the effectiveness of our approach on large-scale real-world graphs, we adopt the widely used ogbn-arxiv dataset and compare all the methods under the few-shot setting (i.e., from 1% to 2.5% label ratio). We summarize their performance for few-shot

Table 8. Test accuracy on few-shot semi-supervised node classification on Cora-ML and CiteSeer: mean accuracy (%) ± 95% confidence interval.

Method	Cora-ML		CiteSeer	
	3-shot	5-shot	3-shot	5-shot
MLP	41.07 ± 0.76	51.12 ± 0.61	43.34 ± 0.56	44.90 ± 0.60
LP	62.07 ± 0.71	68.01 ± 0.62	54.07 ± 0.59	55.73 ± 1.19
GCN	48.02 ± 0.89	67.32 ± 1.02	53.60 ± 0.86	62.60 ± 0.58
SGC	49.60 ± 0.55	67.24 ± 0.86	57.37 ± 0.98	61.55 ± 0.53
GLP	65.57 ± 0.26	71.26 ± 0.31	65.76 ± 0.49	71.36 ± 0.18
IGCN	66.60 ± 0.29	72.50 ± 0.20	67.47 ± 0.29	$\underline{72.92 \pm 0.10}$
M3S	64.66 ± 0.31	69.64 ± 0.18	65.12 ± 0.20	68.18 ± 0.18
APPNP	$\underline{72.39 \pm 0.98}$	$\underline{78.32 \pm 0.58}$	$\underline{67.55 \pm 0.77}$	71.08 ± 0.61
DAGNN	71.86 ± 0.75	77.20 ± 0.69	66.62 ± 0.27	70.55 ± 0.12
C&S	68.93 ± 0.68	73.37 ± 0.24	63.02 ± 0.72	64.72 ± 0.53
GPR-GNN	70.98 ± 0.84	75.18 ± 0.52	64.32 ± 0.81	65.28 ± 0.52
Meta-PN	$\mathbf{74.94 \pm 0.25}$	$\mathbf{79.88 \pm 0.15}$	$\mathbf{70.48 \pm 0.34}$	$\mathbf{74.14 \pm 0.50}$

Table 9. Test accuracy on few-shot semi-supervised node classification on PubMed and MS-CS: mean accuracy (%) ± 95% confidence interval.

Method	PubMed		MS-CS	
	3-shot	5-shot	3-shot	5-shot
MLP	56.59 ± 0.93	59.90 ± 0.84	70.33 ± 0.37	79.41 ± 0.31
LP	58.75 ± 0.89	59.91 ± 0.85	57.96 ± 0.69	62.98 ± 0.61
GCN	58.89 ± 0.80	65.77 ± 0.98	69.24 ± 0.94	84.43 ± 0.89
SGC	63.37 ± 0.93	64.93 ± 0.81	72.11 ± 0.76	87.51 ± 0.27
GLP	65.34 ± 0.54	65.26 ± 0.29	86.10 ± 0.21	86.94 ± 0.23
IGCN	62.28 ± 0.23	65.19 ± 0.13	85.83 ± 0.06	87.01 ± 0.05
M3S	63.40 ± 0.32	68.85 ± 0.26	84.96 ± 0.18	86.83 ± 0.29
APPNP	70.52 ± 0.62	$\underline{74.24 \pm 0.87}$	$\underline{86.65 \pm 0.42}$	90.13 ± 0.86
DAGNN	$\underline{71.22 \pm 0.82}$	73.91 ± 0.71	86.32 ± 0.57	$\underline{90.30 \pm 0.66}$
C&S	70.51 ± 0.57	73.22 ± 0.57	85.86 ± 0.45	87.99 ± 0.24
GPR-GNN	71.03 ± 0.73	74.08 ± 0.65	86.12 ± 0.37	90.29 ± 0.38
Meta-PN	$\mathbf{73.25 \pm 0.77}$	$\mathbf{77.78 \pm 0.92}$	$\mathbf{88.99 \pm 0.29}$	$\mathbf{91.31 \pm 0.22}$



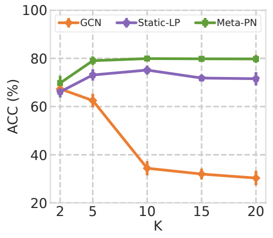
(a) label ratio = 1.0% (b) label ratio = 1.5% (c) label ratio = 2.0% (d) label ratio = 2.5%

Figure 9. Comparison results on ogbn-arxiv w.r.t different size of training labels.

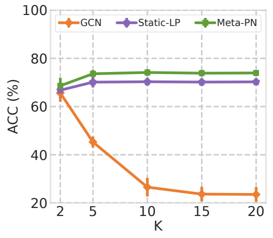
semi-supervised node classification on ogbn-arxiv in Figure 9 by changing the ratio of training labels, in which we omit MLP as its test accuracy is much lower than the other methods. We can observe that Meta-PN can significantly outperform all the baseline models under different few-shot environments. Compared to the other baseline methods, the performance of Meta-PN is relatively stable when we decrease the ratio of training labels, which demonstrates the robustness of Meta-PN in handling noisy and complex real-world graphs. Remarkably, our approach can achieve close performance to the vanilla GCN on ogbn-arxiv with much fewer labeled nodes (2.5% vs. 54%).

3.1.4.3 Parameter & Ablation Analysis.

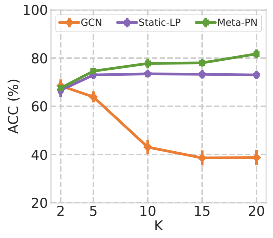
To demonstrate the effects of using different propagation steps and the importance of the meta-leaned label propagation strategy for Meta-PN, we compare our approach with two baselines under the 5-shot (or 1.0% label ratio for ogbn-arxiv) semi-supervised setting with varying number of propagation steps. Specifically, *GCN* learns the node representation with the standard message-passing scheme while *Static-LP* representing the variant of Meta-PN that uses fixed teleport probabilities instead of meta-learned ones. The evaluation results are shown in Figure 10. As we can observe from the



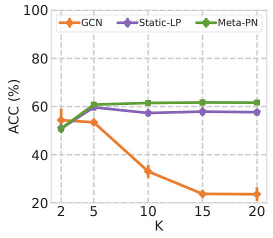
(a) Cora-ML



(b) CiteSeer



(c) PubMed



(d) Ogbn-arxiv

Figure 10. Few-shot (i.e., 5-shot or 1.0% label ratio) evaluation on different datasets w.r.t. propagation steps (K).

figure, *GCN* can achieve very close performance with the other two methods when the number of propagation steps is relatively small. While if we largely increase the number of propagation steps, the performance of *GCN* breaks down due to the oversmoothing issue. Empowered by the idea of label propagation, *Static-LP* can largely alleviate the oversmoothing issue and significantly outperform *GCN*. This verifies that larger propagation steps or receptive fields are necessary for improving the performance of GNN when labeled data is extremely limited. In the meantime, *Static-LP* still falls behind Meta-PN, mainly because of the infeasibility of balancing the importance of different receptive fields. On the contrary, Meta-PN is able to address this issue by inferring optimal pseudo labels on unlabeled nodes with our meta-learning algorithm. Its performance becomes stable when $K \geq 10$, indicating that Meta-PN can obtain good performance considering both efficiency and effectiveness with a moderate number of propagation steps (e.g., $K = 10$).

3.2 Cross-domain Graph Anomaly Detection

3.2.1 Introduction

Attributed graphs are a type of graphs that not only model the attributes of each data instance, but also encode the inherent dependencies among them. They have been widely used to model complex systems such as social media networks (Qi et al. 2011), academic graphs (J. Tang et al. 2008), financial transaction networks (Akoglu, Tong, and Koutra 2015). However, anomalous nodes – whose patterns significantly deviate from the majority – can be rampant in attributed graphs and cause real-world societal effects. For example, spammers in social networks can coordinate among themselves to launch various attacks such as spreading ads to generate sales, disseminating pornography, viruses, phishing, etc (X. Hu et al. 2013); fraud behaviors in financial networks may lead to huge financial loss for both customers and merchants (West and Bhattacharya 2016). Therefore, it is critical to detect anomalies on attributed graphs.

For a real-world anomaly detection system, it is often unrealistic to obtain abundant labeled data for every domain (e.g., *Hotels* and *Restaurants* are two different domains in Yelp) due to the expensive labeling cost (Qu et al. 2019; Pang et al. 2020). As such, graph anomaly detection is commonly performed in the single-domain setting, and unsupervised methods are proposed to handle those unlabeled domains (Akoglu, Tong, and Koutra 2015). However, the performances of unsupervised approaches may be limited without any supervision information. Thus when the target graph is from an unlabeled domain, it is natural and important to explore the auxiliary knowledge from other related domains that come from the same data platform. Specifically, we would like to investigate whether the anomaly detection performance on an

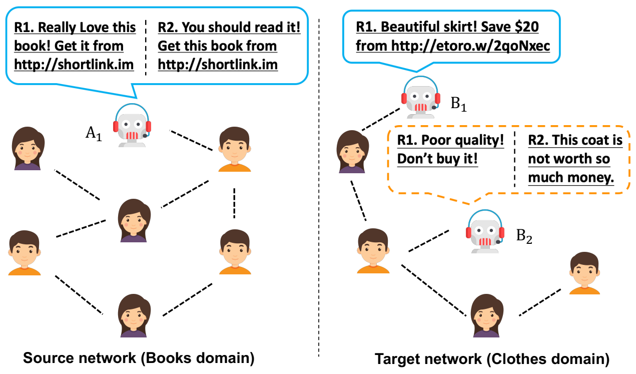

R1. Really Love this book! Get it from http://shortlink.im
R2. You should read it! Get this book from http://shortlink.im
A$_1$
Source network (Books domain)
R1. Beautiful skirt! Save $20 from http://etoro.w/2qoNxec
B$_1$
R1. Poor quality! Don't buy it!
R2. This coat is not worth so much money.
B$_2$
Target network (Clothes domain)

Figure 11. An example of cross-domain graph anomaly detection. A_1 and B_1 can be considered as the *shared anomalies* since they show similar behaviors across two graphs from different domains, while B_2 is an instance of *unshared anomalies* since such type of anomalies only exist in the target graph.

unlabeled attributed graph (target graph) can be improved by leveraging another labeled attributed graph (source graph). Recent advancements on domain adaptation have shown promising results in learning domain-invariant features across domains in various research disciplines, including computer vision (Saenko et al. 2010; Ganin and Lempitsky 2014; Hoffman et al. 2014) to natural language processing (Collobert et al. 2011; Glorot, Bordes, and Bengio 2011). In light of this, we propose to tackle the novel problem of cross-domain graph anomaly detection by adapting domain discrepancies between two attributed graphs.

Despite the unprecedented success of deep domain adaptation, directly grafting it for detecting anomalies on attributed graphs is infeasible due to the following challenges. First, compared to conventional text or image data, attributed graphs are notoriously difficult to handle due to the data heterogeneity from both structure and attribute perspectives (Ding et al. 2019). As such, applying conventional domain adaptation techniques to our problem may result in unsatisfactory results as they are not tailored for attributed graphs. Therefore, the first challenge centers around how to model two arbitrarily structured attributed graphs from different domains and learn domain-invariant node representations for detecting anomalies. Second, in order to detect anomalies on the unlabeled target graph, one straightforward solution is to train a domain-adapted classifier as existing work shows (Ganin and Lempitsky 2014; Tzeng et al. 2017; Qu et al. 2019). However, the domain-adapted classifier may render unsatisfactory anomaly detection performance. Figure 11 shows an example of detecting anomalies on attributed graphs in the cross-domain setting. As we can see, the labeled fraudulent reviewers in the Books domain (e.g., A_1) continuously spread promotion links instead of reviewing books, which can be treated as a typical type of anomalies. Although we are able to detect the anomalies that reveal similar behaviors (i.e., *shared anomalies*) in the Clothes domain (e.g., B_1) by domain adaptation, domain B has another type of fraudulent reviewers who generate negative reviews to sabotage the reputation of targeted products (e.g., B_2). The domain-adapted classifier may not work well for detecting such type of anomalies (i.e., *unshared anomalies*) since they do not appear in the source domain graph. Therefore, the second challenge lies in how to spot both the *shared* and *unshared anomalies* on the target graph simultaneously.

In this work, we propose COMMANDER (cross-domain anomaly detection on attributed networks), a novel end-to-end framework which consists of four principled components to address the above challenges. For the first challenge, COMMANDER employs a shared graph attentive encoder building on top of the graph attention networks (Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018c) to learn node representations of both source and target attributed graphs. Meanwhile, by deceiving the domain discriminator to distinguish the domain assignment of nodes, the graph attentive encoder gradually maps node representations from both source and target graphs to a domain-invariant feature space. For the second challenge, COMMANDER can detect the *shared anomalies* with the domain-adapted anomaly classifier trained from the labeled source graph. Meanwhile, COMMANDER uses an attribute decoder to spot the *unshared anomalies* by measuring the attribute reconstruction error of each node. As such, the synergistic collaboration between anomaly classifier and attribute decoder empowers COMMANDER to achieve superior anomaly detection performance on the target graph. To summarize, our contributions of this study are as follows:

- ***Problem***: To the best of our knowledge, we are the first to study the novel problem of cross-domain graph anomaly detection. In particular, we emphasize its importance and give a formal problem definition.
- ***Algorithm***: We develop an end-to-end framework for cross-domain graph anomaly detection. The proposed framework bridges the domain discrepancy between two attributed graphs and detects both the shared and unshared anomalies on the target graph.
- ***Evaluation***: We perform extensive experiments on real-world datasets to verify the effectiveness of our proposed model. The experimental results demonstrate its superior performance for cross-domain graph anomaly detection.

3.2.2 Related Works

Domain adaptation (Pan and Yang 2009) aims at mitigating the generalization bottleneck introduced from domain shift. With the rapid growth of deep neural networks, deep domain adaptation has drawn much attention lately. In general, deep domain adaptation methods are trying to locate a domain-invariant feature space that can reduce the differences between the source and target domains. This goal is accomplished either by transforming the features from one domain to be closer to the other domain, or projecting both domains into a domain-invariant latent space (Ganin and Lempitsky 2014; J. Yu et al. 2018; Shu et al. 2019). For instance, Tzeng et al. (Tzeng et al. 2014) leverage an adaptation layer and a domain confusion loss to learn the domain-invariant representations. TLDA (Zhuang et al. 2015) is a deep autoencoder-based model which tries to learn to domain-invariant representations and useful for label classification. Inspired by the idea of Generative Adversarial Network (GAN) (Goodfellow et al. 2014), researchers also propose to perform domain adaptation in an adversarial training paradigm (Ganin and Lempitsky 2014; Ganin et al. 2016; Tzeng et al. 2017; Shu et al. 2019). By exploiting a domain discriminator to distinguish the domain labels while learning deep features to confuse the discriminator, DANN (Ganin et al. 2016) achieves superior domain adaptation performance. ADDA (Tzeng et al. 2017) learns a discriminative representation using labeled source domain data and then map the target data to the same space through an adversarial loss. Later on, researchers also try to apply domain adaptation techiniques on graph-structured data (Y. Zhang et al. 2019; Dai et al. 2019; Shen et al. 2020; M. Wu et al. 2020) to handle the domain discrepancy between source and target graphs. For example, DANE (Y. Zhang et al. 2019) applies a shared weight graph convolutional network architecture with constraints of adversarial learning regularization, enabling cross-network knowledge transfer fro unsupervised network embedding. Similarly, UDA-GCN (M. Wu et al. 2020) further propose a dual graph convolutional networks to capture both the local and global consistency relationship of each graph, and use inter-graphed based attention mechanism to better represent each node. However, cross-domain anomaly detection remains unsolved in the graph learning community.

3.2.3 Anomaly Analysis Across Domains.

To gain insight into the relations between anomalies in a single domain or across different domains, we conduct an initial exploration on a pair of real-world datasets covering two different domains (i.e., *Hotel* and *Restaurant*) in Yelp (The details of the datasets are introduced in Section 3.2.6.1). There are regular users and anomalies in both domains. In this analysis, we regard *Hotel* as our target domain for which we want to detect anomalies. As shown in Figure 12, we compare the cosine similarity between different user pairs. Note that each user is represented with a feature vector constructed with the bag-of-word features from all his/her reviews. For Group 3 (G3), we calculate the similarity between each anomaly and all the regular users in *Hotel* and show the average value for each anomaly. Compared with G1, in which we show the average similarity between each anomaly and the other anomalies in the same domain, the values in G3 are significantly smaller. Such discrepancy between anomalies and regular users–which represent the majority of users in the platform– can be utilized for anomaly detection under the unsupervised setting. To investigate whether the labeled anomalies in the source domain (*Restaurant* in this case) can give guidance to anomaly detection in *Hotel*, we evaluate the similarities between anomalies across these two domains (shown in G2). The fact that the anomalies in *Hotel* are closer to anomalies in *Restaurant* than regular users in *Hotel* demonstrates that the supervised information from the source domain (*Restaurant*) can be potentially leveraged for detecting anomalies in the target domain (*Hotel*). However, the values of similarity in G2 are still smaller than those in G1, meaning that there exist some anomalies in *Hotel* revealing unshared patterns compared with anomalies in *Restaurant*. We

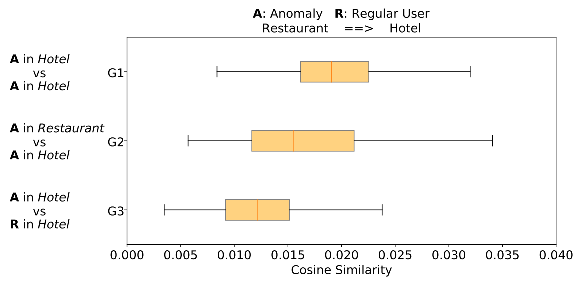


Figure 12. Cross-domain data analysis *w.r.t.* feature similarity between different user groups.

observe similar data patterns in other pairs of cross-domain datasets, which motivates our design of Commander.

3.2.4 Problem Definition

To legibly describe the studied problem, we follow the commonly used notations as introduced in Section 2.1.3. In order to provide more interpretable results, graph anomaly detection is commonly considered as a ranking problem (Akoglu, Tong, and Koutra 2015; Ding et al. 2019). Accordingly, we define the problem of cross-domain graph anomaly detection as follows:

Problem Definition 3 ***Cross-domain Graph Anomaly Detection****: Given a labeled attributed graph $G^s = (\mathbf{X}^s, \mathbf{A}^s)$ from the source domain and another unlabeled attributed graph $G^t = (\mathbf{X}^t, \mathbf{A}^t)$ from the target domain, here we follow previous works and assume G^s and G^t share the same feature space but do not have overlapped nodes or edges. The objective is to learn an anomaly detection model, which is capable of generalizing the knowledge from the labeled graph G^s, to detect the anomalies on the target graph $\mathbf{G}^t$. Ideally, anomalous nodes should be ranked on higher positions over normal nodes in the returned list.*

3.2.5 Architecture Overview

In this section, we present the details of the proposed framework that consists of four dedicated components (see Figure 13): (1) a graph attentive encoder; (2) a domain discriminator; (3) an anomaly classifier; and (4) an attribute decoder. Specifically, COMMANDER accomplishes domain adaptation on attributed graphs with the graph attentive encoder and domain discriminator. The anomaly classifier and attribute decoder are employed to detect anomalies on the target attributed graph synergistically.

3.2.5.1 Domain Adaptation on Attributed Graphs

Deep domain adaptation has recently drawn much attention with the booming development of deep neural networks (DNNs). Those deep domain adaptation methods have been proven to be effective in different learning tasks, such as image classification, sentiment classification and text matching (Ganin and Lempitsky 2014; Qu et al. 2019). The main intuition behind these methods is to learn the domain-invariant representations of combined samples from both source and target domains. In order to perform cross-domain anomaly detection on attributed graphs, we propose to follow a prevalent line of study (Ganin and Lempitsky 2014; Zheng Li et al. 2018; Shu et al. 2019) and first employ a shared encoder to extract the latent representation of each node in both G^s and G^t. However, apart from the image or text data that we can directly feed the combined samples from both source and target domains into a shared feature extractor, different attributed graphs have distinctive topological structures.

Thus, it is unclear that how we can model two arbitrarily structured attributed graphs using a shared encoder.

Graph Attentive Encoder (*Enc*). To counter this problem, we build our shared encoder grounded on the graph attention networks (Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018c) (GATs). GAT is an attention-based GNN model that allows specifying fine-grained weights when aggregating information from neighbors (as shown in Figure 13). Formally, in each layer l, node v_i integrates the features of neighboring nodes to obtain representations of layer $l+1$ via:

$$\mathbf{h}_i^{(l+1)} = \sigma\Big(\sum_{j \in \mathcal{N}_i \cup v_i} \alpha_{ij} \mathbf{W}\mathbf{h}_j^{(l)}\Big), \tag{3.9}$$

where σ denotes the nonlinear activation function (e.g., ReLU). $\mathcal{N}_i$ denotes the set of neighbors for v_i and α_{ij} is the attention coefficient between node v_i and node v_j, which can be computed as:

$$\alpha_{ij} = \frac{\exp\big(\sigma\big(\mathbf{a}^{\mathrm{T}}[\mathbf{W}\mathbf{h}_i^{(l)} \oplus \mathbf{W}\mathbf{h}_j^{(l)}]\big)\big)}{\sum_{k \in \mathcal{N}_i \cup v_i} \exp\big(\sigma\big(\mathbf{a}^{\mathrm{T}}[\mathbf{W}\mathbf{h}_i^{(l)} \oplus \mathbf{W}\mathbf{h}_k^{(l)}]\big)\big)}, \tag{3.10}$$

where $\oplus$ is the concatenation operation and the attention vector $\mathbf{a}$ is a trainable weight vector that assigns importance to the different neighbors of node v_i, allowing the model to highlight the features of the important neighboring node that is more task-relevant.

The benefits of using graph attention networks are mainly two-fold: (1) graph attention networks employ a trainable aggregator function to learn the representation of each node, which eliminates the dependency on the global graph structure. In this way, our shared encoder is capable of learning node representations for both G^s and G^t (Veličković, Cucurull, Casanova, Romero, Lio, et al. 2018c); (2) due to the fact that malicious users might build spurious connections with normal users to camouflage their noxious intentions, graph attention networks can better assess the abnormality of each node by specifying fine-grained attentions on the neighboring nodes. Thus, the graph attentive encoder is able to learn high-quality node representations from the two attributed graphs G^s and G^t. Moreover, we build the graph attentive encoder Enc with multiple GAT layers:

$$\begin{aligned} \mathbf{h}_i^{(1)} &= \sigma\big(\sum_{j \in \mathcal{N}_i \cup v_i} \alpha_{ij}^{(1)} \mathbf{W}^{(1)} \mathbf{x}_j\big), \\ &\dots \\ \mathbf{z}_i &= \sigma\big(\sum_{j \in \mathcal{N}_i \cup v_i} \alpha_{ij}^{(L)} \mathbf{W}^{(L)} \mathbf{h}_j^{(L-1)}\big), \end{aligned} \tag{3.11}$$

where $\mathbf{z}_i$ is the latent representation of node i. In this way, the graph attentive encoder Enc can capture the non-linearity of topological structure and nodal attributes. Following previous domain adaptation works (Ganin et al. 2016; M. Wu et al. 2020), we use Enc as a shared architecture and encodes G^s and G^t one by one in each epoch. This way the graph attentive encoder is able to map the learned node representations from two graphs to an aligned embedding space and further enables knowledge transfer across graphs from different domains.

Domain Discriminator (*Dis*). In order to further perform domain adaptation on two attributed graphs from different domains, we adopt the idea of adversarial machine learning (Goodfellow et al. 2014) to perform adversarial domain adaptation (Tzeng et al. 2017; Pan et al. 2019) in a two-player minimax game. As illustrated in Figure 13, the first player is the domain discriminator Dis which tries to distinguish whether an embedded node is from the source domain or the target domain, and the second player is the graph attentive encoder Enc which is adversarially trained to deceive the domain discriminator. The domain discriminator Dis is built with a feed-forward

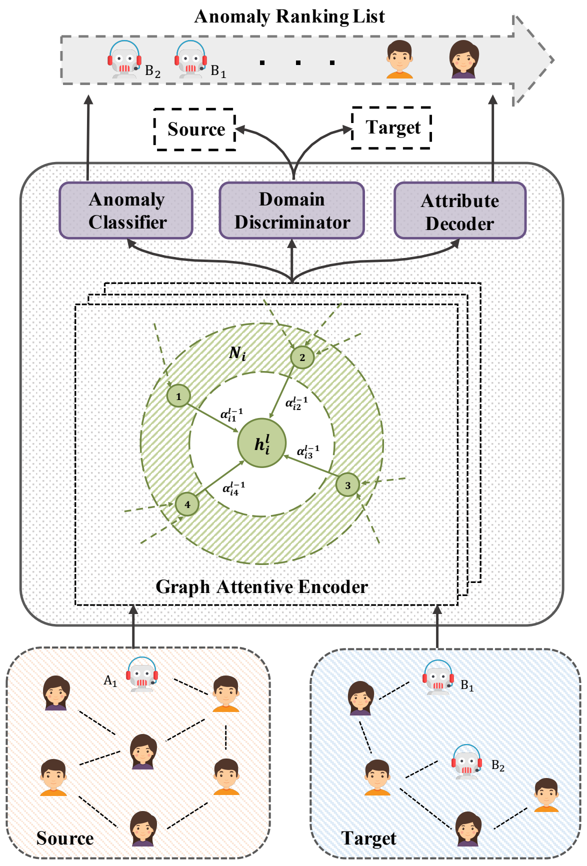


Figure 13. Overview of the COMMANDER framework for cross-domain graph anomaly detection. Figure best viewed in color.

layer with tanh non-linearity, followed by a sigmoid function:

$$\begin{aligned}\mathbf{o}_i^D &= \tanh(\mathbf{W}^D \mathbf{z}_i + \mathbf{b}^D),\\ \hat{y}_i &= \mathrm{sigmoid}(\mathbf{u}^{\mathrm{T}} \mathbf{o}_i^D),\end{aligned} \tag{3.12}$$

where $\mathbf{W}^D$ and $\mathbf{b}^D$ denote the trainable parameter matrix and bias, respectively. $\mathbf{o}_i^D$ is the output of the feed-forward layer. Here $\mathbf{u}$ is another trainable weight vector, and $\hat{y}_i$ is the predicted domain label. The adversarial domain loss can be mathematically formulated as:

$$\mathcal{L}_D = -\frac{1}{N_D} \sum_{i=1}^{N_D} \Big[d_i \log \hat{y}_i + (1 - d_i) \log(1 - \hat{y}_i) \Big], \tag{3.13}$$

where N_D denotes the number of all the nodes in both G^s and G^t. Here d_i represents the domain label of node i and $\hat{y}_i$ is the predicted domain label.

Since our goal is to bridge the domain discrepancy between two graphs, here we choose to maximize the above cross-entropy loss. In other words, after the feature encoding phase, the domain label of nodes would not be accurately recognized by the domain discriminator, and the shared graph attentive encoder would be able to extract domain-invariant node representations from both source graph G^s and target graph G^t.

3.2.5.2 Cross-domain Anomaly Detection

In the previous subsection, we have discussed how to bridge the domain discrepancy between two attributed graphs from different domains. This subsection introduces how to detect both *shared anomalies* and *unshared anomalies* on the target graph G^t.

Anomaly Classifier (*Clf*). Following the idea of other domain adaptation learning tasks (Shu et al. 2019), we train an anomaly classifier *Clf* right after the shared graph attentive encoder, to distinguish whether a node from G^s is an anomaly or not. *Clf* is built with a feed-forward layer with tanh non-linearity, followed by a sigmoid function:

$$\begin{aligned} \mathbf{o}_i^C &= \tanh(\mathbf{W}^C\mathbf{z}_i + \mathbf{b}^C), \\ \bar{y}_i &= \text{sigmoid}(\mathbf{v}^{\mathrm{T}}\mathbf{o}_i^C), \end{aligned} \tag{3.14}$$

where $\mathbf{W}^C$ and $\mathbf{b}^C$ are the trainable parameter matrix and bias, $\mathbf{v}$ is a trainable weight vector. Specifically, the anomaly classification loss can be defined as the binary cross-entropy:

$$\mathcal{L}_C = -\frac{1}{N_C}\sum_{i=1}^{N_C}\Big[y_i \log \bar{y}_i + (1-y_i)\log(1-\bar{y}_i)\Big], \tag{3.15}$$

where N_C denotes the number of nodes sampled from the labeled graph G^s. y_i and $\bar{y}_i$ denote the ground truth anomaly label and the predicted anomaly label of node i, respectively. Note that here we sample an equal number of normal nodes and abnormal nodes from G^s for addressing data imbalance. The shared graph attentive encoder maps data from different domains to a domain-invariant feature space by deceiving the domain discriminator, then the domain-adapted anomaly classifier can be directly used for detecting the *shared anomalies* on the target attributed graph.

Nevertheless, one critical issue is that not all anomalies share similar characteristics across graphs from different domains. As discussed in the previous sections, some specific types of anomalies that exist in G^t may not appear in G^s. Thus solely relying on a classifier trained on the labeled source graph cannot accurately trace such *unshared anomalies*, rendering unsatisfactory anomaly detection performance on the target attributed graph.

Attribute Decoder (*Dec*). As suggested by recent studies (Xia et al. 2015; Li, Dani, Hu, and Liu 2017; Ding et al. 2019), the reconstruction error between original data and estimated data is a strong indicator to show the abnormality of each data instance. The intuition is that anomalies usually cannot be well reconstructed from the observed data and have large reconstruction errors since their patterns deviate significantly from the majority. Therefore, we build an attribute decoder *Dec* following the graph attentive encoder for reconstructing two attributed graphs. Since node dependency information is inherently encoded in each GAT layer, we propose to reconstruct the node attributes for simplicity. Specifically, we build *Dec* with multiple GAT layers:

$$\begin{aligned}
\mathbf{h}_i^{(L+1)} &= \sigma\big(\sum_{j\in\mathcal{N}_i\cup v_i} \alpha_{ij}^{(L+1)}\mathbf{W}^{(L+1)}\mathbf{z}_j\big),\\
&\dots\\
\tilde{\mathbf{x}}_i &= \sigma\big(\sum_{j\in\mathcal{N}_i\cup v_i} \alpha_{ij}^{(\tilde{L})}\mathbf{W}^{(\tilde{L})}\mathbf{h}_j^{(\tilde{L}-1)}\big),
\end{aligned} \tag{3.16}$$

where $\tilde{\mathbf{x}}_i$ is the estimated attribute of node v_i. The reconstruction error computed by this deep autoencoder network provides a precise assessment of node abnormality (Zhou and Paffenroth 2017; Chalapathy, Menon, and Chawla 2017; Ding et al. 2019) and enables us to spot the *unshared anomalies.* Specifically, the reconstruction loss can be defined as:

$$\mathcal{L}_R = ||\widetilde{\mathbf{X}}^s - \mathbf{X}^s||_F^2 + ||\widetilde{\mathbf{X}}^t - \mathbf{X}^t||_F^2, \tag{3.17}$$

where $\widetilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_n]$ denotes the reconstructed attribute matrix of a graph.

In this way, our anomaly classifier and attribute decoder are able to synergistically perform anomaly detection on the target attributed graph. Intuitively, the anomaly classifier would spot the *shared anomalies* with high precision, meanwhile the attribute decoder is capable of providing complementary insight for detecting the *unshared anomalies.* As another benefit, the incorporation of the attribute decoder can also improve the feature learning quality of the graph attentive encoder through back-propagation, and relieve the overfitting problem when training the anomaly classifier (LeCun, Bengio, and Hinton 2015).

3.2.5.3 Model Learning

So far, we have introduced the architecture of our framework Commander for solving the problem of cross-domain graph anomaly detection. This joint architecture requires dedicated training objective for each component. The complete objective function can be formulated as follows:

$$\begin{aligned}\mathcal{L} &= -\mathcal{L}_D + \mathcal{L}_C + \mathcal{L}_R \\ &= \frac{1}{N_D}\sum_{i=1}^{N_D}\Big[d_i \log \hat{y}_i + (1-d_i)log(1-\hat{y}_i)\Big] \\ &- \frac{1}{N_C}\sum_{i=1}^{N_C}\Big[y_i \log \bar{y}_i + (1-y_i)log(1-\bar{y}_i)\Big] \\ &+ ||\widetilde{\mathbf{X}}^s - \mathbf{X}^s||_F^2 + ||\widetilde{\mathbf{X}}^t - \mathbf{X}^t||_F^2.\end{aligned} \tag{3.18}$$

We summarize the training procedure of Commander in Algorithm 1. By minimizing the dedicated objective functions, Commander gradually closes the domain shift between G^s and G^t, and learns a powerful anomaly detector. All the parameters of Commander are optimized by the standard back-propagation algorithm (LeCun, Bengio, and Hinton 2015). Specifically, for each node, we use the output from *Clf* as a learned weight to re-weight the reconstruction errors from *Dec*, and the final anomaly score of node v_i can be formulated as:

$$score(v_i) = \bar{y}_i||\tilde{\mathbf{x}}_i - \mathbf{x}_i||_2^2, \tag{3.19}$$

where $\bar{y}_i \in [0,1]$ and the final scores represent the node abnormality computed by both the anomaly classifier and the attributed decoder.

3.2.6 Performance Evaluation

3.2.6.1 Experiment Settings

Evaluation Datasets. To evaluate the performance of different methods, we adopt two pairs of real-world datasets for evaluation. All the datasets are public and have

Table 10. Statistics of the Real-world Datasets.

	YelpHotel	$\rightleftharpoons$	YelpRes	YelpNYC	$\rightleftharpoons$	Amazon
# nodes	5,196		5,102	21,040		18,601
# edges	171,743		239,738	303,949		274,458
# attributes	8,000		8,000	10,000		10,000
# anomalies	250		275	1000		750

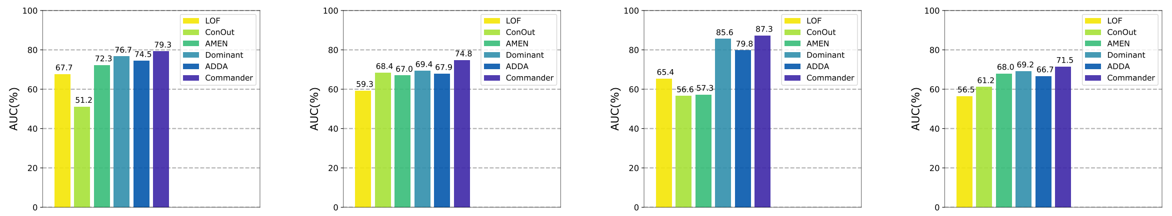
been widely used for graph anomaly detection problems (Rayana and Akoglu 2015; Kaghazgaran, Caverlee, and Squicciarini 2018). The dataset statistics are listed in Table 10 and we summarize the details of those two dataset pairs as follows:

- **YelpHotel $\rightleftharpoons$ YelpRes**: YelpHotel and YelpRes are collected from Yelp on two major business domains, i.e., hotel and restaurant, in the Chicago area (Rayana and Akoglu 2015). For each dataset, users are considered as nodes and a link will be created if two users commented on the same hotel or same restaurant. By using the Yelp anti-fraud filter, the users from each dataset can be separated into two classes: anomaly (authors of filtered reviews), and regular users (authors with no filtered reviews), which can be considered as the ground truth labels.
- **YelpNYC $\rightleftharpoons$ Amazon**: To further study the effect of different levels of domain discrepancy on the performance improvements, we also adopt another pair of attributed graphs collected from two different platforms (domains with higher discrepancy), i.e., Yelp and Amazon. Specifically, YelpNYC collects data for the restaurants located in New York City (Rayana and Akoglu 2015). Amazon is another attributed graph collected from an E-commerce platform by (Kaghazgaran, Caverlee, and Squicciarini 2018). In this dataset, a user is flagged as a fraudulent user if he/she has reviewed two or more products that have been targeted by crowdsourcing efforts (Kaghazgaran, Caverlee, and Squicciarini 2018), otherwise the user is considered as legitimate.

For all the datasets above, we apply bag-of-words model (Zhang, Jin, and Zhou 2010) to obtain the attributes of each node. The vocabulary is built on top of the textual contents related to the nodes from both source and target graphs. With the processed datasets, we are able to conduct the evaluation across 4 domain shifts in our experiments, including **YelpHotel** $\rightarrow$ **YelpRes**, **YelpRes** $\rightarrow$ **YelpHotel**, **YelpNYC** $\rightarrow$ **Amazon**, and **Amazon** $\rightarrow$ **YelpNYC**. Notably, "A $\rightarrow$ B" represents the task which aims at detecting anomalies on the target domain attributed graph B, by adapting the knowledge from the labeled source domain attributed graph A. In addition, as anomalies usually consist of a small portion of a dataset, we randomly sampled out part of the spammers or fraudulent reviewers to make our experiments more realistic and challenging.

Compared Methods. In the experiments, we compare the proposed framework Commander with several state-of-the-art representative anomaly detection methods. Specifically, **LOF** (Breunig et al. 2000) detects anomalies at the contextual level and only considers nodal attributes. **ConOut** (Sánchez et al. 2014) detects anomalies in the local context by determining its subgraph and its relevant subset of attributes. **AMEN** (Perozzi and Akoglu 2016) uses both attribute and graph structure information to detect anomalous neighborhoods. Specifically, it analyzes the abnormality of each node from the ego-network point of view. **DOMINANT** (Ding et al. 2019) is the state-of-the-art model for detecting anomalies on attributed graphs. By developing a graph convolutional networks based autoencoder, the reconstruction errors can be used for spotting anomalies. **ADDA** (Tzeng et al. 2017) is an adversarial domain adaptation model for image classification. We adopt the architecture of this model to conduct cross-domain graph anomaly detection by omitting the graph structures.

Due to the fact that cross-domain graph anomaly detection remains an under-



(a) YelpHotel→YelpRes (b) YelpRes→YelpHotel (c) YelpNYC→Amazon (d) Amazon→YelpNYC

Figure 14. Results of cross-domain graph anomaly detection w.r.t. AUC scores.

Table 11. Results of cross-domain graph anomaly detection w.r.t. Precision@K on YelpRes and YelpHotel.

Precision@K						
	YelpHotel → YelpRes			YelpRes → YelpHotel		
K	50	150	250	50	150	250
LOF	0.460	0.260	0.176	0.440	0.213	0.172
ConOut	0.260	0.107	0.064	0.480	0.280	0.216
AMEN	0.040	0.073	0.092	0.160	0.113	0.080
DOMINANT	0.580	0.327	0.236	0.560	0.320	0.224
ADDA	0.460	0.233	0.176	0.500	0.247	0.172
COMMANDER	**0.620**	**0.360**	**0.244**	**0.600**	**0.347**	**0.228**

studied task, it is worth mentioning that none of the above methods is exactly developed for solving our studied problem. Since no labels are available on the target graph, we first select four state-of-the-art baselines (i.e., LOF, ConOut, AMEN and DOMINANT) for unsupervised anomaly detection on attributed graphs. We directly run each of them on the target graph, and report the corresponding detection performance to make a fair comparison. Additionally, we also compare with ADDA, which is a state-of-the-art domain adaptation method. As it is not designed for graph-based anomaly detection problem, we omit the topological structure and use the probability predicted by ADDA to rank all the nodes on the target graph.

Table 12. Results of cross-domain graph anomaly detection w.r.t. Precision@K on YelpNYC and Amazon.

Precision@K						
	YelpNYC → Amazon			Amazon → YelpNYC		
K	50	150	250	50	150	250
LOF	0.140	0.073	0.052	0.380	0.200	0.168
ConOut	0.040	0.020	0.012	0.660	0.407	0.328
AMEN	0.020	0.013	0.012	0.580	0.333	0.264
DOMINANT	0.480	0.433	0.444	0.620	0.407	0.320
ADDA	0.380	0.220	0.184	0.540	0.353	0.312
COMMANDER	**0.500**	**0.460**	**0.456**	**0.680**	**0.420**	**0.332**

Table 13. Results of cross-domain graph anomaly detection w.r.t. Recall@K on YelpRes and YelpHotel.

Recall@K						
	YelpHotel → YelpRes			YelpRes → YelpHotel		
K	50	150	250	50	150	250
LOF	0.084	0.142	0.160	0.088	0.128	0.172
ConOut	0.047	0.058	0.058	0.096	0.168	0.216
AMEN	0.007	0.040	0.084	0.032	0.068	0.080
DOMINANT	0.105	0.178	0.215	0.112	0.192	0.224
ADDA	0.084	0.127	0.160	0.100	0.148	0.172
COMMANDER	**0.113**	**0.196**	**0.222**	**0.120**	**0.208**	**0.228**

3.2.6.2 Evaluation Results

Firstly, we evaluate the performance of the proposed framework COMMANDER and other unsupervised baseline methods on four different domain shifts. The results with respect to AUC scores are presented in Figure 14. We also report the Precision@K scores and Recall@K scores in Table 11, Table 12, Table 14 and Table **??**, respectively. From a comprehensive view, we can clearly find our approach COMMANDER achieves considerable improvements over the state-of-the-art unsupervised methods on all the domain shifts. Take AUC value as an example, the performance of COMMANDER

Table 14. Results of cross-domain graph anomaly detection w.r.t. Recall@K on YelpNYC and Amazon.

Recall@K						
	YelpNYC → Amazon			Amazon → YelpNYC		
K	50	150	250	50	150	250
LOF	0.009	0.015	0.017	0.019	0.030	0.042
ConOut	0.003	0.004	0.004	0.033	0.061	0.082
AMEN	0.001	0.003	0.004	0.029	0.050	0.066
DOMINANT	0.032	0.087	0.148	0.031	0.061	0.080
ADDA	0.025	0.044	0.061	0.027	0.053	0.078
COMMANDER	**0.033**	**0.092**	**0.152**	**0.034**	**0.063**	**0.083**

is 2.6% higher than the best baseline on the YelpHotel → YelpRes case, and the corresponding improvements on YelpHotel → YelpRes, YelpNYC → Amazon, Amazon → YelpNYC are reported with 5.4%, 1.7% and 1.6%, respectively. Meanwhile, our approach consistently outperforms the best performing baselines according to Precision@K and Recall@K results, which indicates that COMMANDER is capable of discovering more anomalous nodes in its top return lists and once again demonstrates the effectiveness our approach.

Note that the unsupervised methods, including LOF, ConOut and AMEN, cannot achieve competitive results in comparison. In particular, the performance of LOF is limited by its inability of modeling node dependencies. We also observe that AMEN performs poorly in the task of ranking anomalous nodes. One explanation is that AMEN is designed for detecting anomalous neighborhoods rather than nodes. Even though DOMINANT performs best amongst all the unsupervised methods owing to the excellent expressive power of graph convolutional network (GCN), it is still largely behind our approach as it is unable to accurately spot those *shared anomalies* by utilizing labeled data from the source graph.

Next, we compare the performance of the domain adaptation method ADDA with our proposed framework COMMANDER. With the reported results (w.r.t. AUC scores), we observe that COMMANDER outperforms ADDA by a significant margin, reaching around 10% to 20% relative improvement in most cases. Meanwhile, as shown in Table 11, Table 12 Table 14 and Table **??**, COMMANDER is able to discover more true anomalies on its top anomaly ranking list than ADDA. There are two major reasons that result in the ineffectiveness of ADDA for the studied problem: First, node dependency information is indispensable for assessing the abnormality of a node while ADDA cannot model such information modality; Second, ADDA is unable to detect the *unshared anomalies* on the target graph since it is not tailored for anomaly detection problem. On the contrary, our approach COMMANDER is able to detect *unshared anomalies* on the target graph using the Attribute Decoder *Dec*.

Additionally, the results show that our approach is able to achieve larger improvements in the first two domain shifts than the last two. Compared with the attributed graphs YelpHotel and YelpRes, the attributed graphs YelpNYC and Amazon are not only from two different business domains, but also from two different platforms. Thus, this observation implies that the model performance is strongly associated with the degree of domain discrepancy. In brief, smaller domain discrepancy could be easier adapted, leading to better cross-domain anomaly detection performance.

3.2.6.3 Ablation Study

To investigate how much is the contribution of each component, in this subsection, we design the ablation study and show the corresponding experimental results. Specifically, we compare our proposed framework COMMANDER with the following three variants:

Table 15. Ablation results on two cross-domain settings: YelpHotel → YelpRes and YelpRes → YelpHotel.

Methods	YelpHotel → YelpRes		YelpRes → YelpHotel	
	Pre@50	AUC	Pre@50	AUC
Clf	0.280	0.461	0.220	0.431
Clf+Dis	0.500	0.758	0.420	0.688
Dec	0.540	0.765	0.540	0.695
w/o GAT	0.580	0.776	0.580	0.722
COMMANDER	0.620	0.793	0.600	0.748

- *Clf*: We exclude the domain discriminator and attribute decoder from COMMANDER, and only use the anomaly classifier to detect anomalies on the target domain attributed graph G^t.
- *Clf+Dis*: We exclude the attribute decoder from the proposed framework COMMANDER and use the anomaly classifier and domain discriminator to detect anomalies on the target domain attributed graph G^t.
- *Dec*: We exclude the anomaly classifier and domain discriminator from the proposed framework COMMANDER and only employ attribute decoder for detecting anomalies on the target domain attributed graph G^t.
- *w/o GAT*: We replace the GAT layers in COMMANDER with GCN layers to examine the effectiveness of using GAT for anomaly detection.

The comparison results on YelpHotel → YelpRes and YelpRes → YelpHotel are shown in Table 15, and the results on YelpNYC → Amazon and Amazon → YelpNYC are shown in Table 16. Due to the space limit, we only show the results in terms of Precison@50 and AUC in our ablation study. From the reported results, we make the following observations:

- By examining the performance of *Clf* on four domain shifts, we can clearly find that it performs poorly overall. On the contrary, the variant *Clf+Dis* improves the detection performance to a large extent with the join of *Dis*, which demonstrates that an anomaly classifier trained on the G^s cannot be directly used on G^t without domain adaptation.

- Comparing to the variant *Clf+Dis*, *Dec* achieves superior detection performance in our experiments. The reasonable explanation is that the attribute decoder provides a more comprehensive assessment and is capable of detecting both *shared anomalies* and *unshared anomalies* to some extent.
- By replacing the GAT layers in the COMMANDER framework with vanilla GCN layers, the performance decreases a noticeable margin, which shows the advantage of using graph attention mechanism for detecting anomalies.
- Despite *Clf+Dis* and *Dec* considerably improve the detection performance, they still cannot achieve competitive results with our approach COMMANDER in the evaluations. It validates our assumption that *Dis* assists the anomaly classifier *Clf* to detect the *shared anomalies*, meanwhile *Dec* is the key component to detect those *unshared anomalies* on the target graph.

Table 16. Ablation results on two cross-domain settings: YelpNYC → Amazon and Amazon → YelpNYC.

Methods	YelpNYC → Amazon		Amazon → YelpNYC	
	Pre@50	AUC	Pre@50	AUC
Clf	0.040	0.558	0.320	0.445
Clf+Dis	0.420	0.812	0.560	0.677
Dec	0.480	0.848	0.600	0.696
w/o GAT	0.460	0.857	0.640	0.702
COMMANDER	0.500	0.873	0.680	0.715

3.3 Conclusion

In this chapter, firstly, for solving the problem of node classification with *incomplete* supervision, we propose a new graph meta-learning framework, Meta Label Propagation (Meta-LP). Based on the meta-learned label propagation strategy, we are able to generate informative pseudo labels on unlabeled nodes, in order to augment the insufficient labeled data and learn powerful GNN model. Though built with simple neural networks, Meta-LP effectively enables larger receptive fields and avoid oversmoothing when learning with very few labeled data. We test Meta-LP on a spectrum of benchmark datasets and the results well demonstrate its effectiveness. For future work, it would be interesting to investigate other pseudo-labeling strategies for solving the studied problem.

Secondly, to leverage the *indirect* supervision, we propose a novel cross-domain anomaly detection framework called COMMANDER, to tackle the problem of anomaly detection in the unlabeled attributed graphs. The proposed framework consists of four principled components: graph attentive encoder, anomaly classifier, domain discriminator and attribute decoder. These components are tightly coupled to bridge the domain discrepancy between two attributed graphs from different domains and then perform accurate anomaly detection on the target attributed graph. We perform extensive experiments to corroborate the effectiveness of the proposed COMMANDER framework.

Chapter 4

GRAPH SELF-SUPERVISED LEARNING

To alleviate the demand for massive labeled data and provide sufficient supervision, self-supervised learning has been introduced and achieved great success in various domains. In this chapter, I will elaborate my research on graph self-supervised learning in two lines: (1) *graph contrastive learning*; and (2) *graph generative modeling*.

In graph contrastive learning, most of the existing methods learn node representations solely based on the information from the local neighborhoods due to the shallow property of GNNs. While for real-world graphs, *nodes sharing similar characteristics may not always be geographically close*, requiring the learning algorithm to retain such "global" awareness. To address this critical problem, I designed a simple yet effective graph contrastive learning model S^3-*CL* to elicit global structural and semantic knowledge from the input graph, which demonstrates its superior performance over state-of-the-art methods.

For my research on graph generative modeling, I have developed a self-supervised graph anomaly detection models – *Dominant*, which have been considered as the representative works in deep graph anomaly detection. Based on the idea of reconstructing the input graph naturally serves as a self-supervision pretext for training the graph neural networks, *Dominant* performs both structure and feature reconstruction to capture the patterns of the majority nodes in a self-supervised fashion. This way the anomalies can be effectively detected by measuring the reconstruction errors.

4.1 Unsupervised Graph Contrastive Learning

4.1.1 Introduction

Learning expressive node representations of graph-structured data plays an essential role in a variety of real-world applications, ranging from social network analysis (Thomas N. Kipf and Welling 2017b), to drug discovery (Fout et al. 2017), to financial fraud detection (Ding et al. 2019). Recently, graph neural networks (GNNs), which generally follow a recursive message-passing scheme, have emerged as powerful architectures in graph machine learning (**sgc**; Thomas N. Kipf and Welling 2017b; Veličković, Cucurull, Casanova, Romero, Liò, et al. 2018; Hamilton, Ying, and Leskovec 2017; Ding, Wang, Li, Shu, et al. 2020; J. Wang et al. 2020). Though GNNs are empirically effective in handling supervised or semi-supervised graph machine learning tasks, the labor-intensive and resource-expensive data labeling cost is meanwhile unbearable (Ding, Xu, et al. 2022; C. Zhang et al. 2022; Ding, Zhang, et al. 2022). To relieve the burdensome reliance on human-annotated labels, unsupervised (self-supervised) node representation learning with GNNs has drawn much research attention lately (Thomas N Kipf and Welling 2017a; Veličković et al. 2019; Y. You et al. 2020).

More recently, contrastive learning (He et al. 2020; T. Chen et al. 2020) has been actively explored to advance the performance of GNNs in graph self-supervised learning (Veličković et al. 2019; Y. You et al. 2020; Hassani and Khasahmadi 2020; Qiu et al. 2020; Zhu et al. 2020b). In general, graph contrastive learning (GCL) methods learn representations by creating two augmented views of each graph element and maximizing the agreement between the encoded representations of the two augmented views. Correspondingly, the relevant view pairs (positive) will be pulled together, and the irrelevant view pairs (negative) will be pushed away in the latent space. With only non-semantic labels, unsupervised GCL can provide generalizable node representations for various downstream tasks (Y. You et al. 2020; Hassani and Khasahmadi 2020; Du et al. 2021), becoming a prevailing paradigm in unsupervised node representation learning.

Despite the success, the research of unsupervised GCL is still in its infancy – most of the existing GCL methods learn node representations based on the information from the local neighborhoods due to the shallow property of conventional GNNs. While for real-world graphs, *nodes sharing similar characteristics may not always be geographically close*, requiring the learning algorithm to retain such "global" awareness. However, it is a non-trivial task for the existing GCL methods built on top of shallow GNNs since they have inherent limitations in capturing either *structural global knowledge* or *semantic global knowledge*. Specifically: **(i)** from the structural perspective, long-range node interactions are highly desired for capturing structural global knowledge, especially for many downstream tasks that have large problem radii (Alon and Yahav 2021). To this end, a straightforward way is to employ a deeper GNN encoder to encode the augmented graphs. However, directly stacking multiple GNN layers will not only lead to information distortion caused by the oversmoothing issue (D. Chen et al. 2020), but also introduce additional training parameters that hamper the model training efficiency; and **(ii)** from the semantic perspective, existing unsupervised GCL methods predominately focus on instance-level contrast that leads to a latent space where all nodes are well-separated and each node is locally smooth (Junnan Li et al. 2021) (i.e., input with different augmentations have similar representations), while the underlying semantic structure (i.e., intra-cluster compactness and inter-cluster separability) of the input graph is largely ignored (Junnan Li et al. 2021). The lack of prior knowledge of ground-truth labels (e.g., cluster/class numbers) leaves a significant gap for unsupervised GCL to consolidate the semantic structure from a global view in the latent space. Yet, how to bridge this gap remains unattended.

In this work, we address the aforementioned limitations by proposing a simple yet effective GCL framework, namely, S^3-CL (*$\underline{S}$imple Neural Networks with $\underline{S}$tructural and $\underline{S}$emantic $\underline{C}$ontrastive $\underline{L}$earning*). The proposed two new contrastive learning algorithms enable the framework to outperform other GCL counterparts with a much simpler and parameter-less encoding backbone, such as an MLP or even a one-layer neural network. To capture long-range node interactions without oversmoothing, the *structural contrastive learning* algorithm first generates multiple augmented views of the input graph based on different feature propagation scales (i.e., multi-scale feature propagation). Then by performing contrastive learning on the node representations learned from the local and multiple high-order views, the encoder network can improve node-wise discrimination by exploiting the consistency between the local and global structure information of each node. In the meantime, the *semantic contrastive learning* algorithm further enhances intra-cluster compactness and inter-cluster separability to better consolidate the semantic structure from a global view. Specifically, it infers the clusters among nodes and their corresponding prototypes by a new Bayesian non-parametric algorithm and then performs semantic contrastive learning to enforce those nodes that are semantically similar to cluster around their corresponding cluster prototypes in the latent space. By jointly optimizing the structural and semantic contrastive losses, the pre-trained encoder network can learn highly expressive node representations for various downstream tasks without using any human-annotated labels. We summarize our contributions as follows:

- We develop a new GCL framework S^3-CL, which can learn expressive node representations in a self-supervised fashion by using a simple and parameter-less encoding backbone.
- We propose structural and semantic contrastive learning algorithms, which can be used for explicitly capturing the global structural and semantic patterns of the input graph.
- We conduct extensive experiments to show that our approach significantly outperforms the state-of-the-art GCL counterparts on various downstream tasks.

4.1.2 Preliminaries – Graph Contrastive Learning

We follow the commonly used notations for attributed graph as described in 2.1.3. In general, graph contrastive learning aims to pre-train a graph encoder that can maximize the node-wise agreement between two augmented views of the same graph element in the latent space via a contrastive loss. Generally, given an attributed graph $\mathcal{G} = (\mathbf{X}, \mathbf{A})$, two different augmented views of the graph, denoted as $\mathcal{G}^{(1)} = (\mathbf{X}^{(1)}, \mathbf{A}^{(1)})$ and $\mathcal{G}^{(2)} = (\mathbf{X}^{(2)}, \mathbf{A}^{(2)})$, are generated through the data augmentation function(s). The node representations on $\mathcal{G}^1$ and $\mathcal{G}^{(2)}$ are denoted as $\mathbf{H}^{(1)} = f_{\boldsymbol{\theta}}(\mathbf{X}^{(1)}, \mathbf{A}^{(1)})$ and $\mathbf{H}^{(2)} = f_{\boldsymbol{\theta}}(\mathbf{X}^{(2)}, \mathbf{A}^{(2)})$, where $f_{\boldsymbol{\theta}}(\cdot)$ is an encoder network. The agreement between the node representations is commonly measured through Mutual Information (MI). Thus, the contrastive objective can be generally formulated as:

$$\max_{\theta} \sum_{i=1}^{N} \mathcal{MI}(\mathbf{h}_i^{(1)}, \mathbf{h}_i^{(2)}). \tag{4.1}$$

Following this formulation, Deep Graph Infomax (DGI) (Veličković et al. 2019) is the first method that contrasts the patch representations with high-level graph representations by maximizing their mutual information. MVGRL (Hassani and

Khasahmadi 2020) adopts graph diffusion to generate an augmented view, and contrast representations of first-order neighbors with a graph diffusion. GCC (Qiu et al. 2020) and GRACE (Zhu et al. 2020a) create the augmented views by sampling subgraphs. MERIT (Jin et al. 2021) adopts a siamese self-distillation network and performs contrastive learning across views and networks at the same time. Nonetheless, existing unsupervised GCL methods only focus on short-range node interactions and are also ineffective in capturing the semantic structure of graphs.

4.1.3 Architecture Overview

In this work, we propose a novel graph contrastive learning framework S^3-CL for unsupervised/self-supervised node representation learning. The overall framework is illustrated in Figure 15. Our proposed framework consists of three main components: (i) a simple (e.g., 1-layer) encoder network; (ii) a structural contrastive learning algorithm; and (iii) a semantic contrastive learning algorithm.

4.1.3.1 Structural Contrastive Learning

Existing GCL methods for unsupervised node representation learning aim to achieve node-wise discrimination by maximizing the agreement between the representations of the same graph element in different augmented views. Despite their success, they commonly ignore the global structure knowledge due to the limitations of either the adopted data augmentation function or the GNN encoder. In this work, we propose the *structural contrastive learning* algorithm, which enables a simple neural network to

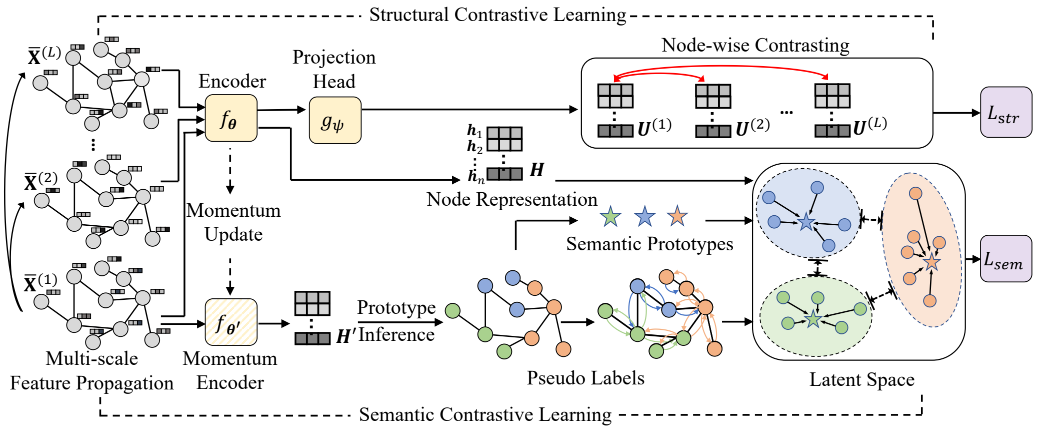


Figure 15. Illustration of the overall framework S^3-CL for self-supervised node representation learning.

capture both local and global structural knowledge by performing contrastive learning on multi-scale augmented graph views.

Multi-scale Feature Propagation. In order to capture long-range node interactions without suffering the oversmoothing issue, in our structural contrastive learning algorithm, we propose to adopt *multi-scale feature propagation* to augment the input graph from the structural perspective. Compared to arbitrarily modifying the graph structure such as perturbing edges or nodes, feature propagation not only allows incorporating long-range node interactions but also mitigates the noises in the original graph (Ding, Xu, et al. 2022). Unlike existing GCL algorithms that perform only two augmentations for each instance, we perform feature propagation with different scales to generate L augmented feature matrices $\{\bar{\mathbf{X}}^{(l)}\}_{l=1}^{L}$, each of which encodes the l-hop node interactions in the graph. Then each augmented feature matrix $\bar{\mathbf{X}}^{(l)}$ can be encoded by a encoder network $f_{\boldsymbol{\theta}}(\cdot)$ and the corresponding node representations can be computed by:

$$\mathbf{H}^{(l)} = f_{\boldsymbol{\theta}}(\bar{\mathbf{X}}^{(l)}) = \mathrm{ReLU}(\bar{\mathbf{X}}^{(l)}\boldsymbol{\Theta}), \quad \bar{\mathbf{X}}^{(l)} = \mathbf{T}^{l}\mathbf{X}, \tag{4.2}$$

where $\mathbf{T} \in \mathbb{R}^{N\times N}$ is a generalized transition matrix and we take $\mathbf{T} = \tilde{\mathbf{A}}_{sym} =$

$\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$ in this work. $\mathbf{H}^{(1)}$ is learned from a local view as the message-passing is only enabled between direct neighbors, while $\{\mathbf{H}^{(l)}\}_{l=2}^{L}$ are learned from a set of high-order views that encode the long-range node interactions at different scales.

It is noteworthy that our model inherently separates the feature propagation step, i.e., $\bar{\mathbf{X}}^{(l)} = \mathbf{T}^{l}\mathbf{X}$, and transformation step, i.e., $f_{\boldsymbol{\theta}}(\bar{\mathbf{X}}^{(l)})$ into the data augmentation and representation learning modules, respectively. Compared to standard GNNs that couple the two steps together in each layer, this decoupling strategy allows the model to perform the high-order feature propagation without conducting non-linear transformations, reducing the risk of over-smoothing (Feng et al. 2020; Ding, Wang, et al. 2022) in contrastive learning. In the meantime, we can use a much simpler encoding backbone to transform the augmented features to node representations without stacking multiple GNN layers.

Structural Contrastive Objective. In our structural contrastive learning, we aim to maximize the agreement between the representations of each node learned from the local view and its different high-order views by maximizing their mutual information. Instead of directly contrasting the output of the encoder network, we follow previous research in contrastive learning (T. Chen et al. 2020) and apply a *projection head* $g_{\boldsymbol{\psi}}(\cdot)$ to the node representations computed by the encoder network. As such, the representations we contrast in our structural contrastive learning can be denoted by $\{\mathbf{U}^{(l)}\}_{l=1}^{L}$, where $\mathbf{U}^{(l)} = g_{\boldsymbol{\psi}}(\mathbf{H}^{(l)})$, and $g_{\boldsymbol{\psi}}(\cdot)$ is a two-layer MLP in our implementation.

In our work, we adopt InfoNCE (Oord, Li, and Vinyals 2018) to estimate the lower bound of the mutual information between the node representations learned from a local view $\mathbf{U}^{(1)}$ and different high-order views $\{\mathbf{U}^{(l)}\}_{l=2}^{L}$ of the input graph. The loss function of structural contrastive learning can be defined as:

$$\mathcal{L}_{str} = -\sum_{i=1}^{N}\sum_{l=2}^{L}\log\frac{\exp(\mathbf{u}_i^{(1)}\cdot\mathbf{u}_i^{(l)}/\tau_1)}{\sum_{j=1}^{M+L-1}\exp(\mathbf{u}_i^{(1)}\cdot\mathbf{u}_j^{(l)}/\tau_1)}, \tag{4.3}$$

where τ_1 is the temperature parameter. Note that $\{\mathbf{u}_j^{(l)}\}_{j=1}^{M+L-1}$ contains $L-1$ positive examples and M negative examples sampled from augmented views of other nodes.

By performing the proposed structural contrastive learning based on multi-scale augmentations of the input graph, the encoder network $f_{\boldsymbol{\theta}}(\cdot)$ not only encourages accurate node-wise discrimination but also captures multi-scale global structural knowledge during the learning process. The resulted node representations $\mathbf{H}$ can be computed by feeding the mixed-order propagated features $\bar{\mathbf{X}}$ to the encoder network as:

$$\mathbf{H} = f_{\boldsymbol{\theta}}(\bar{\mathbf{X}}) = \mathrm{ReLU}(\bar{\mathbf{X}}\boldsymbol{\Theta}), \quad \bar{\mathbf{X}} = \frac{1}{L}\sum_{l=0}^{L}\mathbf{T}^l\mathbf{X}. \tag{4.4}$$

This enables the learned node representations to preserve both local and global structure information compared with directly using $\mathbf{T}^L\mathbf{X}$ (K. Xu et al. 2018; Feng et al. 2020).

4.1.3.2 Semantic Contrastive Learning

Despite the structural contrastive learning algorithm can provide better node-wise discrimination by exploiting the global structural knowledge based on the multi-scale propagated features, it has the same limitation as existing GCL efforts – cannot explicitly encode the semantic structure of the input graph. To further capture the semantic global knowledge, we propose a *semantic contrastive learning* algorithm that encourages the intra-cluster compactness and inter-cluster separability in the semantic latent space.

Since the prior knowledge of node clusters is unknown, we propose to iteratively infer the clusters among nodes and the corresponding prototypes based on the learned node representations, and perform semantic contrastive learning to promote those nodes that are semantically similar clustering around their corresponding cluster prototypes.

We denote the cluster prototype representation via a matrix $\mathbf{C} \in \mathbb{R}^{K \times D'}$, where K is the number of prototypes inferred from the data. We use $\mathbf{c}_k$ to denote the k-th row of $\mathbf{C}$, which is the representation of the k-th prototype in the latent space. The prototype assignments or pseudo labels of nodes are denoted by $\mathcal{Z} = \{z_i\}_{i=1}^{n}$, where $z_i \in \{1, ..., K\}$ is the pseudo label of node v_i.

Bayesian Non-parametric Prototype Inference. A key function of our semantic contrastive learning algorithm is to infer highly representative cluster prototypes. However, the optimal number of clusters is unknown under the setting of unsupervised node representation learning. To bridge the gap, we propose a Bayesian non-parametric prototype inference algorithm to approximate the optimal number of clusters and simultaneously compute the cluster prototypes. Specifically, we build a Dirichlet Process Mixture Model (DPMM) and assume the distribution of node representations is a mixture of Gaussians, in which each component is used to model the prototype of a cluster. Note that the components share the same fixed covariance matrix $\sigma\mathbf{I}$. The DPMM model is defined as:

$$G \sim \mathrm{DP}(G_0, \alpha), \quad \boldsymbol{\phi}_i \sim G, \quad \mathbf{h}_i \sim \mathcal{N}(\boldsymbol{\phi}_i, \sigma\mathbf{I}), \tag{4.5}$$

where G is a Gaussian distribution drawn from the Dirichlet process $\mathrm{DP}(G_0, \alpha)$, and α is the concentration parameter for $\mathrm{DP}(G_0, \alpha)$. $\boldsymbol{\phi}_i$ is the mean of the Gaussian sampled for node representation $\mathbf{h}_i$. G_0 is the prior over means of the Gaussians. We take G_0 to be a zero-mean Gaussian $\mathcal{N}(\mathbf{0}, \rho\mathbf{I})$, where $\rho\mathbf{I}$ is the covariance matrix.

Next, we use a collapsed Gibbs sampler (Resnik and Hardisty 2010) to infer the Gaussian components. The Gibbs sampler iteratively samples pseudo labels for the nodes given the means of the Gaussian components and samples the means of the

Gaussian components given the pseudo labels of the nodes. Following (Kulis and Jordan 2011), such a process is almost equivalent to K-Means when the variance of the Gaussian components $\sigma \to 0$. The almost zero variance eliminates the need to estimate the variance σ, thus making the inference efficient. Let $\tilde{K}$ denote the number of inferred prototypes at the current iteration step, the prototype assignment update can be formulated as:

$$
\begin{aligned}
z_i &= \arg\min_k \{d_{ik}\}, \\
d_{ik} &= \begin{cases} ||\mathbf{h}_i - \mathbf{c}_k||^2 & \text{for } k = 1, ..., \tilde{K} \\ \xi & \text{for } k = \tilde{K} + 1, \end{cases}
\end{aligned}
\tag{4.6}
$$

where d_{ik} is the distance to determine the pseudo labels of node representation $\mathbf{h}_i$. ξ is the margin to initialize a new prototype.

With the formulation in Equation (4.6), a node will be assigned to the prototype modeled by the Gaussian component corresponding to the closest mean of Gaussian, unless the squared Euclidean distance to the closest mean is greater than ξ. In this case, we initialize a new prototype with such node representation. After obtaining the pseudo labels, the cluster prototype representations can be computed by: $\mathbf{c}_k = \sum_{z_i=k} \mathbf{h}_i / \sum_{z_i=k} 1$, for $k = 1, ..., \tilde{K}$.

Note that we iteratively update prototype assignments and prototype representations till convergence, and we set the number of prototypes K to be the number of inferred prototypes $\tilde{K}$. Afterward, we refine the cluster prototypes using label propagation, and we attach the details in Supplementary **??** due to the space limit.

Semantic Contrastive Objective. After obtaining the prototype assignments $\mathcal{Z}$ and prototype representations $\mathbf{C}$, our semantic contrastive objective aims to consolidate the semantic structure (i.e., intra-cluster compactness and inter-cluster separability) of the learned node representation $\mathbf{H}$ by updating the encoder parameter $\boldsymbol{\theta}$. To this end, we maximize the likelihood of each node in the graph given $\boldsymbol{\theta}$ and $\mathbf{C}$:

$$\begin{aligned} Q(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \log p(\mathbf{h}_i|\boldsymbol{\theta}, \mathbf{C}) \\ &= \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{h}_i, k|\boldsymbol{\theta}, \mathbf{C}), \end{aligned} \tag{4.7}$$

where p is the probability density function. Directly optimizing log-likelihood $Q(\boldsymbol{\theta})$ is intractable as the labels of nodes are unknown. Instead, we optimize the variational lower bound of $Q(\boldsymbol{\theta})$, given by:

$$\begin{aligned} Q(\boldsymbol{\theta}) \geq & \sum_{i=1}^{N} \sum_{k=1}^{K} p(k|\mathbf{h}_i) \log \frac{p(\mathbf{h}_i, k|\boldsymbol{\theta}, \mathbf{C})}{p(k|\mathbf{h}_i)} \\ = & \sum_{i=1}^{N} \sum_{k=1}^{K} p(k|\mathbf{h}_i) \log p(\mathbf{h}_i, k|\boldsymbol{\theta}, \mathbf{C}) \\ & - \sum_{i=1}^{N} \sum_{k=1}^{K} p(k|\mathbf{h}_i) \log p(k|\mathbf{h}_i). \end{aligned} \tag{4.8}$$

Note that we can drop the second term of the right-hand side of Equation (4.8) as it is a constant. To maximize the remaining part, we can estimate $p(k|\mathbf{h}_i)$ by $p(k|\mathbf{h}_i, \boldsymbol{\theta}, \mathbf{C}) = \mathbb{1}_{\{k=z_i\}}$, as we assign $\mathbf{h}_i$ to cluster z_i given $\mathbf{C}$ in our DPMM model. Thus, we can maximize $Q(\boldsymbol{\theta})$ by minimizing the following loss function:

$$\mathcal{L}_{sem} = - \sum_{i=1}^{N} \log p(\mathbf{h}_i, z_i|\boldsymbol{\theta}, \mathbf{C}). \tag{4.9}$$

Under the assumption of a uniform prior distribution of node representation, we have $p(\mathbf{h}_i, z_i|\boldsymbol{\theta}, \mathbf{C}) \propto p(z_i|\mathbf{h}_i, \boldsymbol{\theta}, \mathbf{C})$. Since the distribution of node representation around each prototype generated by the DPMM is an isotropic Gaussian, we can estimate $p(k|\mathbf{h}_i, \boldsymbol{\theta}, \mathbf{C})$ by $\exp(||\mathbf{h}_i - \mathbf{c}_{z_i}||^2/\sigma^2) / \sum_{i=1}^{K} \exp(||\mathbf{h}_i - \mathbf{c}_k||^2/\sigma^2)$. After applying ℓ_2 normalization on the representation of nodes and prototypes, we can estimate $p(z_i|\mathbf{h}_i, \boldsymbol{\theta}, \mathbf{C})$ by:

$$p(z_i|\mathbf{h}_i, \boldsymbol{\theta}, \mathbf{C}) = \frac{\exp(\mathbf{h}_i \cdot \mathbf{c}_{z_i}/\tau_2)}{\sum_{k=1}^{K} \exp(\mathbf{h}_i \cdot \mathbf{c}_k/\tau_2)}, \tag{4.10}$$

where $\mathbf{c}_{z_i}$ is the representations of z_i-th prototype. The temperature parameter $\tau_2 \propto \sigma^2$ is related to the concentration of node representation around each prototype, and σ is the variance of the Gaussians in the DPMM model defined by Equation (4.5). For the simplicity of training, we directly take τ_2 as a hyperparameter. Taking Equation (4.10) into Equation (4.9), we can maximize $Q(\boldsymbol{\theta})$ by minimizing the following loss function:

$$\mathcal{L}_{sem} = -\sum_{i=1}^{N} \log \frac{\exp(\mathbf{h}_i \cdot \mathbf{c}_{z_i}/\tau_2)}{\sum_{k=1}^{K} \exp(\mathbf{h}_i \cdot \mathbf{c}_k/\tau_2)}. \tag{4.11}$$

4.1.3.3 Model Learning

Given the proposed S^3-CL learning framework, our goal is to learn expressive node representations that preserve both valuable structural and semantic knowledge without any semantic labels. In this section, we will introduce the overall loss function, and also the optimization of the proposed framework with regard to the network parameters, prototype assignments, and prototype representations.

Overall Loss. To train our model in an end-to-end fashion and learn the encoder $f_{\boldsymbol{\theta}}(\cdot)$, we jointly optimize both the structural and semantic contrastive learning losses. The overall objective function is defined as:

$$\mathcal{L} = \gamma \mathcal{L}_{str} + (1-\gamma)\mathcal{L}_{sem}, \tag{4.12}$$

where we aim to minimize $\mathcal{L}$ during training, and γ is a balancing parameter to control the contribution of each contrastive learning loss. For the sake of the stability of the training of the encoder, we apply our Bayesian non-parametric prototype inference algorithm on the node representations computed by a momentum encoder (He et al. 2020).

Notably, in semantic contrastive learning, the computed pseudo labels $\mathcal{Z}$ can be utilized in the negative example sampling process in our structural contrastive learning to avoid sampling bias issues (Chuang et al. 2020). We select negative samples in Equation (4.3) for each node from nodes assigned to different prototypes.

Model Optimization via EM. Specifically, we adopt EM algorithm to alternately estimate the posterior distribution $p(z_i|\mathbf{x}_i, \boldsymbol{\theta}, \mathbf{C})$ and optimize the network parameters $\boldsymbol{\theta}$. We describe the details for the E-step and M-step applied in our methods as follows:

- **E-step.** In this step, we fix the network parameter $\boldsymbol{\theta}$, and estimate the prototypes $\mathbf{C}$ and the prototype assignment $\mathscr{Z}$ with our proposed Bayesian non-parametric prototype inference algorithm.
- **M-step.** Given the posterior distribution computed by the E-step, we aim to maximize the expectation of log-likelihood $Q(\boldsymbol{\theta})$, by directly optimizing the semantic contrastive loss function $\mathcal{L}_{sem}$. In order to perform structural and semantic contrastive learning at the same time, we instead optimize a joint overall loss function as formulated in Equation (4.12).

Note that after the self-supervised pre-training is done, the pre-trained encoder can be directly used to generate node representations for various downstream tasks.

4.1.4 Experiments

4.1.4.1 Experimental Settings

Evaluation Datasets. In our experiments, we evaluate S^3-CL on six public benchmark datasets that are widely used for node representation learning, including Cora (Sen et al. 2008b), Citeseer (Sen et al. 2008b), Pubmed (Namata et al. 2012), Amazon-P (Shchur et al. 2018), Coauthor CS (Shchur et al. 2018) and ogbn-arxiv (W. Hu et al. 2020). Cora, Citeseer, and Pubmed are the three most widely used citation networks. Amazon-P is a co-purchase graph and Coauthor CS is a co-authorship graph.

The ogbn-arxiv is a large-scale citation graph benchmark dataset.

Compared Methods. To demonstrate the effectiveness of our proposed method, six state-of-the-art graph self-supervised learning methods are compared in our experiments, including DGI (Veličković et al. 2019), MVGRL (Hassani and Khasahmadi 2020), GMI (Peng et al. 2020), GRACE (Zhu et al. 2020a), MERIT (Jin et al. 2021), and SUGRL (Mo et al. 2022). As we consider node classification as our downstream task, we also include five representative supervised node classification methods, namely MLP (Veličković et al. 2019), LP (Zhu, Ghahramani, and Lafferty 2003), GCN (Thomas N. Kipf and Welling 2017b), GAT (Veličković, Cucurull, Casanova, Romero, Liò, et al. 2018), and SGC (**sgc**), as baselines for the evaluation on the node classification task. To evaluate the model performance for node clustering, we compare S^3-CL against methods including K-Means (Lloyd 1982), GAE (Thomas N Kipf and Welling 2017a), adversarially regularized GAE (ARGA) and VGAE (ARVGA) (Pan et al. 2018), GALA (J. Park et al. 2019), DGI, DBGAN (Zheng et al. 2020), MVGRL, MERIT, and SUGRL.

4.1.4.2 General Comparisons

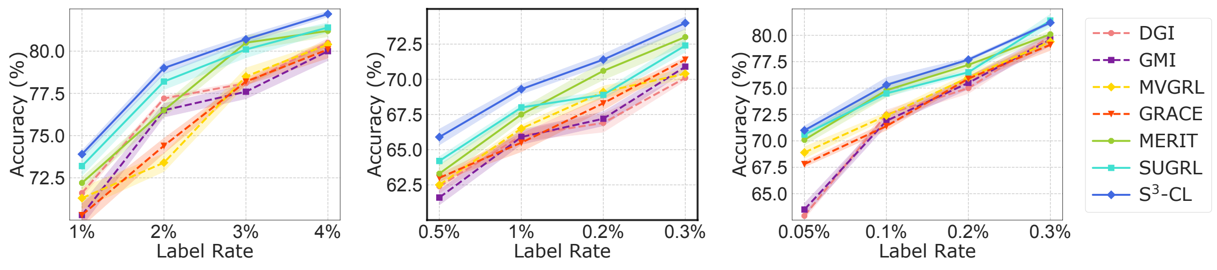
Node Classification. To evaluate the trained encoder network, we adopt a linear evaluation protocol by training a separate logistic regression classifier on top of the learned node representations. We follow the evaluation protocols in previous

Table 17. Node Classification Performance Comparison on Benchmark Datasets.

Methods	Cora	Citeseer	Pubmed	Amazon-P	Coauthor CS	ogbn-arxiv
			SUPERVISED			
MLP	55.2 ± 0.4	46.5 ± 0.5	71.4 ± 0.3	78.5 ± 0.2	76.5 ± 0.3	55.5 ± 0.2
LP	68.0 ± 0.5	45.3 ± 0.6	63.0 ± 0.3	75.4 ± 0.0	74.3 ± 0.0	68.3 ± 0.0
GCN	81.7 ± 0.4	70.5 ± 0.3	79.4 ± 0.4	87.3 ± 1.0	91.8 ± 0.1	71.7 ± 0.3
GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3	86.2 ± 1.5	90.5 ± 0.7	$\mathbf{73.2 \pm 0.2}$
SGC	81.5 ± 0.2	73.1 ± 0.1	79.7 ± 0.4	88.3 ± 1.1	91.5 ± 0.3	69.8 ± 0.2
			SELF-SUPERVISED + FINE-TUNING			
DGI	81.7 ± 0.6	71.5 ± 0.7	77.3 ± 0.6	83.1 ± 0.3	90.0 ± 0.3	67.1 ± 0.4
GMI	82.7 ± 0.2	73.0 ± 0.3	80.1 ± 0.2	85.1 ± 0.0	91.0 ± 0.0	69.6 ± 0.3
MVGRL	82.9 ± 0.7	72.6 ± 0.7	79.4 ± 0.3	87.3 ± 0.1	91.3 ± 0.1	71.3 ± 0.2
GRACE	80.0 ± 0.4	71.7 ± 0.6	79.5 ± 1.1	81.8 ± 0.8	90.1 ± 0.8	71.1 ± 0.2
MERIT	83.1 ± 0.6	$\underline{74.0 \pm 0.7}$	80.1 ± 0.4	$\underline{88.8 \pm 0.4}$	$\underline{92.4 \pm 0.4}$	71.7 ± 0.1
SUGRL	$\underline{83.4 \pm 0.5}$	73.0 ± 0.5	$\mathbf{81.9 \pm 0.5}$	88.5 ± 0.2	92.2 ± 0.5	69.3 ± 0.2
S^3-CL (ours)	$\mathbf{84.5 \pm 0.4}$	$\mathbf{74.6 \pm 0.4}$	$\underline{80.8 \pm 0.3}$	$\mathbf{89.0 \pm 0.5}$	$\mathbf{93.1 \pm 0.4}$	$\underline{72.8 \pm 0.3}$

Table 18. Node Clustering Performance Comparison on Benchmark Datasets.

Methods	Cora			Citeseer			Pubmed		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-Means	49.2	32.1	22.9	54.0	30.5	27.8	59.5	31.5	28.1
GAE	59.6	42.9	34.7	40.8	17.6	12.4	67.2	27.7	27.9
ARGA	64.0	44.9	35.2	57.3	35.0	34.1	66.8	30.5	29.5
ARVGA	64.0	45.0	37.4	54.4	26.1	24.5	69.0	29.0	30.6
GALA	74.5	57.6	53.1	69.3	44.1	44.6	69.3	32.7	32.1
DGI	55.4	41.1	32.7	51.4	31.5	32.6	58.9	27.7	31.5
DBGAN	$\underline{74.8}$	56.0	$\underline{54.0}$	67.0	40.7	41.4	69.4	32.4	32.7
MVGRL	73.2	56.2	51.9	68.1	43.2	43.4	69.3	34.4	32.3
MERIT	73.6	57.1	52.8	68.9	43.9	44.1	$\underline{69.5}$	34.7	32.8
SUGRL	73.9	$\underline{58.5}$	53.0	$\underline{70.5}$	$\underline{45.8}$	$\underline{47.0}$	$\underline{69.5}$	$\underline{35.0}$	$\underline{33.4}$
S^3-CL (ours)	**75.1**	**60.7**	**56.6**	**71.2**	**46.3**	**48.5**	**71.3**	**36.0**	**34.7**



(a) Cora (b) Citeseer (c) Pubmed

Figure 16. Node Classification Results with Limited Training Labels.

works (Veličković et al. 2019; W. Hu et al. 2020) for node classification. The mean classification accuracy with standard deviation on the test nodes after 10 runs of training is reported. To avoid the out-of-memory issue when evaluating MVGRL, GRACE, and MERIT on the ogbn-arxiv dataset, we subsample 512 nodes as negative samples for each node during the self-supervised learning phase.

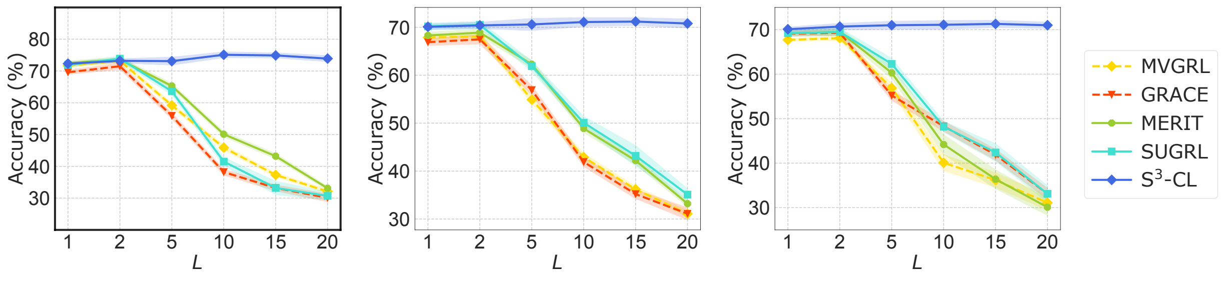
The node classification results of different methods are reported in Table 17. We can clearly see that S^3-CL outperforms the state-of-the-art self-supervised node representation learning methods across the five public benchmarks. Such superiority mainly stems from two factors: (i) our approach S^3-CL grants each node access to information of nodes in a larger neighborhood; (ii) S^3-CL infers the semantic information of nodes, and enforces intra-cluster compactness and inter-cluster separability on the node representation. With the help of this extra information, node representations generated by S^3-CL are more informative and distinctive. Without access to labels, S^3-CL even outperforms supervised methods like SGC and GAT.

Node Clustering. To evaluate the quality of the node representations learned by different methods, we conduct experiments on node clustering. We follow the same evaluation protocol as in (Hassani and Khasahmadi 2020). K-Means is applied on the learned node representation to get clustering results. We use accuracy (ACC), normalized mutual information (NMI), and adjusted rand index (ARI) to measure the performance of clustering. We report the averaged clustering results over 10 times of execution.

The clustering results are displayed in Table 18. It is observed that our approach achieves remarkable performance gain over compared methods. For example, the NMI on Cora is improved by 2.2% against the previous SOTA method SUGRL. Such improvement greatly attributes to the fact that S^3-CL explores the semantic information of nodes instead of enforcing node-wise discrimination alone as other GCL methods. Thus, the node representation learned by S^3-CL works well for clustering algorithms.

Node Classification with Few Labels. We further evaluate the impact of label rate on the downstream node classification task. Specifically, we evaluate all self-supervised learning methods from Table 17 under different low-resource settings. The results in Figure 16 show that our proposed framework S^3-CL can still outperform existing methods given a lower label rate. It validates that the node representations learned by our approach S^3-CL can encode valuable structural and semantic knowledge from the input graph. As a result, the node representations can be effectively used for the node classification task even with an extremely small label ratio.

Effect of Feature Propagation. Next, we investigate the effect of multi-scale feature propagation in the structural contrastive learning by altering the propagation steps L. A larger L allows message-passing within a larger neighborhood for learning the node representations. To demonstrate the power of our approach in utilizing structural global knowledge, we compare S^3-CL against GRACE, MVGRL, MERIT, and SUGRL with different numbers of layers L. The node clustering accuracy of different methods is shown in Figure 17. By increasing the propagation steps (number



(a) Cora (b) Citeseer (c) Pubmed

Figure 17. Node Clustering Results of GCL Methods with Various Propagation Steps (L).

of layers), we can clearly observe that existing unsupervised GCL methods severely degrade due to the oversmoothing issue. In contrast, S^3-CL consistently achieves improved performance by making use of information in a larger neighborhood for node representation learning.

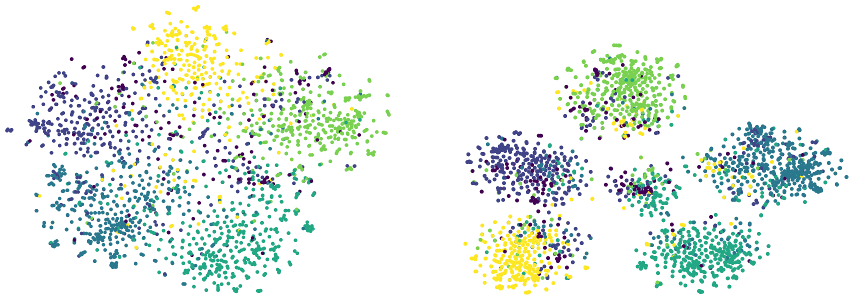
Ablation Study. To validate the effectiveness of the structural contrastive learning and semantic contrastive learning in S^3-CL, we conduct an ablation study on Citesser, Cora, and Pubmed with two variants of S^3-CL, each of which has one of the contrastive learning components removed. The node classification results are shown in Table 19. We can observe that the performance of S^3-CL degrades when any of the components are removed. Our S^3-CL using all components achieves the best performance as the structural and semantic contrastive components complement each other. Hence, the effectiveness of each component is verified.

Table 19. Ablation Study on Contrastive Components.

Method	Citeseer	Cora	Pubmed
w/o structural	73.1±0.2	83.3±0.3	80.0± 0.3
w/o semantic	71.9±0.4	82.2±0.5	79.3± 0.2
S^3-CL	**74.6±0.4**	**84.5±0.4**	**80.8±0.3**

4.1.4.3 Representation Visualization.

To visually show the superior quality of the node representations learned by S^3-CL, we use t-SNE to visualize and compare the learned node representations between S^3-CL and the best-performing baseline on Citeseer, i.e., SUGRL. The visualization results are shown in Figure 18, where each dot represents the representation of a node, and the color of the dot denotes its ground-truth label. From the figure, we can observe that though some classes can be identified by SUGRL, the boundaries between different classes are unclear. Our proposed model is able to enforce better intra-cluster compactness and inter-cluster separability.



(a) SUGRL (b) S^3-CL

Figure 18. Representation Visualization on the Citeseer Dataset.

4.2 Self-supervised Graph Anomaly Detection

4.2.1 Introduction

Attributed networks provide a potent tool to handle the data heterogeneity that we are often confronted with in vast amounts of information systems. Apart from traditional plain networks in which only node-to-node interactions are observed, attributed networks also encode a rich set of features for each node (Akoglu et al. 2012; Huang, Li, and Hu 2017; Li, Dani, Hu, Tang, et al. 2017). They are increasingly used to model a wide range of complex systems, such as social media networks, critical infrastructure networks, and gene regulatory networks (Akoglu et al. 2012; Pfeiffer III et al. 2014). For example, in social networks, users not only are connected with each other by performing various social activities but also are affiliated with rich profile information; in critical infrastructure networks, different power stations form grids, and are also associated with additional attribute information (e.g., electricity capacity); in gene regulatory networks, genes interact with each other to control specific cell functions in addition to the rich gene sequence expressions. Studies from social science have shown that data often exhibits correlation among the attributes of connected individuals (McPherson, Smith-Lovin, and Cook 2001; Shalizi and Thomas 2011), and such insights are helpful in distilling actionable knowledge from such networks.

Detecting anomalies from data (e.g., attribute-value data, networked data) is a vital research problem of pressing societal concerns, with significant implications in many security-related applications, ranging from social spam detection, financial fraud detection to network intrusion detection (Akoglu, Tong, and Koutra 2015). Due to the strong modeling power of attributed networks in unifying information of different modalities, there is a surge of research interests in detecting anomalous nodes whose patterns deviate significantly from other majority nodes on attributed networks. Generally, the abnormality of nodes on attributed networks is not only determined by their mutual interactions with others (w.r.t.*topological structure*), but also is measured by their content dissonance (w.r.t. *nodal attributes*).

Due to the prohibitive cost for accessing the ground truth anomalies, existing efforts are mostly unsupervised. Among them, one family of methods study the problem at the *mesoscope* with ego-network (Perozzi and Akoglu 2016) or community analysis (Gao et al. 2010) and then identify anomalies by measuring the abnormality of ego-networks or comparing the current node with other nodes within the same community. Another family of methods heavily rely on subspace selection and attempt to find anomalies in a node feature subspace (Sánchez et al. 2013; Sánchez et al. 2014; Müller et al. 2013; Perozzi et al. 2014). Recently, residual analysis has emerged as another way to find anomalous nodes (Li, Dani, Hu, and Liu 2017; Peng et al. 2018), where anomalies are defined as the nodes that cannot be approximated from others. Despite their empirical success, the following challenges remain for anomaly detection on attributed networks: (1) *Network sparsity* - the network structure could be very sparse on real-world attributed networks; thus ego-network or community analysis is difficult to perform as they highly depend on the observed node interactions. (2) *Data nonlinearity* - the node interactions and nodal attributes are highly non-linear in nature while existing subspace selection based anomaly detectors mainly model the attributed networks with linear mechanisms. (3) *Complex modality interactions* - attributed networks are notoriously difficult to tackle due to the bewildering combination of two information sources, which necessitates an unified feature space to capture their complex interactions for anomaly detection.

To address the challenges above, we propose to model the attributed networks with graph convolutional network (GCN) (Thomas N. Kipf and Welling 2017b). GCN, which takes the topological structure and nodal attributes as input, is able to learn discriminative node embeddings by stacking multiple layers of linear units and non-linear activation functions. Even though GCN emerges to be a principled tool to model attributed networks and achieves the state-of-the-art performance in the semi-supervised node classification task, it remains unclear how its power can be shifted to the anomaly detection problem. To bridge the gap, we propose a novel graph convolutional autoencoder framework called DOMINANT (Deep Anomaly Detection on Attributed Networks) to support anomaly detection on attributed networks. Specifically, DOMINANT first compresses the input attributed network to succinct low-dimensional embedding representations using graph convolutional network as an encoder function; then we aim to reconstruct both the topological structure and nodal attributes with corresponding decoder functions. The reconstruction errors of nodes following the encoder and decoder phases are then leveraged for spotting anomalous nodes on attributed networks. The main contributions of this work are as follow:

- We systematically analyze the limitations of existing shallow anomaly detection methods and show the significance of developing a novel deep architectured anomaly detector on attributed networks.
- We develop a principled graph convolutional autoencoder DOMINANT which seamlessly models the attributed network and conducts anomaly detection in a joint framework. In particular, the proposed model can spot anomalies by analyzing the reconstruction errors of nodes from both the structure and the attribute perspectives.
- We evaluate our proposed model on various attributed networks from different domains. Empirical experimental results demonstrate the superior performance of our proposed framework.

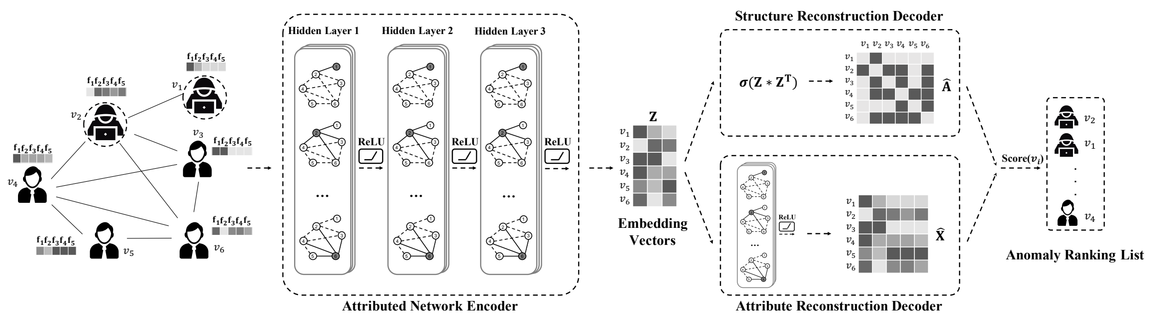

Hidden Layer 1
Hidden Layer 2
Hidden Layer 3
ReLU
Attributed Network Encoder
Z
Embedding Vectors
Structure Reconstruction Decoder
$\sigma(\mathbf{Z} * \mathbf{Z}^{\mathrm{T}})$
$\hat{\mathbf{A}}$
$\hat{\mathbf{X}}$
Attribute Reconstruction Decoder
Score(v_i)
Anomaly Ranking List

Figure 19. The Overall Framework of Our Proposed DOMINANT for Deep Anomaly Detection on Attributed Networks.

4.2.2 Problem Definition

The notations and definition for attributed networks are explained in Section 2.1.3. To make the results more interpretable, we formulate the task of anomaly detection on attributed networks as a ranking problem:

Problem Definition 4 ***Anomaly Ranking on Attributed Networks****: Given an attributed network $\mathcal{G}$, with the adjacency matrix* $\mathbf{A}$ *and attribute information matrix* $\mathbf{X}$ *of n node instances, the task is to rank all the nodes according to the degree of abnormality, such that the nodes that differ singularly from the majority reference nodes should be ranked on high positions.*

4.2.3 Architecture Overview

In this section, we present the proposed framework of DOMINANT in detail. The architecture of the deep model is illustrated in Figure 19. As can be observed, the fundamental building block of DOMINANT is the deep autoencoder (Goodfellow et al. 2016) and it consists of three essential components: (i) *attributed network encoder* - which models network structure and nodal attributes seamlessly in a joint framework for node embedding representation learning with GCN; (ii) *structure reconstruction decoder* - which aims to reconstruct the original network topology with the learned node embeddings; and (iii) *attribute reconstruction decoder* - which attempts to reconstruct the observed nodal attributes with the obtained node embeddings. Afterwards, the reconstruction errors of nodes are then leveraged to flag anomalies on attributed networks.

4.2.3.1 Preliminary - Deep Autoencoder

As suggested by (Tong and Lin 2011; Zhou and Paffenroth 2017; Li, Dani, Hu, and Liu 2017), the disparity between the original data and the estimated data (i.e., reconstruction errors) is a strong indicator to show the abnormality of instances in a dataset. Specifically, the data instances with large reconstruction errors are more likely to be considered as anomalies, since their patterns deviate significantly from the majority and cannot be accurately reconstructed from the observed data. Among various reconstruction based anomaly detection methods, deep autoencoder achieves state-of-the-art performance. Deep autoencoder is a type of deep neural network that is used to learn latent representations of data in an unsupervised manner by stacking multiple layers of encoding and decoding functions together. It has achieved promising learning performance in various domains, such as computer vision, natural language processing, and speech recognition (Goodfellow et al. 2016).

Given an input dataset $\mathbf{X}$, the encoder $Enc(\cdot)$ is first applied to map the data into a latent low-dimensional feature space, and then the decoder $Dec(\cdot)$ tries to recover the original data based on the latent representations. The learning process can be described as minimizing a cost function described below:

$$\min \mathbb{E}[dist(\mathbf{X}, Dec(Enc(\mathbf{X}))], \tag{4.13}$$

where $dist(\cdot, \cdot)$ is a predefined distance metric. In practice, we often choose the ℓ_2-norm distance to measure the reconstruction errors. It also should be noted that deep autoencoder is able to capture the highly non-linear information from high-dimensional input by applying multiple layers of linear units and nonlinear activation functions in the encoder and decoder phases, which is advantageous compared to conventional shallow learning models. Subsequently, in this study, we propose to solve the problem of anomaly detection on attributed networks in a deep autoencoder architecture.

4.2.3.2 Attributed Network Encoder

As a rich network representation, attributed networks encode not only the network structure but also abundant nodal attributes. However, conventional deep autoencoders can only handle *i.i.d.* attribute-value data (Zhou and Paffenroth 2017; W. Yu et al. 2018), which cannot be directly applied to our scenario. How to design an effective encoder to capture the underlying properties of attributed networks remains a daunting task as we need to address the three challenges (i.e., *network sparsity*, *data nonlinearity*, and *complex modality interactions*) simultaneously. To this end, we propose a new type of attributed network encoder inspired by the graph convolutional network (GCN) model (Thomas N. Kipf and Welling 2017b). Specifically, GCN considers the high-order node proximity when learning the embedding representations, thus it mitigates the network sparsity issue beyond the observed links among nodes. Meanwhile, through multiple layers of nonlinear transformations, it captures the nonlinearity of data and the complex interactions of two information modalities on attributed networks.

Mathematically, GCN extends the operation of convolution to networked data in the spectral domain and learns a layer-wise new latent representation by a spectral convolution function:

$$\mathbf{H}^{(l+1)} = f(\mathbf{H}^{(l)}, \mathbf{A}|\mathbf{W}^{(l)}), \tag{4.14}$$

where $\mathbf{H}^{(l)}$ is the input for the convolution layer l, and $\mathbf{H}^{(l+1)}$ is the output after the convolution layer. We take the attribute matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ as the input of first layer, which is equivalent to $\mathbf{H}^{(0)}$. $\mathbf{W}^{(l)}$ is a layer-specific trainable weight matrix we need to learn in the neural network. Each layer of the graph convolutional network can be expressed with the function $f(\mathbf{H}^{(l)}, \mathbf{A}|\mathbf{W}^{(l)})$ as follows:

$$f(\mathbf{H}^{(l)}, \mathbf{A}|\mathbf{W}^{(l)}) = \sigma(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{4.15}$$

where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\widetilde{\mathbf{D}}$ is the diagonal matrix of $\widetilde{\mathbf{A}}$ with the diagonal element as $\widetilde{\mathbf{D}}_{i,i} = \sum_j \widetilde{\mathbf{A}}_{i,j}$. Thus we can directly compute $\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}$ as a pre-processing step. Note that $\sigma(\cdot)$ is a non-linear activation function, such as $Relu(x) = max(0, x)$. It is worth noting that the filter or feature map parameters $\mathbf{W}^l$ are shared for all nodes on the attributed network. Given the attribute matrix $\mathbf{X}$ as input, the k^{th}-hop neighborhood of each node can be effectively captured by successively stacking a number of k convolutional layers. Therefore, the embeddings $\mathbf{Z}$ not only encode the attribute information of each node but also involve the k^{th}-order node proximity information. In this work, we propose to use three convolutional layers for constructing the attributed network encoder, but it should be noted that more layers can also be stacked for building a deeper network. The attributed network encoder can be formulated as:

$$\mathbf{H}^{(1)} = f_{Relu}(\mathbf{X}, \mathbf{A}|\mathbf{W}^{(0)}) \tag{4.16}$$

$$\mathbf{H}^{(2)} = f_{Relu}(\mathbf{H}^{(1)}, \mathbf{A}|\mathbf{W}^{(1)}) \tag{4.17}$$

$$\mathbf{Z} = \mathbf{H}^{(3)} = f_{Relu}(\mathbf{H}^{(2)}, \mathbf{A}|\mathbf{W}^{(2)}). \tag{4.18}$$

Here, $\mathbf{W}^{(0)} \in \mathbb{R}^{n\times h_1}$ is an input-to-hidden layer with h_1 feature maps. Similarly, $\mathbf{W}^{(1)} \in \mathbb{R}^{h_1\times h_2}$ and $\mathbf{W}^{(3)} \in \mathbb{R}^{h_2\times h_3}$ are two hidden-to-hidden weight matrices. After applying three layers of convolution, the input attributed network can be transferred to the h_3-dimensional latent representations $\mathbf{Z}$, which can capture the high non-linearity in the topological network structure and nodal attributes.

4.2.3.3 Structure Reconstruction Decoder

In this subsection, we will discuss how to reconstruct the original network structure with the learned latent representations $\mathbf{Z}$, which is from the aforementioned encoder module. Let $\widehat{\mathbf{A}}$ denote the estimated adjacency matrix, then the structure reconstruction error $\mathbf{R}_S = \mathbf{A} - \widehat{\mathbf{A}}$ can be exploited to determine structural anomalies on the network. Specifically, for a certain node, if its structure information can be approximated through the structure reconstruction decoder, thus it is of low probability to be anomalous. On the opposite side, if the connectivity patterns cannot be well reconstructed, it implies that its structure information does not conform to the patterns of majority normal nodes. Therefore, a larger norm of $\mathbf{R}_S(i,:)$ indicates that the i^{th} node on the attributed network has a higher probability of being an anomaly from the network structure aspect. Specifically, the decoder takes the latent representations as input and predicts whether there is a link between each pair of two nodes:

$$p(\widehat{\mathbf{A}}_{i,j} = 1|\mathbf{z}_i, \mathbf{z}_j) = sigmoid(\mathbf{z}_i, \mathbf{z}_j^{\mathrm{T}}). \tag{4.19}$$

Accordingly, we train a link prediction layer based on the output of attributed network encoder $\mathbf{Z}$, which can be presented as follows:

$$\widehat{\mathbf{A}} = sigmoid(\mathbf{Z}\mathbf{Z}^{\mathrm{T}}). \tag{4.20}$$

4.2.3.4 Attribute Reconstruction Decoder

Similarly, to compute the reconstruction errors of nodal attributes, we propose an attribute reconstruction decoder that approximates the nodal attributes information from the encoded latent representations $\mathbf{Z}$. Specifically, the attribute reconstruction decoder leverages another graph convolutional layer to predict the original nodal attributes as follows:

$$\widehat{\mathbf{X}} = f_{Relu}(\mathbf{Z}, \mathbf{A}|\mathbf{W}^{(3)}). \tag{4.21}$$

With the computed reconstruction errors $\mathbf{R}_A = \mathbf{X} - \widehat{\mathbf{X}}$, we can spot anomalies on the attributed networks from the attribute perspective.

4.2.3.5 Anomaly Detection

Until now, we have introduced how to reconstruct the topological network structure, and nodal attributes using structure reconstruction decoder and attribute reconstruction decoder, respectively. To jointly learn the reconstruction errors, the objective function of our proposed deep graph convolutional autoencoder can be formulated as:

$$\begin{aligned}\mathcal{L} &= (1-\alpha)\mathbf{R}_S + \alpha\mathbf{R}_A \\ &= (1-\alpha)||\mathbf{A} - \widehat{\mathbf{A}}||_F^2 + \alpha||\mathbf{X} - \widehat{\mathbf{X}}||_F^2,\end{aligned} \tag{4.22}$$

where α is an important controlling parameter which balances the impacts of structure reconstruction and attribute reconstruction.

By minimizing the above objective function, our proposed deep graph convolutional autoencoder can iteratively approximate the input attributed network based on the encoded latent representations until the objective function converges. The final reconstruction errors are then employed to assess the abnormality of nodes. Note that the weight matrices of the deep graph convolutional autoencoder are trained using gradient descent on the objective function. After a certain number of iterations, we can compute the anomaly score of each node $\mathbf{v}_i$ according to:

$$score(\mathbf{v}_i) = (1-\alpha)||\mathbf{a} - \widehat{\mathbf{a}}_i||_2 + \alpha||\mathbf{x}_i - \widehat{\mathbf{x}}_i||_2. \tag{4.23}$$

Specifically, instances with larger scores are more likely to be considered as anomalies; thus we can compute the ranking of anomalies according to the corresponding anomaly scores.

4.2.4 Experiments

In this section, we perform empirical evaluations on real-world attributed networks to verify the effectiveness of the proposed Dominant framework.

Table 20. Details of the Three Attributed Network Datasets with Injected Anomalies.

	BlogCatalog	Flickr	ACM
# nodes	5,196	7,575	16,484
# edges	171,743	239,738	71,980
# attributes	8,189	12,047	8,337
# anomalies	300	450	600

4.2.4.1 Experimental Settings

In this section, we introduce the detailed experimental settings, including the compared baseline methods and evaluation metrics.

Evaluation Datasets. In order to have a comprehensive evaluation, we adopt three real-world attributed network datasets that have been widely used in previous research (Jundong Li et al. 2015; X. Huang et al. 2018; Ding, Li, and Liu 2019) in our experiments:

- **BlogCatalog**: BlogCatalog is a blog sharing website. The bloggers in blogcatalog can follow each other forming a social network. Users are associated with a list of tags to describe themselves and their blogs, which are regarded as node attributes.
- **Flickr**: Flickr is an image hosting and sharing website. Similar to BlogCatalog, users can follow each other and form a social network. Node attributes of users are defined by their specified tags that reflect their interests.
- **ACM**: ACM is another attributed network from academic field. It is a citation network where each paper is regarded as a node on the network, and the links are the citation relations among different papers. The attributes of each paper are generated from the paper abstract.

As there is no ground truth of anomalies in the above datasets, thus we need to inject anomalies into the attributed networks for our empirical evaluation. In particular, we refer to two anomaly injection methods that has been used in previous research (Ding, Li, and Liu 2019; Song et al. 2007) to generate a combined set of anomalies for each dataset by perturbing topological structure and nodal attributes, respectively. On one hand, to perturb the topological structure of an attributed network, we adopt the method introduced by (Ding, Li, and Liu 2019) to generate some small cliques. The intuition behind this method is that in many real-world scenarios, small clique is a typical anomalous substructure in which a small set of nodes are much more closely linked to each other than average (Skillicorn 2007). Therefore, after we specify the clique size as m, we randomly select m nodes from the network and then make those nodes fully connected, and then all the m nodes in the clique are regarded as anomalies. We iteratively repeat this process until a number of n cliques are generated and the total number of structral anomalies is $m \times n$. In our experiments, we fix the clique size m to 15 and set n to 10, 15 and 20 for BlogCatalog, Flickr and ACM, respectively. In addition to the injection of structural anomalies, we adopt another attribute perturbation schema introduced by (Song et al. 2007) to generate anomalies from attribute perspective. To guarantee an equal number of anomalies from structural perspective and attribute perspective will be injected into the attributed network, we first randomly select another $m \times n$ nodes as the attribute perturbation candidates. For each selected node i, we randomly pick another k nodes from the data and select the node j whose attributes deviate the most from node i among the k nodes by maximizing the Euclidean distance $||x_i - x_j||_2$. Afterwards, we then change the attributes x_i of node i to x_j. In our experiments, we set the value of k to 50. The details of these three attributed network datasets are shown in Table 20.

Compared Methods. We compare the proposed DOMINANT framework with the following popular anomaly detection methods:

- **LOF** (Breunig et al. 2000) detects anomalies at the contextual level and only considers nodal attributes.
- **SCAN** (X. Xu et al. 2007) is a structure based detection method which detects anomalies at the structural level.
- **AMEN** (Perozzi and Akoglu 2016) uses both attribute and network structure information to detect anomalous neighborhoods. Specifically, it analyzes the abnormality of each node from the ego-network point of view.
- **Radar** (Li, Dani, Hu, and Liu 2017) is the state-of-the-art unsupervised anomaly detection framework for attributed networks. It detects anomalies whose behaviors are singularly different from the majority by characterizing the residuals of attribute information and its coherence with network information.
- **ANOMALOUS** (Peng et al. 2018) performs joint anomaly detection and attribute selection to detect anomalies on attributed networks based on the CUR decomposition and residual analysis.

Evaluation Metrics In the experiments, two 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 (Li, Dani, Hu, and Liu 2017; Peng et al. 2018), 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

Table 21. Performance of different anomaly detection methods w.r.t. precision@K and recall@K.

Precision@K									
	BlogCatalog			Flickr			ACM		
K	50	100	200	50	100	200	50	100	200
LOF	0.300	0.220	0.180	0.420	0.380	0.270	0.060	0.060	0.045
Radar	0.660	0.670	0.550	0.740	0.700	0.635	0.560	0.580	0.520
ANOMALOUS	0.640	0.650	0.515	**0.780**	0.710	0.650	0.600	0.570	0.510
DOMINANT	**0.760**	**0.710**	**0.590**	0.760	**0.730**	**0.685**	**0.620**	**0.590**	**0.540**

Recall@K									
	BlogCatalog			Flickr			ACM		
K	50	100	200	50	100	200	50	100	200
LOF	0.050	0.073	0.120	0.047	0.084	0.120	0.005	0.010	0.015
Radar	0.110	0.223	0.367	0.082	0.156	0.282	0.047	0.097	0.173
ANOMALOUS	0.107	0.217	0.343	**0.087**	0.158	0.289	0.050	0.095	0.170
DOMINANT	**0.127**	**0.237**	**0.393**	0.084	**0.162**	**0.304**	**0.052**	**0.098**	**0.180**

ranked higher than a normal node. If the AUC value approaches 1, the method is of high quality.

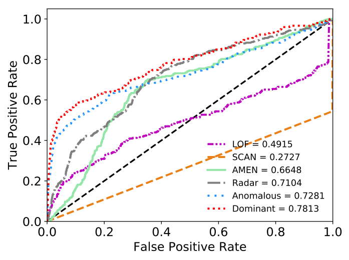
- **Precision@**K: As each anomaly detection method outputs a ranking list according to the anomalous scores of different nodes, we use Precision@K to measure the proportion of true anomalies that a specific detection method discovered in its top K ranked nodes.
- **Recall@**K: This metric measures the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

Parameter Settings In the experiments on different datasets, we propose to optimize the loss function with Adam (Kingma and Ba 2014) algorithm and train the proposed model for 300 epochs for the performance evaluation. We set the learning rate to 0.005. In addition, the attributed network encoder is built with three convolutional layers (64-neuron, 32-neuron and 16-neuron, respectively). For the other baselines, we retain to the settings described in the corresponding papers.

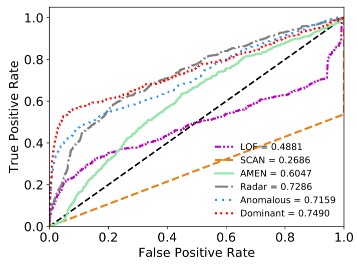
4.2.4.2 Experimental Results

In the experiments, we evaluate the performance of our proposed model DOMINANT by comparing it with the aforementioned baselines. We first present the experimental results in terms of ROC-AUC on the three datasets in Figure 20. Then we present the results w.r.t. Precision@K and Recall@K for other methods on all the attributed networks in Table 21. Note that we do not include the results of SCAN and AMEN in Table 21 as only limited number of ground truth anomalies can be detected on the top in our experiments. From the evaluation results, we make the following observations:

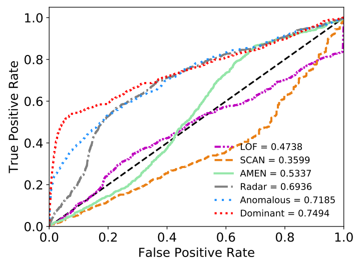
- The proposed deep model DOMINANT outperforms other baseline methods on all the three attributed networks. It verifies the effectiveness of performing anomaly detection on attributed networks by deep architecture.
- LOF and SCAN cannot achieve satisfying results in our experiments as they merely consider the nodal attributes or topological structure. Even though AMEN is designed for anomaly detection on attributed networks, it centers around finding anomalous neighborhoods rather than nodes, which also result in relatively poor performance.
- The residual analysis based models (Radar and Anomalous) are superior to the conventional anomaly detection methods (LOF, SCAN and AMEN). However, these models are still limited by their shallow mechanisms to handle the network sparsity, data nonlinearity, and complex modality interactions issues.
- DOMINANT shows a stronger ability to rank anomalies on higher positions according to the results of precision@K and recall@K. It can achieve better detection performance when the objective is to find more true anomalies within the ranking list of limited length.



(a) BlogCatalog



(b) Flickr



(c) ACM

Figure 20. ROC Curves and AUC Scores of All Methods on Different Datasets.

4.2.4.3 Parameter Analysis

Next, we investigate the impact of the controlling parameter α in our proposed DOMINANT framework and report the performance variance results in Figure 21. Here we present the AUC values on the three attributed network datasets. The controlling parameter α balances the impact of attribute reconstruction errors and structure reconstruction errors on model training and anomaly scores computation. In two extreme cases, DOMINANT will only consider the structure reconstruction errors when α is set to 1 while merely consider the attribute reconstruction errors when α is set 0. The results indicate that it is necessary to find a balance between the structure reconstruction errors and attribute reconstruction errors for achieving a better performance. The reasonable choice of α is around 0.4 to 0.7 for BlogCatalog and Flickr datasets, and 0.5 to 0.8 for ACM dataset.

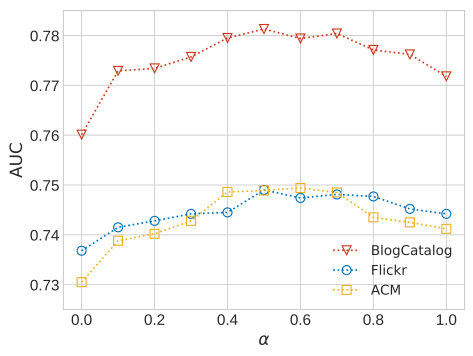


Figure 21. Impact of different α w.r.t. AUC values.

4.3 Conclusion

In this chapter, I summarize my research on self-supervised graph learning on two aspects: My first effort focus on learning generalizable node representations in a self-supervised manner. In general, the contrastive learning process in GCL is performed on top of the representations learned by a graph neural network (GNN) backbone, which transforms and propagates the node contextual information based on its local neighborhoods. However, nodes sharing similar characteristics may not always be geographically close, which poses a great challenge for unsupervised GCL efforts due to their inherent limitations in capturing such global graph knowledge. In this work, we address those inherent limitations by proposing a simple yet effective framework – *Simple Neural Networks with Structural and Semantic Contrastive Learning* (S^3-CL). Notably, by virtue of the proposed structural and semantic contrastive learning algorithms, even a simple neural network can learn expressive node representations that preserve valuable global structural and semantic patterns. Our experiments demonstrate that the node representations learned by S^3-CL achieve superior performance on different downstream tasks compared with the state-of-the-art unsupervised GCL methods.

Secondly, I discuss my work in making the first investigation on the research problem of anomaly detection on attributed networks by a self-supervised deep learning model. Specifically, we address the limitations of existing methods and model the attributed networks with graph convolutional network (GCN). As GCN handles the high-order node interactions with multiple layers of nonlinear transformations, it alleviates the network sparsity issue and can capture the nonlinearity of data as well as the complex interactions between two sources of information on attributed networks. To further enable the detection of anomalous nodes, we introduce a deep autoencoder framework to reconstruct the original attributed network with the learned node embeddings from GCN. The reconstruction errors of nodes are then employed to flag anomalies. The experimental results demonstrate the superiority of the proposed deep model over the state-of-the-art methods.

Chapter 5

CONCLUSION

This dissertation covers the topics of data-efficient graph learning through graph few-shot learning, graph weakly-supervised learning, as well as graph self-supervised learning:

In Chapter 2, we delve into the topic of graph few-shot learning. Specifically, we propose a method called Graph Prototypical Networks (GPN) for the problem of few-shot node classification. GPN is designed to handle new node classes in real-world graphs using only a small amount of labeled data. It uses graph meta-learning to extract meta-knowledge from many-shot seen node classes and apply it to few-shot unseen node classes. We conduct extensive experiments to demonstrate the superior capability of GPN in few-shot node classification. Additionally, we introduce Meta Graph Deviation Networks (Meta-GDN) for performing few-shot anomaly detection on a previously unseen graph by learning from other related graphs in the same domain. Utilizing the Graph Deviation Networks and Cross-network Meta Learning algorithm, Meta-GDN can transfer knowledge from auxiliary graphs and quickly adapt to the target graph with limited labeled anomalies. Through extensive experimental evaluations, we demonstrate the superiority of Meta-GDN over the state-of-the-art methods.

In Chapter 3, we focus on my research in graph weakly-supervised learning. Firstly, we propose a new graph meta-learning framework called Meta Label Propagation (Meta-LP), which can generate informative pseudo-labels for unlabeled nodes based on a meta-learned label propagation strategy to augment the limited labeled data. Despite its simple neural network architecture, our experiments demonstrate that Meta-LP offers easy and substantial performance gains compared to existing techniques on various benchmark datasets. Then we propose a new cross-domain anomaly detection framework COMMANDER, to address the problem of detecting anomalies in unlabeled attributed graphs. The framework includes four main components: a graph attentive encoder, anomaly classifier, domain discriminator, and attribute decoder. These components work together to bridge the gap between two attributed graphs from different domains and perform accurate anomaly detection on the target attributed graph. We conduct thorough experiments to validate the effectiveness of the proposed COMMANDER framework.

In Chapter 4, we explore the area of graph self-supervised learning. I first propose a model called S^3-CL that goes beyond existing unsupervised graph contrastive methods by capturing global graph knowledge from both structural and semantic perspectives. The results of our experiments show that the node representations learned by S^3-CL outperform state-of-the-art unsupervised GCL methods on various downstream tasks. Additionally, I present a graph generative framework called DOMINANT for the problem of graph anomaly detection, which models the input attributed graph by reconstructing both structure and attribute information from learned node representations. The idea behind DOMINANT is that anomalies usually cannot be well reconstructed, leading to higher reconstruction errors, thus the reconstruction errors can be used to measure the abnormality of each node. Our experiments show that the proposed DOMINANT outperforms the state-of-the-art methods.

5.1 Future Directions

In the future, I plan to have an adaptive research agenda to keep updated with emerging challenges in machine learning and data mining. Furthermore, I hope to leverage my expertise to conduct interdisciplinary research to extend the power of graph neural networks. Below, I discuss two future research directions:

- **Resource-efficient Deep Learning:** Deep neural networks especially large pre-trained models (e.g., PLMs and ViT) are becoming omnipresent in different domains ranging from computer vision to natural language processing. Despite their success that largely relies on the over-parameterization and huge amount of labeled data, training or fine-tuning these models can be very resource-expensive and challenging in practical scenarios. To democratize AI algorithms at scale, it is of vital importance to invest on enhancing the efficiency of large-scale models in terms of both labeled training data and model parameters. For the research on data efficiency (e.g., weakly-supervised learning, few-shot learning, and self-supervised learning), I have developed a series of works on graph learning (Ding, Xu, et al. 2022) as well as natural language processing problems (Ding, D. Li, et al. 2021). In the next stage, I will continue my explorations and develop more fundamental and theoretical understanding of data-efficient deep learning. Moreover, I plan to start my research on parameter-efficient transfer learning, which is another critical research field for advancing the generalization of deep learning models to different resource-constrained scenarios. By developing a suite of new models, algorithms, and theories on both data and parameter efficient deep learning, we are able to free AI machines from the resource-hungry beast.

- **Trustworthy Machine Learning:** Though in recent years huge advances have been made in deep learning, current machine learning systems, however, generally lacks of trustwothiness, which poses great challenges for their usability in many high-stake applications such as fraud detection and drug design. Hence, I plan to study the core topics regarding trustworthy machine learning, with a focus on the statistical understructure of various topics under this umbrella topic, and also branching out the central understanding to solve multiple problems in topics such as reliability, causality, and uncertainty quantification, with a main application domain of graph learning. For instance, machine learning models deployed in the open world often struggle with out-of-distribution (OOD) input samples from a different distribution that the model has not been exposed to during training. How to detect out-of-distribution samples and enable out-of-distribution generalization is important to improve the reliability of the deployed machine learning system. Moreover, to answer the research question "Why and how does a ML model work?", I am specifically interested in studying generative counterfactual explanation, which aims to analyze the feature importance and model sensitivity by generating counterfactual examples. Such counterfactual explanations can help us to understand how decisions made by ML models, which bridges the gap between artificial intelligence and human intelligence.

REFERENCES

Akoglu, Leman, Hanghang Tong, and Danai Koutra. 2015. "Graph based anomaly detection and description: a survey." *DMKD.*

Akoglu, Leman, Hanghang Tong, Brendan Meeder, and Christos Faloutsos. 2012. "Pics: Parameter-free identification of cohesive subgroups in large attributed graphs." In *SDM,* 439–450.

Alon, Uri, and Eran Yahav. 2021. "On the bottleneck of graph neural networks and its practical implications." In *ICLR.*

Arazo, Eric, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. 2020. "Pseudo-labeling and confirmation bias in deep semi-supervised learning." In *IJCNN.*

Bandyopadhyay, Sambaran, N Lokesh, and M Narasimha Murty. 2019. "Outlier aware network embedding for attributed networks." In *AAAI.*

Baydin, Atilim Gunes, Robert Cornish, David Martinez Rubio, Mark Schmidt, and Frank Wood. 2018. "Online Learning Rate Adaptation with Hypergradient Descent." In *ICLR.*

Bose, Avishek Joey, Ankit Jain, Piero Molino, and William L Hamilton. 2019. "Meta-Graph: Few Shot Link Prediction via Meta Learning." *arXiv preprint arXiv:1912.09867.*

Breunig, Markus M, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. "LOF: identifying density-based local outliers." In *SIGMOD.*

Chalapathy, Raghavendra, Aditya Krishna Menon, and Sanjay Chawla. 2017. "Robust, deep and inductive anomaly detection." In *ECML-PKDD.*

Chen, Deli, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view." In *AAAI.*

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. "A simple framework for contrastive learning of visual representations." In *ICML.*

Chien, Eli, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. "Adaptive Universal Generalized PageRank Graph Neural Network." In *ICLR.*

Chuang, Ching-Yao, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. 2020. “Debiased contrastive learning.” *arXiv preprint arXiv:2007.00224.*

Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. “Natural language processing (almost) from scratch.” *Journal of machine learning research* 12 (Aug): 2493–2537.

Dai, Quanyu, Xiao Shen, Xiao-Ming Wu, and Dan Wang. 2019. “Network transfer learning via adversarial domain adaptation with graph convolution.” *arXiv preprint arXiv:1909.01541.*

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. 2016. “Convolutional neural networks on graphs with fast localized spectral filtering.” In *NeurIPS.*

Deng, Shumin, Ningyu Zhang, Jiaojian Kang, Yichi Zhang, Wei Zhang, and Huajun Chen. 2020. “Meta-Learning with Dynamic-Memory-Based Prototypical Network for Few-Shot Event Detection.” In *Proceedings of the International Conference on Web Search and Data Mining.*

Ding, Kaize, Dingcheng Li, Alexander Hanbo Li, Xing Fan, Chenlei Guo, Yang Liu, and Huan Liu. 2021. “Learning to Selectively Learn for Weakly-supervised Paraphrase Generation.” In *EMNLP.*

Ding, Kaize, Jundong Li, Nitin Agarwal, and Huan Liu. 2020. “Inductive Anomaly Detection on Attributed Networks.” In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence.*

Ding, Kaize, Jundong Li, Rohit Bhanushali, and Huan Liu. 2019. “Deep anomaly detection on attributed networks.” In *Proceedings of the SIAM International Conference on Data Mining.*

Ding, Kaize, Jundong Li, and Huan Liu. 2019. “Interactive anomaly detection on attributed networks.” In *Proceedings of the ACM International Conference on Web Search and Data Mining.*

Ding, Kaize, Jianling Wang, James Caverlee, and Huan Liu. 2022. “Meta Propagation Networks for Few-shot Semi-supervised Learning on Graphs.” In *AAAI.*

Ding, Kaize, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. “Be More with Less: Hypergraph Attention Networks for Inductive Text Classification.” In *EMNLP.*

Ding, Kaize, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. 2020. "Graph prototypical networks for few-shot learning on attributed networks." In *CIKM.*

Ding, Kaize, Zhe Xu, Hanghang Tong, and Huan Liu. 2022. "Data Augmentation for Deep Graph Learning: A Survey." *arXiv preprint arXiv:2202.08235.*

Ding, Kaize, Chuxu Zhang, Jie Tang, Nitesh Chawla, and Huan Liu. 2022. "Toward Graph Minimally-Supervised Learning." In *KDD.*

Ding, Kaize, Qinghai Zhou, Hanghang Tong, and Huan Liu. 2021. "Few-shot Network Anomaly Detection via Cross-network Meta-learning." In *TheWebConf.*

Dou, Yingtong, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. 2020. "Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters." In *CIKM.*

Du, Yuanqi, Shiyu Wang, Xiaojie Guo, Hengning Cao, Shujie Hu, Junji Jiang, Aishwarya Varala, Abhinav Angirekula, and Liang Zhao. 2021. "GraphGT: Machine Learning Datasets for Deep Graph Generation and Transformation." In *NeurIPS.*

Feng, Wenzheng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. 2020. "Graph random neural networks for semi-supervised learning on graphs." In *NeurIPS.*

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. 2017. "Model-agnostic meta-learning for fast adaptation of deep networks." In *ICML.*

Fout, Alex, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. 2017. "Protein Interface Prediction using Graph Convolutional Networks." In *NeurIPS.*

Ganin, Yaroslav, and Victor Lempitsky. 2014. "Unsupervised domain adaptation by backpropagation." *arXiv preprint arXiv:1409.7495.*

Ganin, Yaroslav, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. "Domain-adversarial training of neural networks." In *JMLR.*

Gao, Jing, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, and Jiawei Han. 2010. "On community outliers and their efficient detection in information networks." In *KDD.*

Garcia, Victor, and Joan Bruna. 2018. "Few-shot learning with graph neural networks." *Proceedings of the International Conference on Learning Representations.*

Garcia-Teodoro, Pedro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. 2009. "Anomaly-based network intrusion detection: Techniques, systems and challenges." *computers & security.*

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. 2011. "Domain adaptation for large-scale sentiment classification: A deep learning approach." In *ICML.*

Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning.* Vol. 1.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. "Generative adversarial nets." In *NeurIPS.*

Grover, Aditya, and Jure Leskovec. 2016. "node2vec: Scalable feature learning for networks." In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining.*

Hadsell, Raia, Sumit Chopra, and Yann LeCun. 2006. "Dimensionality reduction by learning an invariant mapping." In *CVPR.*

Hamilton, Will, Zhitao Ying, and Jure Leskovec. 2017. "Inductive representation learning on large graphs." In *NeurIPS.*

Hassani, Kaveh, and Amir Hosein Khasahmadi. 2020. "Contrastive multi-view representation learning on graphs." In *ICML.*

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. "Momentum contrast for unsupervised visual representation learning." In *CVPR.*

Hochreiter, Sepp, A Steven Younger, and Peter R Conwell. 2001. "Learning to learn using gradient descent." In *ICANN.*

Hoffman, Judy, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. 2014. "LSDA: Large scale detection through adaptation." In *NeurIPS.*

Hu, Weihua, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. "Open graph benchmark: Datasets for machine learning on graphs." In *NeurIPS.*

Hu, Xia, Jiliang Tang, Yanchao Zhang, and Huan Liu. 2013. "Social spammer detection in microblogging." In *IJCAI.*

Huang, Qian, Horace He, Abhay Singh, Ser-Nam Lim, and Austin R Benson. 2021. "Combining Label Propagation and Simple Models Out-performs Graph Neural Networks." In *ICLR.*

Huang, Xiao, Jundong Li, and Xia Hu. 2017. "Label informed attributed network embedding." In *WSDM,* 731–739. ACM.

Huang, Xiao, Qingquan Song, Jundong Li, and Xia Hu. 2018. "Exploring expert cognition for attributed network embedding." In *WSDM.*

Jin, Ming, Yizhen Zheng, Yuan-Fang Li, Chen Gong, Chuan Zhou, and Shirui Pan. 2021. "Multi-Scale Contrastive Siamese Networks for Self-Supervised Graph Representation Learning." In *IJCAI.*

Kaghazgaran, Parisa, James Caverlee, and Anna Squicciarini. 2018. "Combating crowdsourced review manipulators: A neighborhood-based approach." In *WSDM.*

Kingma, Diederik P, and Jimmy Ba. 2014. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980.*

Kipf, Thomas N, and Max Welling. 2017a. "Variational graph auto-encoders." In *ICLR.*

———. 2017b. "Semi-Supervised Classification with Graph Convolutional Networks." In *ICLR.*

Klicpera, Johannes, Aleksandar Bojchevski, and Stephan Günnemann. 2019. "Predict then propagate: Graph neural networks meet personalized pagerank." In *ICLR.*

Klicpera, Johannes, Janek Groß, and Stephan Günnemann. 2019. "Directional Message Passing for Molecular Graphs." In *ICLR.*

Koch, Gregory, Richard Zemel, and Ruslan Salakhutdinov. 2015. "Siamese neural networks for one-shot image recognition." In *ICML deep learning workshop.*

Kriegel, Hans-Peter, Peer Kroger, Erich Schubert, and Arthur Zimek. 2011. "Interpreting and unifying outlier scores." In *SDM.*

Kulis, Brian, and Michael I Jordan. 2011. "Revisiting k-means: New algorithms via Bayesian nonparametrics." *arXiv preprint arXiv:1111.0352.*

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep learning." *nature.*

Li, Ao, Zhou Qin, Runshi Liu, Yiqun Yang, and Dong Li. 2019. "Spam review detection with graph convolutional networks." In *CIKM.*

Li, Jundong, Harsh Dani, Xia Hu, and Huan Liu. 2017. "Radar: Residual Analysis for Anomaly Detection in Attributed Networks." In *IJCAI.*

Li, Jundong, Harsh Dani, Xia Hu, Jiliang Tang, Yi Chang, and Huan Liu. 2017. "Attributed network embedding for learning in a dynamic environment." In *CIKM,* 387–396.

Li, Jundong, Xia Hu, Jiliang Tang, and Huan Liu. 2015. "Unsupervised streaming feature selection in social media." In *CIKM,* 1041–1050.

Li, Junnan, Pan Zhou, Caiming Xiong, and Steven CH Hoi. 2021. "Prototypical contrastive learning of unsupervised representations." In *ICLR.*

Li, Qimai, Zhichao Han, and Xiao-Ming Wu. 2018. "Deeper insights into graph convolutional networks for semi-supervised learning." In *AAAI.*

Li, Qimai, Xiao-Ming Wu, Han Liu, Xiaotong Zhang, and Zhichao Guan. 2019. "Label efficient semi-supervised learning via graph filtering." In *CVPR.*

Li, Yuening, Xiao Huang, Jundong Li, Mengnan Du, and Na Zou. 2019. "SpecAE: Spectral AutoEncoder for Anomaly Detection in Attributed Networks." In *CIKM.*

Li, Zheng, Ying Wei, Yu Zhang, and Qiang Yang. 2018. "Hierarchical attention transfer network for cross-domain sentiment classification." In *AAAI.*

Li, Zhenguo, Fengwei Zhou, Fei Chen, and Hang Li. 2017. "Meta-sgd: Learning to learn quickly for few-shot learning." *arXiv preprint arXiv:1707.09835.*

Liu, Hanxiao, Karen Simonyan, and Yiming Yang. 2018. "DARTS: Differentiable Architecture Search." In *ICLR.*

Liu, Lu, Tianyi Zhou, Guodong Long, Jing Jiang, Xuanyi Dong, and Chengqi Zhang. 2021. "Isometric Propagation Network for Generalized Zero-shot Learning." In *ICLR.*

Liu, Lu, Tianyi Zhou, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. 2019. "Prototype propagation networks (PPN) for weakly-supervised few-shot learning on category graph." In *NeurIPS.*

Liu, Lu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. “Learning to propagate for graph meta-learning.” In *Proceedings of the Annual Conference on Neural Information Processing Systems.*

Liu, Meng, Hongyang Gao, and Shuiwang Ji. 2020. “Towards deeper graph neural networks.” In *KDD.*

Lloyd, Stuart. 1982. “Least squares quantization in PCM.” *IEEE transactions on information theory.*

Manning, Christopher D, Hinrich Schütze, and Prabhakar Raghavan. 2008. *Introduction to information retrieval.* Cambridge university press.

McAuley, Julian, Rahul Pandey, and Jure Leskovec. 2015. “Inferring networks of substitutable and complementary products.” In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

McPherson, Miller, Lynn Smith-Lovin, and James M Cook. 2001. “Birds of a feather: Homophily in social networks.” *Annual review of sociology.*

Mishra, Nikhil, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. “A Simple Neural Attentive Meta-Learner.” In *Proceedings of the International Conference on Learning Representations.*

Mo, Yujie, Liang Peng, Jie Xu, Xiaoshuang Shi, and Xiaofeng Zhu. 2022. “Simple Unsupervised Graph Representation Learning.” AAAI.

Müller, Emmanuel, Patricia Iglesias Sánchez, Yvonne Mülle, and Klemens Böhm. 2013. “Ranking outlier nodes in subspaces of attributed graphs.” In *ICDE Workshop.*

Namata, Galileo, Ben London, Lise Getoor, Bert Huang, and UMD EDU. 2012. “Query-driven active surveying for collective classification.” In *Workshop on MLG.*

Northcutt, Stephen, and Judy Novak. 2002. *Network intrusion detection.* Sams Publishing.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. 2018. “Representation learning with contrastive predictive coding.” *arXiv preprint arXiv:1807.03748.*

Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The pagerank citation ranking: Bringing order to the web.* Technical report. Stanford InfoLab.

Pan, Shirui, Ruiqi Hu, Sai-fu Fung, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. "Learning graph embedding with adversarial training methods." *IEEE Transactions on Cybernetics.*

Pan, Shirui, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. 2018. "Adversarially Regularized Graph Autoencoder for Graph Embedding." In *IJCAI.*

Pan, Sinno Jialin, and Qiang Yang. 2009. "A survey on transfer learning." In *TKDE.*

Pang, Guansong, Chunhua Shen, Longbing Cao, and Anton van den Hengel. 2020. "Deep learning for anomaly detection: A review." *arXiv preprint arXiv:2007.02500.*

Pang, Guansong, Chunhua Shen, and Anton van den Hengel. 2019. "Deep anomaly detection with deviation networks." In *KDD.*

Park, Jiwoong, Minsik Lee, Hyung Jin Chang, Kyuewang Lee, and Jin Young Choi. 2019. "Symmetric graph convolutional autoencoder for unsupervised graph representation learning." In *ICCV,* 6519–6528.

Park, Namyong, Andrey Kan, Xin Luna Dong, Tong Zhao, and Christos Faloutsos. 2019. "Estimating node importance in knowledge graphs using graph neural networks." In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Peng, Zhen, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. "Graph Representation Learning via Graphical Mutual Information Maximization." In *WWW.*

Peng, Zhen, Minnan Luo, Jundong Li, Huan Liu, and Qinghua Zheng. 2018. "ANOMALOUS: A Joint Modeling Approach for Anomaly Detection on Attributed Networks." In *IJCAI.*

Perozzi, Bryan, and Leman Akoglu. 2016. "Scalable anomaly ranking of attributed neighborhoods." In *SDM.*

Perozzi, Bryan, Leman Akoglu, Patricia Iglesias Sánchez, and Emmanuel Müller. 2014. "Focused clustering and outlier detection in large attributed graphs." In *KDD.*

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. 2014. "Deepwalk: Online learning of social representations." In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining.*

Pfeiffer III, Joseph J, Sebastian Moreno, Timothy La Fond, Jennifer Neville, and Brian Gallagher. 2014. "Attributed graph models: Modeling network structure with correlated attributes." In *WWW.*

Qi, Guo-Jun, Charu Aggarwal, Qi Tian, Heng Ji, and Thomas Huang. 2011. "Exploring context and content links in social media: A latent space method." *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Qiao, Limeng, Yemin Shi, Jia Li, Yaowei Wang, Tiejun Huang, and Yonghong Tian. 2019. "Transductive episodic-wise adaptive metric for few-shot learning." In *Proceedings of the IEEE International Conference on Computer Vision.*

Qiu, Jiezhong, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. "Gcc: Graph contrastive coding for graph neural network pre-training." In *KDD.*

Qu, Chen, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W Bruce Croft. 2019. "Learning to Selectively Transfer: Reinforced Transfer Learning for Deep Text Matching." In *WSDM.*

Ravi, Sachin, and Hugo Larochelle. 2017. "Optimization as a model for few-shot learning." In *Proceedings of the International Conference on Learning Representations.*

Rayana, Shebuti, and Leman Akoglu. 2015. "Collective opinion spam detection: Bridging review networks and metadata." In *KDD.*

Ren, Mengye, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. 2018. "Meta-learning for semi-supervised few-shot classification." In *Proceedings of the International Conference on Learning Representations.*

Ren, Mengye, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. "Learning to reweight examples for robust deep learning." In *ICML.*

Resnik, Philip, and Eric Hardisty. 2010. *Gibbs sampling for the uninitiated.* Technical report. Maryland Univ College Park Inst for Advanced Computer Studies.

Ruff, Lukas, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. 2020. "Deep Semi-Supervised Anomaly Detection." In *ICLR.*

Saenko, Kate, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. "Adapting visual category models to new domains." In *ECCV.*

Sánchez, Patricia Iglesias, Emmanuel Muller, Fabian Laforet, Fabian Keller, and Klemens Bohm. 2013. "Statistical selection of congruent subspaces for mining attributed graphs." In *ICDM,* 647–656.

Sánchez, Patricia Iglesias, Emmanuel Müller, Oretta Irmler, and Klemens Böhm. 2014. "Local context selection for outlier ranking in graphs with multiple numeric node attributes." In *SSDBM.*

Santoro, Adam, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. "Meta-learning with memory-augmented neural networks." In *ICML.*

Sen, Prithviraj, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008a. "Collective classification in network data." *AI magazine.*

———. 2008b. "Collective classification in network data." *AI Magazine.*

Shalizi, Cosma Rohilla, and Andrew C Thomas. 2011. "Homophily and contagion are generically confounded in observational social network studies." *Sociological Methods & Research* 40 (2): 211–239.

Shchur, Oleksandr, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. "Pitfalls of graph neural network evaluation." *arXiv preprint arXiv:1811.05868.*

Shen, Xiao, Quanyu Dai, Fu-lai Chung, Wei Lu, and Kup-Sze Choi. 2020. "Adversarial Deep Network Embedding for Cross-Network Node Classification." In *AAAI.*

Shu, Yang, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. 2019. "Transferable Curriculum for Weakly-Supervised Domain Adaptation." In *AAAI.*

Skillicorn, David B. 2007. "Detecting anomalies in graphs." In *ISI.*

Snell, Jake, Kevin Swersky, and Richard Zemel. 2017. "Prototypical networks for few-shot learning." In *Proceedings of the Annual Conference on Neural Information Processing Systems.*

Song, Xiuyao, Mingxi Wu, Christopher Jermaine, and Sanjay Ranka. 2007. "Conditional anomaly detection." *TKDE.*

Sun, Ke, Zhouchen Lin, and Zhanxing Zhu. 2020. "Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes." In *AAAI.*

Sung, Flood, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. “Learning to compare: Relation network for few-shot learning.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

Tang, Jie, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. “Arnetminer: extraction and mining of academic social networks.” In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Tang, Xianfeng, Yandong Li, Yiwei Sun, Huaxiu Yao, Prasenjit Mitra, and Suhang Wang. 2020. “Transferring Robustness for Graph Neural Network Against Poisoning Attacks.” In *WSDM.*

Tong, Hanghang, and Ching-Yung Lin. 2011. “Non-negative residual matrix factorization with application to graph anomaly detection.” In *SDM.*

Tzeng, Eric, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. “Adversarial discriminative domain adaptation.” In *CVPR.*

Tzeng, Eric, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. “Deep domain confusion: Maximizing for domain invariance.” *arXiv preprint arXiv:1412.3474.*

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018a. “Graph attention networks.” In *Proceedings of the International Conference on Learning Representations.*

———. 2018b. “Graph attention networks.” In *ICLR.*

———. 2018c. “Graph attention networks.” In *ICLR.*

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. “Graph Attention Networks.” In *ICLR.*

Veličković, Petar, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. “Deep Graph Infomax.” In *ICLR.*

Vinyals, Oriol, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. “Matching networks for one shot learning.” In *Proceedings of the Annual Conference on Neural Information Processing Systems.*

Wang, Daixin, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. "A Semi-supervised Graph Attentive Network for Financial Fraud Detection." In *ICDM.*

Wang, Jianling, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. "Next-item Recommendation with Sequential Hypergraphs." In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Wang, Ning, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. 2020. "Graph Few-shot Learning with Attribute Matching." In *CIKM.*

Wang, Xiao, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. "Heterogeneous graph attention network." In *TheWebConf.*

West, Jarrod, and Maumita Bhattacharya. 2016. "Intelligent financial fraud detection: a comprehensive review." *Computers & security* 57:47–66.

Wu, Felix, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. "Simplifying graph convolutional networks." In *ICML.*

Wu, Man, Shirui Pan, Lan Du, Ivor Tsang, Xingquan Zhu, and Bo Du. 2019. "Long-short Distance Aggregation Networks for Positive Unlabeled Graph Learning." In *CIKM.*

Wu, Man, Shirui Pan, Chuan Zhou, Xiaojun Chang, and Xingquan Zhu. 2020. "Unsupervised Domain Adaptive Graph Convolutional Networks." In *The Web Conference.*

Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. "A comprehensive survey on graph neural networks." *TNNLS.*

Xia, Yan, Xudong Cao, Fang Wen, Gang Hua, and Jian Sun. 2015. "Learning discriminative reconstructions for unsupervised outlier removal." In *ICCV.*

Xu, Keyulu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. "How powerful are graph neural networks?" In *ICLR.*

Xu, Keyulu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. "Representation learning on graphs with jumping knowledge networks." In *ICML.*

Xu, Xiaowei, Nurcan Yuruk, Zhidan Feng, and Thomas AJ Schweiger. 2007. "Scan: a structural clustering algorithm for networks." In *Proceedings of the 13th ACM*

SIGKDD International Conference on Knowledge Discovery and Data mining (KDD).

Yao, Huaxiu, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh Chawla, and Zhenhui Li. 2020a. "Graph few-shot learning via knowledge transfer." In *AAAI.*

Yao, Huaxiu, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh V Chawla, and Zhenhui Li. 2020b. "Graph few-shot learning via knowledge transfer." In *Proceedings of the AAAI Conference on Artificial Intelligence.*

You, Jiaxuan, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. 2018. "Graph convolutional policy network for goal-directed molecular graph generation." In *NeurIPS.*

You, Yuning, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. "Graph contrastive learning with augmentations." In *NeurIPS.*

Yu, Jianfei, Minghui Qiu, Jing Jiang, Jun Huang, Shuangyong Song, Wei Chu, and Haiqing Chen. 2018. "Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce." In *WSDM.*

Yu, Wenchao, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. 2018. "NetWalk: A Flexible Deep Embedding Approach for Anomaly Detection in Dynamic Networks." In *KDD,* 2672–2681.

Zafarani, Reza, Mohammad Ali Abbasi, and Huan Liu. 2014. *Social media mining: an introduction.* Cambridge University Press.

Zhang, Chuxu, Kaize Ding, Jundong Li, Xiangliang Zhang, Yanfang Ye, Nitesh V Chawla, and Huan Liu. 2022. "Few-Shot Learning on Graphs: A Survey." In *IJCAI.*

Zhang, Jian, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaokang Yang. 2019. "Variational few-shot learning." In *Proceedings of the IEEE International Conference on Computer Vision.*

Zhang, Tiantian, and Bin Wu. 2012. "A method for local community detection by finding core nodes." In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining.*

Zhang, Yin, Rong Jin, and Zhi-Hua Zhou. 2010. “Understanding bag-of-words model: a statistical framework.” *International Journal of Machine Learning and Cybernetics.*

Zhang, Yizhou, Guojie Song, Lun Du, Shuwen Yang, and Yilun Jin. 2019. “Dane: Domain adaptive network embedding.” In *IJCAI.*

Zhang, Zhen-Yu, Peng Zhao, Yuan Jiang, and Zhi-Hua Zhou. 2019. “Learning from incomplete and inaccurate supervision.” In *KDD.*

Zhao, Tong, Chuchen Deng, Kaifeng Yu, Tianwen Jiang, Daheng Wang, and Meng Jiang. 2020. “Error-Bounded Graph Anomaly Loss for GNNs.” In *CIKM.*

Zheng, Shuai, Zhenfeng Zhu, Xingxing Zhang, Zhizhe Liu, Jian Cheng, and Yao Zhao. 2020. “Distribution-induced bidirectional generative adversarial network for graph representation learning.” In *CVPR.*

Zhou, Chong, and Randy C Paffenroth. 2017. “Anomaly detection with robust deep autoencoders.” In *KDD.*

Zhou, Dawei, Jingrui He, Hongxia Yang, and Wei Fan. 2018. “Sparc: Self-paced network representation for few-shot rare category characterization.” In *KDD.*

Zhou, Dengyong, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. “Learning with local and global consistency.” In *NeurIPS.*

Zhou, Fan, Chengtai Cao, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Ji Geng. 2020. “Fast Network Alignment via Graph Meta-Learning.” In *INFOCOM.*

Zhou, Fan, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. 2019. “Meta-GNN: On Few-shot Node Classification in Graph Meta-learning.” In *CIKM.*

Zhou, Qinghai, Liangyue Li, Nan Cao, Lei Ying, and Hanghang Tong. 2019. “ADMIRING: Adversarial multi-network mining.” In *ICDM.*

Zhou, Qinghai, Liangyue Li, and Hanghang Tong. 2019. “Towards Real Time Team Optimization.” In *Big Data.*

Zhu, Xiaojin, and Zoubin Ghahramani. 2002. “Learning from labeled and unlabeled data with label propagation.” *Technical Report.*

Zhu, Xiaojin, Zoubin Ghahramani, and John D Lafferty. 2003. “Semi-supervised learning using gaussian fields and harmonic functions.” In *ICML.*

Zhu, Yanqiao, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020a. "Deep Graph Contrastive Representation Learning." In *ICML Workshop.*

———. 2020b. "Graph Contrastive Learning with Adaptive Augmentation." In *TheWebConf.*

Zhuang, Fuzhen, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. 2015. "Supervised representation learning: Transfer learning with deep autoencoders." In *IJCAI.*

Zügner, Daniel, Amir Akbarnejad, and Stephan Günnemann. 2018. "Adversarial attacks on neural networks for graph data." In *KDD.*