Algorithmic Fairness in Graph Machine Learning: Explanation, Optimization, and Certification

YUSHUN DONG

Ph.D. Dissertation

Advisor: Dr. Jundong Li

Doctoral Committee:
Dr. Cong Shen, Chair
Dr. Jundong Li
Dr. Nikolaos D. Sidiropoulos
Dr. Hanghang Tong
Dr. Aidong Zhang

A report submitted in fulfillment of the requirements for
the Ph.D. Dissertation

Department of Electrical and Computer Engineering

University of Virginia

June 2024

Abstract

Network data is ubiquitous across diverse domains such as credit scoring, social networking, recommendation systems, and medical diagnosis. In this landscape, graph machine learning algorithms, e.g., Graph Neural Networks (GNNs), have emerged as powerful tools for modeling such data and performing predictive tasks. However, despite the effectiveness of existing graph machine learning algorithms, they usually bear the problem of exhibiting bias in the prediction results. This is particularly alarming under high-stakes decision-making contexts, since these algorithms could play pivotal roles and influence life-altering choices for involved individuals. Consequently, it becomes paramount to delve into the root causes of such biases and improve the level of fairness, thereby enhancing the trustworthiness of these algorithms.

Nevertheless, addressing such a problem is non-trivial, and a plethora of unresolved questions loom large. For example, why does bias arise in graph machine learning? How to quantitatively measure the exhibited bias under different notions? How can we mitigate these biases, ensuring fair outcomes when these algorithms guide critical decisions? How to robustify the fairness level of graph machine learning algorithms against potential malicious attacks? How to remove the sensitive information that may lead to bias when it has been encoded in the graph machine learning algorithms in the training stage? It is necessary to properly answer these questions to ensure the trustworthiness of graph machine learning algorithms deployed in real-world applications. However, despite the significance of the algorithmic bias issue, the corresponding study remains at an early stage.

To answer the questions above, my Ph.D. dissertation mainly contributes to the advancement of graph machine learning through a fairness lens. Specifically, this dissertation mainly focuses on three research themes, including algorithmic fairness explanation in graph machine learning, algorithmic fairness optimization in graph machine learning, and fairness certification in graph machine learning. In the first theme, we present qualitative and quantitative analysis to understand why bias arises and where the bias comes from in graph machine learning. In the second theme, we present principled frameworks and strategies to mitigate the bias exhibited in graph machine learning algorithms, where multiple fairness notions are taken into consideration. In the third theme, we propose theoretical certification on graph machine learning algorithms to achieve defense on the level of fairness and removal of sensitive information that may bring bias from the model to achieve protection for privacy.

With these research themes, this dissertation aims to enhance the understanding of fair decision-making in the realm of graph machine learning and paves the way for future advances.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my Ph.D. advisor, Dr. Jundong Li for his guidance, support, and encouragement throughout my Ph.D. journey. His key insights, forward-looking research ideas, and meticulous attention to details have greatly benefited my growth as a professional researcher. Dr. Li's dedication to fostering a collaborative lab environment has been truly inspiring. I am fortunate to learn from and work with such an outstanding mentor, whose wisdom and kindness have played a key role in my academic and personal development.

I would also like to extend my sincere appreciation to my dedicated dissertation committee members for their invaluable contributions and support. I am deeply grateful to Dr. Nikolaos D. Sidiropoulos and Dr. Cong Shen for serving as the chair of my committee for the dissertation proposal and final dissertation defense, respectively. I would also like to express my heartfelt thanks to Dr. Aidong Zhang for her expertise, constructive suggestions, and thought-provoking questions, which have greatly enhanced the quality and depth of my research. I am particularly grateful to Dr. Hanghang Tong from the University of Illinois at Urbana-Champaign, who has not only served as a member of my dissertation committee but also as a supportive collaborator. Dr. Tong's expertise in data mining and graph learning have significantly contributed to the success of our collaborative projects. I am grateful for the efforts he has dedicated to our research collaborations, offering insightful feedback and constructive suggestions. His guidance have been critical in shaping me as a professional researcher, and I am truly fortunate to have had the opportunity to learn from him.

Meanwhile, my heartfelt thanks also go to other collaborators in academia, including Dr. Chen Chen from the University of Virginia, Dr. Tyler Derr from Vanderbilt University, Dr. Kaize Ding from Northwestern University, Dr. Brian Jalaian from the University of West Florida, Dr. Shuiwang Ji from Texas A&M University, Dr. Jian Kang from the University of Rochester, Dr. Ninghao Liu from the University of Georgia, Dr. Jing Ma from Case Western Reserve University, Dr. Daniel Mietchen from the Leibniz Institute of Freshwater Ecology and Inland Fisheries, Dr. Tianhao Wang from the University of Virginia, Dr. Qi Wang from Northeastern University, Dr. Yu Wang from the University of Oregon, Dr. Yue Zhao from the University of Southern California, and Dr. Na Zou from the University of Houston. Their diverse perspectives, collaborative spirit, and intellectual contributions have enriched my research and broadened my horizons. It is an honor to work with these talented researchers.

I am also grateful for the invaluable collaborations and internship opportunities with my industry partners, who have also significantly enriched my research experience and provided me with a deeper understanding of the real-world applications and impact of my work. I would like to express my sincere gratitude to Dr. Tobias Schnabel, my mentor at Microsoft, for his guidance, support, and insightful discussions during my internship at Microsoft Research. His expertise and practical insights have greatly contributed to my growth as a researcher and have helped me bridge the gap between academia and industry. I am also deeply thankful

for my internship experience at Snap, where I had the privilege of working with exceptional mentors and collaborators. I would like to express my appreciation to Dr. Neil Shah and Dr. Tong Zhao, my mentors at Snap, for their guidance, valuable feedback, and collaborative spirit, which have been instrumental in shaping my research direction and honing my skills. I am also grateful to have had the opportunity to collaborate with Dr. William Shiao from the University of California, Riverside, who interned together with me at Snap and became a valued collaborator. Furthermore, I would like to thank Yozen Liu, another collaborator at Snap, for his support and insightful contributions to our research endeavors. I am grateful for the opportunity to have worked on cutting-edge projects and to have gained exposure to the challenges and opportunities in the industry. Additionally, I would like to extend my appreciation to Dr. Huiyuan Chen from Visa for his collaboration and support. These internships and industry collaborations have not only enhanced my research but have also provided me with a comprehensive understanding of the practical applications.

Furthermore, I would like to acknowledge the tremendous support from my fellow researchers in Dr. Jundong Li's lab at the University of Virginia, including Xingbo Fu, Yinhan He, Zhenyu Lei, Haochen Liu, Song Wang, Zhiming Xu, Binchi Zhang, and Zaiyi Zheng. The shared moments of enjoying success and facing challenges have made this journey exciting. I am grateful for our friendships and the invaluable knowledge we have gained together.

I would also like to extend my appreciation to the faculty, staff, and fellow graduate students at the University of Virginia for fostering a supportive academic environment. The resources, facilities, and opportunities provided by the university have been instrumental in the successful completion of my Ph.D. program.

Last but not least, I am grateful to my family and friends for their love and encouragement throughout this challenging yet rewarding journey. Their belief in me has been a constant source of motivation and strength, enabling me to persevere through the ups and downs of my Ph.D. experience. I am especially grateful to my parents, whose sacrifices and unconditional support have made my achievements possible.

Contents

CHAPTER 1

Introduction

Background. Network data is ubiquitous over a wide range of applications, e.g., financial fraud detection [193, 159] and social recommendation [64], due to its proficiency of representing pervasive relational data. To gain deeper understanding of such data, various graph machine learning models, e.g., Graph Neural Networks (GNNs), have become popular over the years [111, 74]. Nevertheless, most existing graph machine learning models do not consider algorithmic fairness. Correspondingly, these models often yield results with discrimination towards specific demographic subgroups described by certain sensitive attributes [47, 37], e.g., gender and race. For example, in financial applications, such as loan approval prediction for bank clients [193, 216], different clients form a network based on their transactions, and the records of clients form their features. Here, the prediction goal is to classify whether a client will be approved for a loan, where GNNs have demonstrated superior performance. However, GNNs usually deliver biased classification results [3], e.g., rejecting a loan request only because the applicant belongs to an underprivileged group. Note that graph machine learning models have been widely deoplyed in various real-world applications, including high-stake decision-making scenarios such as healthcare. Therefore, the decisions made by those graph machine learning models could be life-changing for those involved individuals, and biased predictions could lead to serious consequences. It is thus critical to perform comprehensive study on the algorithmic fairness of graph machine learning for the trustworthiness of the deployed models.

Challenges. The primary goal of my research is to achieve trustworthy graph machine learning models from the perspective of algorithmic fairness. To achieve this goal, various questions remain to be answered. For example, why bias arises in graph machine learning? How to avoid biased decisions in scenarios where graph machine learning algorithms are deployed to help decision-making? How to robustify the fairness level of graph machine learning algorithms against potential malicious attacks? We note that this remains a daunting task due to the following three challenges. (C1): *Challenge of Fairness Explanation*. Most graph machine learning models are black-box, which makes them lack explanability. Although there are existing works explaining certain popular graph machine learning models (e.g., Graph Neural Networks), most of them only focus on given specific classification results, ignoring the level of fairness over the whole population. (C2): *Challenge of Fairness Optimization*. There are already a plethora of existing works focusing on debiasing traditional machine learning models on i.i.d. data. However, compared with them, debiasing graph machine learning models is more difficult. This is because most existing graph machine learning models are equipped with mechanisms that effectively learns the information from the dependency between nodes in the network data, e.g., the message-passing mechanism in GNNs. At the same time,

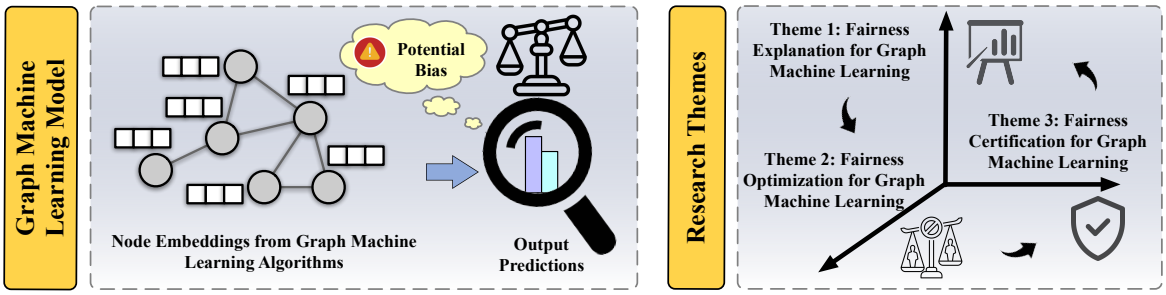


FIGURE 1.1. An overview of the research themes covered in this dissertation.

however, the bias originated from different nodes could also influence each other through the network topology, which makes debiasing more difficult to achieve. (C3): *Challenge of Fairness Certification*. Different from empirical debiasing, theoretical certification for the fairness of graph learning models can provide a way to guarantee trustworthy deployment. For example, by simply injecting several adversarial links in the network data, an attacker can make GNNs deliver advantaged predictions for a specific subgroup (e.g., individuals with a certain gender or nationality) while damaging the interest of others [83]. The defense on such attacks can be guaranteed if the fairness levels of the graph learning models are certified. As another example, biased information may be encoded in graph learning models during the training process. If the removal of such information can be guaranteed, the model can avoid biased outcomes that arise from such information. Nevertheless, such certification is difficult to achieve due to the randomness of optimization (e.g., the randomly initiated parameters) and the complex operations built in the graph learning models.

Dissertation Contributions. To tackle the challenges above (C1, C2, and C3), my research works mainly cover three research themes, including algorithmic fairness explanation in graph machine learning, algorithmic fairness optimization in graph machine learning, and fairness certification in graph machine learning. We present an overview in Fig. 1.1. Specifically, to tackle challenge C1, we introduce two research works that perform interpretation for the bias exhibited by graph learning models from the node and the graph structure perspective, respectively. To tackle challenge C2, we present three research works which aims to mitigate the exhibited bias of graph machine learning models, such that a satisfying balance can be achieved between the model utility and fairness. To tackle challenge C3, we introduce two works that guarantee the fairness level of the model and the removal of certain sensitive information from the model, respectively.

Dissertation Organizations. The remainder of this dissertation is organized as follows. We first review related work in Chapter 2. In Chapter 3, we introduce two research works that perform interpretation for the fairness of graph machine learning algorithms. In Chapter 4, we present three research works that achieve bias mitigation for graph machine learning algorithms. Chapter 5 introduces two research works that achieve certification on top of graph machine learning models to secure fairness levels and achieve information removal. Finally, in Chapter 6, we introduce the conclusion and several potential future directions in this area.

CHAPTER 2

Related Works

2.1 Explaining Graph Machine Learning Models

Among existing graph machine learning models, GNNs are among the most widely studied models in terms of explanation due to its superior performance in various network-based tasks. Generally, existing GNN explanation approaches can be divided into data-level approaches and model-level ones [229]. For data-level approaches, the explanation models identify critical components in the input network data of GNNs, e.g., node features or edges. For example, squared gradient values are regarded as the importance scores of different input features in the node classification task [7]; interpretable surrogate models are leveraged to approximate the prediction of a certain GNN model, where the explanations from the surrogate model can be regarded as the explanation for the corresponding GNN prediction [81, 191]. Another popular approach to identify important components of the input network data is to make perturbations on the input network, then observe the corresponding change in the output. The basic rationale is that if small perturbations lead to dramatic changes in the GNN prediction, then what has been perturbed is regarded as critical for the GNN prediction [228, 170, 197]. However, despite the significance in explaining GNNs, the corresponding study remains scarce. To provide model-level explanations for GNNs, graph generation can be leveraged to maximize the prediction of a GNN regarding a specific prediction (e.g., the probability of a class in graph classification) [230]. If the prediction probability of GNN regarding a specific prediction result can be maximized, then the generated input graph can be regarded as the explanation for this GNN that includes critical graph patterns. Note that most approaches above are designed to explain any specific prediction of the GNNs, while explaining how fairness arises is ignored. To handle such a problem, Dong et al. proposed to explain the exhibited bias at both data and model level. At the data level, a novel framework is proposed to characterize the edges that contribute to the fairness and bias the most [48], through which the exhibited bias can be attributed to each training node. At the model level, explanation strategies are proposed from the perspective of the GNN optimization [51], where the bias in GNNs is attributed to the labeled training nodes via influence function.

2.2 Debiasing Graph Machine Learning Models

Efforts have been made to mitigate bias exhibited by graph machine learning models, where these works can be broadly categorized into either focusing on *group fairness* or *individual fairness*. One of the pioneering methods employed to ensure group fairness is adversarial learning. This technique plays a min-max game between the generator and the discriminator.

If the learned node embeddings from the generator are unbiased, it then becomes difficult for the discriminator to differentiate. This indicates that the model's outputs are more fair across different demographic subgroups. Such an approach has been extensively discussed in studies such as those by Bose et al. [13] and by Dai et al. [37]. Debiasing with adversarial learning can also be achieved in a pre-processing manner, where the debiased network data can then be adopted by different GNNs. For example, Dong et al. [47] proposed a framework based on adversarial learning to pre-process the network data, such that the bias exhibited during the propagation of node attributes is mitigated. Another approach towards achieving group fairness is rebalancing. Such a method mainly aims to ensure any demographic subgroup should not be under-represented. For instance, Rahman et al. [162] proposed to mitigate bias by rebalancing the appearance rate of minority groups during random walks. By ensuring that minority groups appear more frequently during these walks, the model can be trained in a manner that doesn't marginalize these demographic subgroups. Other commonly used approaches include regularization [240, 3], edge re-wiring [47], and orthogonal projection [147].

Compared with the vast amount of works on group fairness, only few works promote individual fairness for graph machine learning models. Dwork et al. [58] first proposed the definition of individual fairness: *similar individuals should be treated similarly*. Here Lipschitz condition is utilized as the distance constraint for instance pairs between the input and outcome of the decision-making model. In graph machine learning, Kang et al. [104] first propose to systematically debias multiple graph mining algorithms based on individual fairness. Specifically, individual fairness is fulfilled by deriving an individual fairness loss on graph datasets and reduce it before, during, and after training of the graph mining model. In addition, individual fairness is also defined and optimized from a ranking perspective, such that this fairness notion can naturally calibrated across different datasets and applications. For example, Dong et al. [46] treat optimization of individual fairness in GNNs as a ranking problem, which helps to bypass the limitation of Lipschitz condition.

2.3 Securing Graph Machine Learning Models

Achieving certification serves as a key strategy to secure the graph machine learning models in different perspectives. In particular, here we mainly focus on two perspectives of securing graph machine learning models, including securing the performance and privacy of graph machine learning models with certification. In terms of securing the performance, existing works are categorized into five mainstreams, namely adversarial training, graph data purification, perturbation detection, adaptive information aggregation, and certified defense. Adversarial training aims to inject adversarial examples (e.g., edges) during training, such that the GNN tends to yield correct predictions for adversarial examples during inference [220, 41, 198]. Graph data purification also works during training, where graph data is purified during learning to weaken the influence of adversarial examples [110]. Perturbation detection is mostly applied in the pre-processing stage, where adversarial edges or nodes can be identified before training [223]. Adaptive information aggregation aims to learn personalized aggregation weights for edges and nodes, such that adversarial examples can be excluded from message passing. Different from all approaches above, certified defense aims to secure the model theoretically, such that attackers cannot find any adversary [171, 192, 10, 89].

However, most existing certified defense approaches only secure the prediction for a specific data point (e.g., a node in node classification). We note that the fairness levels of graph learning models has also been proved to be vulnerable to adversarial attacks [83], while most existing works can hardly be adapted. In fact, the fairness levels of the model predictions are influenced by all inference results under most commonly used fairness notions (e.g., statistical parity [83]), which makes this task more challenging. Different from other existing works, Dong et al. proposed to secure the level of fairness for GNNs [49] based on randomized smoothing [33], which took the first step to achieve fairness certification for various types of GNNs. In addition, securing GNNs in terms of the privacy and other sensitive information of the involved individuals (e.g., their gender or race) is also critical. For example, GNNs have been found to be vulnerable to privacy attacks and thus could leak sensitive information of the involved individuals [146]. In terms of securing the privacy of graph machine learning models, common approaches can be divided into three mainstreams, including the approaches based on differential privacy, federated learning, and adversarial learning [40]. Differential privacy provides theoretical guarantee on the privacy security by adding randomized noise to the training network data [217]; federated learning mainly aims to achieve data storage in a decentralized manner to protect sensitive information [215]; adversarial learning is commonly utilized to exclude certain sensitive information from the learned network embeddings [123]. Correspondingly, the excluded sensitive information cannot be inferred by malicious attackers. However, we note that in real-world applications, it is also necessary to remove certain information from an optimized graph learning model once the consent of using certain information has been withdrawn, i.e., unlearn such information from the model. Here, information removal not only serves as a critical approach for privacy protection, but also facilitates the level of fairness when such removed information brings bias. Several recent studies have taken early steps to achieve certified unlearning for GNNs [209, 30]. However, these works are not flexible enough to handle different types of unlearning requests, nor can they be generalized to different GNNs. Different from existing works, Dong et al. proposed a flexible unlearning framework for GNNs that can handle different common types of unlearning requests and be generalized to different types of GNNs to achieve information removal [53].

CHAPTER 3

Fairness Explanation for Graph Machine Learning

3.1 Interpreting Unfairness in Graph Neural Networks via Training Node Attribution

3.1.1 Introduction

Graph data is pervasive among a plethora of realms, e.g., financial fraud detection [193, 159, 28], social recommendation [64, 178, 72], and chemical reaction prediction [45, 175, 117]. As one of the state-of-the-art approaches to handle graph data, Graph Neural Networks (GNNs) have been attracting increasing attention [111, 74, 188]. Over the years, various graph analytical tasks have benefited from GNNs, where node classification is among the most widely studied ones [111, 205, 214]. Nevertheless, in node classification, GNNs often yield results with discrimination towards specific demographic subgroups described by certain sensitive attributes [47, 37, 3, 237, 201], such as gender, race, and religion. In many high-stake applications, critical decisions are made based on the classification results of GNNs [176], e.g., crime forecasting [92], and the exhibited bias (i.e., unfairness) is destructive for the involved individuals [50, 48, 177]. To tackle this problem, there has been a line of works focusing on debiasing GNNs in node classification [47, 37, 3, 46, 131, 39]. Their goal is to relieve the bias in GNN predictions on the test set, and here we refer to it as model bias.

In addition to debiasing GNNs, it is also critical to interpret how the model bias arises in GNNs. This is because such an understanding not only helps to determine whether a specific node should be involved in the training set, but also has much potential to guide the design of GNN debiasing methods [47, 131, 125]. Nevertheless, most existing GNN interpretation methods aim to understand how a prediction is made [229, 128] instead of other aspects such as fairness. Consequently, although the graph data has been proved to be a significant source of model bias [47, 125], existing works are unequipped to tackle this problem. In this paper, we aim to address this problem at the instance (node) level. Specifically, given a GNN trained for node classification, we aim to answer: "*To what extent the GNN model bias is influenced by the existence of a specific training node in this graph?*"

Nevertheless, answering the above question is technically challenging. Essentially, there are three main challenges: (1) *Influence Quantification.* To depict the influence of each training node on the model bias of GNNs, the first and foremost challenge is to design a principled fairness metric. A straightforward approach is to directly employ traditional fairness metrics (e.g., Δ_{SP} for *Statistical Parity* [58] and Δ_{EO} for *Equal Opportunity* [77]). However, these metrics are not applicable in our task. The reason is that most of them are computed based on

the predicted labels, while a single training node can barely twist these predicted labels on test data [236, 180]. Consequently, the influence of a single training node on the model bias would be hard to capture. (2) *Computation Efficiency.* To compute the influence of each training node on the model bias, a natural way is to re-train the GNN on a new graph with this specific training node being deleted and observe how the exhibited model bias changes. However, such a re-training process is prohibitively expensive. (3) *Non-I.I.D. Characterization.* Graph data goes against the widely adopted i.i.d. assumption, as neighboring nodes are often dependent on each other [135, 228]. Therefore, when a specific node is deleted from the graph, all its neighbors could exert different influences on the model bias of GNN during training. Such complex dependencies bring obstacles towards the node influence analysis on model bias.

To tackle the above challenges, in this paper, we propose a novel framework named BIND (Biased traIning Node iDentification) to quantify and estimate the influence of each training node on the model bias of GNNs. Specifically, to handle the first challenge, we propose *Probabilistic Distribution Disparity* (PDD) as a principled strategy to quantify the model bias. PDD directly quantifies the exhibited bias in the GNN probabilistic output instead of the predicted labels. Therefore, PDD is with finer granularity and is more suitable for capturing the influence of each specific training node compared with traditional fairness metrics. To handle the second challenge, we propose an estimation algorithm for the node influence on model bias, which avoids the re-training process and thus achieves better efficiency. To tackle the third challenge, we also characterize the dependency between nodes based on the analysis of the training loss for GNNs. Finally, experiments on real-world datasets corroborate the effectiveness of BIND. Our contributions are mainly summarized as (1) **Problem Formulation.** We formulate a novel problem of interpreting the bias exhibited in GNNs through attributing to the influence of training nodes; (2) **Metric and Algorithm Design.** We propose a novel framework BIND to quantify and efficiently estimate the influence of each training node on the model bias of GNNs; (3) **Experimental Evaluation.** We perform comprehensive experiments on real-world datasets to evaluate the effectiveness of the proposed framework BIND.

3.1.2 Preliminaries

We first present the notations used in this paper. Then, we define the problem of interpreting GNN unfairness through quantifying the influence of each specific training node.

Notations. In this paper, matrices, vectors, and scalars are represented with bold uppercase letters (e.g., $\boldsymbol{A}$), bold lowercase letters (e.g., $\boldsymbol{x}$), and normal lowercase letters (e.g., n), respectively. We denote an input graph as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$, where $\mathcal{V} = \{v_1, ..., v_n\}$ denotes the node set, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the edge set, $\mathcal{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ is the node attribute vectors, and $\boldsymbol{x}_i$ $(1 \leq i \leq n)$ represents the attribute vector of node v_i. We denote $\mathcal{G}_{-i}$ as the new graph with node v_i being deleted from $\mathcal{G}$. Additionally, we employ $\mathcal{V}'$ $(\mathcal{V}' \subseteq \mathcal{V})$ to represent the training node set, where $|\mathcal{V}'| = m$. The nodes in graph $\mathcal{G}$ are mapped to the output space with a trained GNN $f_{\boldsymbol{W}}$, where $\boldsymbol{W}$ represents the learnable parameters of the GNN model. We denote the optimized parameters (i.e., the parameters after training) as $\hat{\boldsymbol{W}}$. In node classification, the probabilistic classification output for the n nodes is denoted as $\hat{\mathcal{Y}} = \{\hat{\boldsymbol{y}}_1, ..., \hat{\boldsymbol{y}}_n\}$, where $\hat{\boldsymbol{y}}_i \in \mathbb{R}^c$, and c is the number of classes. We use Y and S to denote the ground truth label and the sensitive attribute for nodes, respectively. For an L-layer

GNN $f_{\boldsymbol{W}}$, we define the subgraph up to L hops away centered on v_i as its computation graph (denoted as $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i, \mathcal{X}_i\}$). Here $\mathcal{V}_i, \mathcal{E}_i$, and $\mathcal{X}_i$ denote the set of nodes, edges, and node attributes in $\mathcal{G}_i$, respectively. It is worth noting that existing works have proven that $\mathcal{G}_i$ fully determines the information $f_{\boldsymbol{W}}$ utilizes to make the prediction of v_i [228]. For node v_i, we use $\mathcal{V}_i'$ to indicate the intersection between $\mathcal{V}_i$ and $\mathcal{V}'$, i.e., $\mathcal{V}_i' = \mathcal{V}_i \cap \mathcal{V}'$, which is the set of training nodes in $\mathcal{G}_i$.

Problem Statement. The problem of interpreting GNN unfairness is defined below.

PROBLEM 3.1.1. ***GNN Unfairness Interpretation.*** *Given the graph $\mathcal{G}$ and a GNN model $f_{\hat{\boldsymbol{W}}}$ trained based on $\mathcal{G}$, we define the problem of interpreting GNN unfairness as to quantify the influence of each training node to the unfairness exhibited in GNN predictions on the test set.*

3.1.3 Methodology

In this section, we first briefly introduce GNNs for the node classification task. Then, to tackle the challenge of *Influence Quantification*, we propose Probabilistic Distribution Disparity (PDD) to measure model bias and define node influence on the bias in a trained GNN. Furthermore, to tackle the challenge of *Computation Efficiency*, we design an algorithm to estimate the node influence on the model bias. Finally, we introduce how to characterize the dependency between nodes in influence estimation, which tackles the challenge of *Non-I.I.D. Characterization*.

3.1.3.1 GNNs in Node Classification

In the node classification task, GNNs take the input graph $\mathcal{G}$ and output a probabilistic output matrix $\hat{\boldsymbol{Y}}$, where the i-th row in $\hat{\boldsymbol{Y}}$ is $\hat{\boldsymbol{y}}_i$, i.e., the probabilistic prediction of a node's membership over all possible classes. Usually, there are multiple layers in GNNs, where the formulation of the l-th layer can be summarized as:

$$\boldsymbol{z}_i^{(l+1)} = \sigma\left(\mathrm{AGG}\left(\boldsymbol{z}_i^{(l)}, h\left(\left\{\boldsymbol{z}_j^{(l)} : v_j \in \mathcal{N}(v_i)\right\}\right)\right)\right). \tag{3.1}$$

Here $\boldsymbol{z}_i^{(l)}$ is the embedding of node i at the l-th layer; $\mathcal{N}(v_i)$ is the set of one-hop neighbors around v_i; $h(\cdot)$ is a function with learnable parameters; $\mathrm{AGG}(\cdot)$ and $\sigma(\cdot)$ denote the aggregation function (e.g., mean operator) and activation function (e.g., ReLU), respectively. Later on, a loss function $L_{\mathcal{V}'}$ (e.g., cross-entropy loss) defined on the set of training nodes $\mathcal{V}'$ is employed for GNN training.

3.1.3.2 Probabilistic Distribution Disparity

Traditional bias metrics such as Δ_{SP} for statistical parity and Δ_{EO} for equal opportunity are computed on the predicted class labels. However, a single training node can hardly twist these predicted labels [236, 180]. Hence the node-level contribution to model bias can barely be captured by traditional bias metrics. To capture the influence of a single training node on model bias, we propose Probabilistic Distribution Disparity (PDD) as a novel bias quantification strategy. PDD can be instantiated with different fairness notions to depict the model bias from different perspectives. Specifically, we assume the population is divided into different sensitive subgroups, i.e., demographic subgroups described by the sensitive attribute.

To achieve finer granularity, we define PDD as the Wasserstein-1 distance [107] between the probability distributions of a variable of interest in different sensitive subgroups. Compared with traditional fairness metrics, continuous changes brought by each specific training node are reflected in the measured distributions, and Wasserstein distance is theoretically more sensitive to the change of the measured distributions over other commonly used distribution distance metrics [5]. In addition, we note that the variable of interest depends on the chosen fairness notion in applications, and a larger value of PDD indicates a higher level of model bias. We introduce two instantiations of PDD based on two traditional fairness notions, including *Statistical Parity* [58] and *Equal Opportunity* [77]. Both notions are based on binary classification tasks and binary sensitive attributes (generalizations to non-binary cases can be found in the online version[1]). For example, Statistical Parity requires the probability of positive predictions to be the same across two sensitive subgroups, where the variable of interest is the GNN probabilistic output $\hat{\boldsymbol{y}}$. We use $\hat{\mathcal{Y}}^{(S=j)}$ to denote the set of the probabilistic predictions for test nodes whose sensitive attribute S equals to j ($j \in \{0, 1\}$). Let the distribution of the probabilistic predictions in $\hat{\mathcal{Y}}^{(S=0)}$ and $\hat{\mathcal{Y}}^{(S=1)}$ be $P_{\hat{\boldsymbol{y}}}^{(S=0)}$ and $P_{\hat{\boldsymbol{y}}}^{(S=1)}$, respectively. The PDD instantiated with statistical parity Γ_{SP} is

$$\Gamma_{SP} = \text{Wasserstein}_1(P_{\hat{\boldsymbol{y}}}^{(S=0)}, P_{\hat{\boldsymbol{y}}}^{(S=1)}), \tag{3.2}$$

where $\text{Wasserstein}_1(\cdot, \cdot)$ takes two distributions as input and outputs the Wasserstein-1 distance between them. Denote Y as the ground truth for node classification. Similarly, we can also instantiate PDD based on Equal Opportunity Γ_{EO} as

$$\Gamma_{EO} = \text{Wasserstein}_1(P_{\hat{\boldsymbol{y}}}^{(S=0,Y=1)}, P_{\hat{\boldsymbol{y}}}^{(S=1,Y=1)}). \tag{3.3}$$

$P_{\hat{\boldsymbol{y}}}^{(S=0,Y=1)}$ and $P_{\hat{\boldsymbol{y}}}^{(S=1,Y=1)}$ are model prediction distributions for nodes with $(S = 0, Y = 1)$ and $(S = 1, Y = 1)$, respectively. We then define node influence on model bias.

DEFINITION 3.1.1. ***Node Influence on Model Bias.*** *Let $f_{\hat{\boldsymbol{W}}}$ and $f_{\hat{\boldsymbol{W}}'}$ denote the GNN model trained on graph $\mathcal{G}$ and $\mathcal{G}_{-i}$ (i.e., $\mathcal{G}$ with node $v_i \in \mathcal{V}'$ being deleted), respectively. Let Γ_1 and Γ_2 be the Probabilistic Distribution Disparity value based on the output of $f_{\hat{\boldsymbol{W}}}$ and $f_{\hat{\boldsymbol{W}}'}$ for nodes in test set. We define $\Delta\Gamma = \Gamma_2 - \Gamma_1$ as the influence of node v_i on the model bias.*

The rationale behind this definition is to measure to what extent Γ changes if the GNN model is trained on a graph without v_i. Thus, $\Delta\Gamma$ depicts the influence of node v_i on the model bias. For both instantiations of Γ (i.e., Γ_{SP} and Γ_{EO}), if $\Delta\Gamma > 0$, deleting the training node v_i from $\mathcal{G}$ leads to a more unfair (or biased) GNN model. This indicates that node v_i contributes to improving the fairness level, i.e., v_i is helpful for fairness. Nevertheless, the above computation requires re-training the GNN to obtain the influence of each training node, which is too expensive if we want to compute the influence of all nodes in the training set. In Section 3.1.3.3, we introduce how to efficiently estimate $\Delta\Gamma$.

3.1.3.3 Node Influence on Model Bias Estimation

It is noteworthy that PDD is a function of $\hat{\boldsymbol{W}}$ for a trained GNN, as $\hat{\boldsymbol{W}}$ directly determines the probabilistic predictions for test nodes. Hence we first characterize how a training node in $\mathcal{G}$ influences $\hat{\boldsymbol{W}}$, followed by how this node influences PDD via applying the chain rule.

[1]See online version here https://ojs.aaai.org/index.php/AAAI/article/view/25905 for supplementary discussion and experimental results.

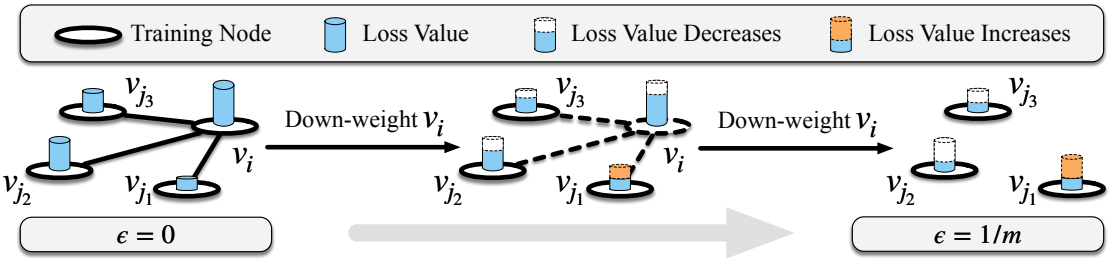


FIGURE 3.1. An illustration of how down-weighting node v_i influences the loss values of the training nodes in $\mathcal{G}_i$ (including v_i, v_{j_1}, v_{j_2}, and v_{j_3}). Scenarios from $\epsilon = 0$ to $\epsilon = 1/m$ are presented.

Formally, the optimal parameters $\hat{\boldsymbol{W}}$ minimize the objective function $L_{\mathcal{V}'}(\mathcal{G}, \boldsymbol{W})$ of the node classification task, so that:

$$\hat{\boldsymbol{W}} \overset{\text{def}}{=} \arg\min_{\boldsymbol{W}} L_{\mathcal{V}'}(\mathcal{G}, \boldsymbol{W}) = \arg\min_{\boldsymbol{W}} \frac{1}{m} \sum_{i=1}^{m} L_{v_i}\left(\mathcal{G}_i, \boldsymbol{W}\right).$$

Here $L_{v_i}\left(\mathcal{G}_i, \boldsymbol{W}\right)$ denotes the loss term associated with node v_i; $\mathcal{G}_i$ is the computation graph of v_i; m is the total number of training nodes. If a training node v_i is deleted from $\mathcal{G}$, the loss function will change and thus leads to a different $\hat{\boldsymbol{W}}$. We take v_i as an example to analyze the influence on $\hat{\boldsymbol{W}}$ after deleting a training node from $\mathcal{G}$. Traditionally, the existence of node v_i is considered as a binary state, which is either one (if v_i exists in $\mathcal{G}$) or zero (otherwise). But in our analysis, we treat it as a continuous variable to depict the intermediate states of the existence of v_i. Suppose that the existence of v_i is down-weighted in the training of a GNN on $\mathcal{G}$. This operation leads to two changes in the loss function: (1) the loss term associated with node v_i, i.e., $L_{v_i}\left(\mathcal{G}_i, \boldsymbol{W}\right)$, is down-weighted; (2) the loss terms associated with other training nodes in the computation graph of v_i would also be influenced. The reason is that these nodes could be affected by the information from node v_i during the message passing in GNNs [111, 228]. Based on the above analysis, we define $\hat{\boldsymbol{W}}_{\epsilon,v_i}$ as the optimal parameter that minimizes the loss function when node v_i is down-weighted as follows:

$$\hat{\boldsymbol{W}}_{\epsilon,v_i} \overset{\text{def}}{=} \arg\min_{\boldsymbol{W}} L_{\mathcal{V}'}(\mathcal{G}, \boldsymbol{W}) - \epsilon \left(L_{v_i}\left(\mathcal{G}_i, \boldsymbol{W}\right) + \tilde{L}_{\mathcal{V}'_i}(\mathcal{G}_i, \boldsymbol{W})\right), \tag{3.4}$$

where $\epsilon \in [0, 1/m]$ controls the scale of down-weighting v_i. An illustration in Fig. 3.1 shows how down-weighting v_i affects the loss values of training nodes in its computation graph. To formally characterize how node v_i influences $\hat{\boldsymbol{W}}$, we have Theorem 3.1.1 as follows (see proofs in the online version[2]).

THEOREM 3.1.1. *According to the optimization objective of $\hat{\boldsymbol{W}}_{\epsilon,v_i}$ in Eq. (3.4), we have*

$$\left.\frac{d\hat{\boldsymbol{W}}_{\epsilon,v_i}}{d\epsilon}\right|_{\epsilon=0} = \left(\frac{\partial^2 L_{\mathcal{V}'}(\mathcal{G}, \hat{\boldsymbol{W}})}{\partial \boldsymbol{W}^2}\right)^{-1} \cdot \left(\frac{\partial L_{v_i}\left(\mathcal{G}_i, \hat{\boldsymbol{W}}\right)}{\partial \boldsymbol{W}} + \frac{\partial \tilde{L}_{\mathcal{V}'_i}(\mathcal{G}_i, \hat{\boldsymbol{W}})}{\partial \boldsymbol{W}}\right). \tag{3.5}$$

Then, we characterize the influence of down-weighting node v_i on the value of PDD. We present Corollary 3.1.1 based on the chain rule as follows (see the proofs in the online version[3]).

[2]See online version for supplementary discussion and experimental results.

[3]See online version for all proofs.

COROLLARY 3.1.1. *Define the derivative of* Γ *w.r.t.* ϵ *at* $\epsilon = 0$ *as* $I_\Gamma(v_i)$. *According to Theorem 3.1.1, we have*

$$I_\Gamma(v_i) \overset{def}{=} \left.\frac{\partial \Gamma}{\partial \epsilon}\right|_{\epsilon=0} = \left(\frac{\partial \Gamma}{\partial \boldsymbol{W}}\right)^\top \left.\frac{d\hat{\boldsymbol{W}}_{\epsilon,v_i}}{d\epsilon}\right|_{\epsilon=0}. \tag{3.6}$$

With Corollary 3.1.1, we can estimate the value change of Γ when node v_i is down-weighted via

$$\Gamma_{\epsilon,v_i} - \Gamma_{0,v_i} = -\epsilon \cdot I_\Gamma(v_i) + o(\epsilon) \approx -\epsilon \cdot I_\Gamma(v_i) \tag{3.7}$$

according to the first-order Taylor expansion. Here Γ_{ϵ,v_i} and Γ_{0,v_i} are the PDD values after and before node v_i is down-weighted, respectively. To estimate the value change in Γ for a GNN trained on $\mathcal{G}_{-i}$, we introduce Theorem 3.1.2 as follows (see the proofs in the online version).

THEOREM 3.1.2. *Compared with the GNN trained on* $\mathcal{G}$, $\Delta\Gamma = \Gamma_{\frac{1}{m},v_i} - \Gamma_{0,v_i}$ *is equivalent to the value change in* Γ *when the GNN mode is trained on graph* $\mathcal{G}_{-i}$.

Theorem 3.1.2 enables us to directly compute the $\Delta\Gamma$ for an arbitrary training node v_i, which helps avoid the expensive re-training process. In the next section, we further define $\tilde{L}_{\mathcal{V}_i'}(\mathcal{G}_i, \hat{\boldsymbol{W}})$ and present an algorithm to efficiently estimate the node influence on model bias.

3.1.3.4 Non-I.I.D. Characterization

Generally, there are two types of dependencies between a training node v_i and other nodes in its computation graph $\mathcal{G}_i$, namely its dependency on other training nodes and its dependency on test nodes. The dependency between training nodes directly influences $\boldsymbol{W}$ during GNN training, and thus influences the probabilistic outcome of all test nodes. Hence it is critical to properly characterize the dependency between v_i and other training nodes. Specifically, we aim to characterize how the loss summation of all training nodes in $\mathcal{G}_i$ changes due to the existence of v_i. We denote the training nodes other than node v_i in $\mathcal{G}_i$ as $\mathcal{V}_i'\backslash\{v_i\}$. For any node $v_j \in \mathcal{V}_i'\backslash\{v_i\}$, we denote $\mathcal{G}_{j,-i}$ as the computation graph of node v_j with node v_i being deleted. $\tilde{L}_{\mathcal{V}_i}(\mathcal{G}_i, \hat{\boldsymbol{W}})$ is then formally given as

$$\tilde{L}_{\mathcal{V}_i}(\mathcal{G}_i, \hat{\boldsymbol{W}}) = \sum_{v_j \in \mathcal{V}_i'\backslash\{v_i\}} \left(L_{v_j}\left(\mathcal{G}_j, \hat{\boldsymbol{W}}\right) - L_{v_j}\left(\mathcal{G}_{j,-i}, \hat{\boldsymbol{W}}\right)\right). \tag{3.8}$$

The first term represents the summation of loss for nodes in $\mathcal{V}_i'\backslash\{v_i\}$ on $\mathcal{G}$, and the second term denotes the summation of loss for these nodes on $\mathcal{G}_{-i}$. In this regard, $\tilde{L}_{\mathcal{V}_i}(\mathcal{G}_i, \hat{\boldsymbol{W}})$ generally depicts to what extent the loss summation changes for nodes in $\mathcal{V}_i'\backslash\{v_i\}$ on graph $\mathcal{G}$ compared with $\mathcal{G}_{-i}$. If v_i is down-weighted by a certain degree, the change of the loss summation for nodes in $\mathcal{V}_i'\backslash\{v_i\}$ can be depicted by a linearly re-scaled $\tilde{L}_{\mathcal{V}_i}(\mathcal{G}_i, \hat{\boldsymbol{W}})$, as described in Eq. (3.4).

Additionally, there could also be dependencies between v_i and test nodes in $\mathcal{G}_i$, as v_i can influence the representations of its neighboring test nodes due to the information propagation mechanism in GNNs during inference. Such a dependency could also influence the value of PDD when v_i is deleted from $\mathcal{G}$. Correspondingly, we introduce the characterization of the dependency between v_i and test nodes. Specifically, we present an upper bound to depict the

Algorithm 1 Node Influence on Model Bias Estimation

Input: $\mathcal{G}$: the graph data; $f_{\hat{\boldsymbol{W}}}$: the trained GNN model; $\mathcal{V}'$: the set of training nodes;
Output: $\mathcal{I}_\Gamma = \{\Gamma_{\frac{1}{m},v_i} - \Gamma_{0,v_i} : v_i \in \mathcal{V}'\}$;
1: Initialize $\mathcal{I}_\Gamma = \varnothing$;
2: Compute $\{\frac{\partial \Gamma}{\partial \hat{\boldsymbol{W}}} : v_i \in \mathcal{V}'\}$ based on $f_{\hat{\boldsymbol{W}}}$;
3: **while** $v_i \in \mathcal{V}'$ **do**
4: Compute $\frac{d\hat{\boldsymbol{W}}_{\epsilon,v_i}}{d\epsilon}\Big|_{\epsilon=0}$ according to Eq. (3.5) and (3.8);
5: Compute $I_\Gamma(v_i)$ according to Eq. (3.6);
6: Compute $\Gamma_{\frac{1}{m},v_i} - \Gamma_{0,v_i}$ according to Eq. (3.7);
7: Append element $\Gamma_{\frac{1}{m},v_i} - \Gamma_{0,v_i}$ onto $\mathcal{I}_\Gamma$;
8: **end while**
9: **Return:** $\mathcal{I}_\Gamma$;

normalized change magnitude of the neighboring test nodes' representations when a training node v_i is deleted. Here the analysis is based on the prevalent GCN model [111], and can be easily generalized to other GNNs. Following widely adopted assumptions in [80, 221], we have Proposition 3.1.1 (see the proofs in the online version[4]).

PROPOSITION 3.1.1. *Denote the representations of node $v_j (v_j \in \mathcal{V}\backslash\mathcal{V}')$ based on $\mathcal{G}$ and $\mathcal{G}_{-i}$ as $\boldsymbol{z}_j$ and $\boldsymbol{z}_j^\star$, respectively. Define $h^{(j,i)}$ and $q^{(j,i)}$ as the distance from v_j to v_i and the number of all possible paths from v_j to v_i, respectively. Define the set of geometric mean node degrees of $q^{(j,i)}$ paths as $\mathcal{D} = \{d_1^{(j,i)}, ..., d_{q^{(j,i)}}^{(j,i)}\}$. Define $d_{min}^{(j,i)}$ as the minimum value of $\mathcal{D}$. Assume the norms of all node representations are the same. We then have $\|\boldsymbol{z}_j^\star - \boldsymbol{z}_j\|_2 / \|\boldsymbol{z}_j\|_2 \leq q^{(j,i)} / (d_{min}^{(j,i)})^{h^{(j,i)}}$.*

From Proposition 3.1.1, we observe that (1) deleting v_i exerts an upper-bounded impact on the representations of other test nodes in its computation graph; and (2) this upper-bound exponentially decays w.r.t. the distance between v_i and test nodes. Hence the dependency between v_i and test nodes has limited influence on Γ during inference when v_i is deleted from the graph. On the contrary, considering that the dependency between v_i and other training nodes directly influences $\hat{\boldsymbol{W}}$ and thus influences the inference results of all nodes, such a dependency should not be neglected. Consequently, we argue that it is reasonable to estimate the influence of each training node on Γ by only considering the dependency between training nodes. We present the algorithmic routine of $\Delta\Gamma$ estimation in Algorithm 2.

3.1.3.5 Complexity Analysis

To better understand the computational cost, here we analyze the time complexity of estimating $\Delta\Gamma$ according to Algorithm 2. We denote the number of parameters in $\boldsymbol{W}$ and the average number of training nodes in the computation graph of an arbitrary training node as t and $\bar{r}$, respectively. For each node v_i, the time complexity to compute $\partial L_{v_i}\left(\mathcal{G}_i, \hat{\boldsymbol{W}}\right)/\partial \boldsymbol{W}$ and $\partial \tilde{L}_{\mathcal{V}_i'}(\mathcal{G}_i, \hat{\boldsymbol{W}})/\partial \boldsymbol{W}$ is $O(t)$ and $O(\bar{r}t)$, respectively. Hence the time complexity is $O(m\bar{r}t)$ to traverse all training nodes. For the Hessian matrix inverse, we employ a widely-used estimation approach (see the online version for details[5]) with linear time complexity w.r.t t. Thus the time complexity of Eq. (3.5) and (3.8) is $O(m\bar{r}t)$. Additionally, the time complexity

[4]See online version for all proofs.
[5]See online version for supplementary discussion and experimental results.

of Eq. (3.6) and (3.7) is $O(mt)$ and $O(m)$, respectively. To summarize, the time complexity of Algorithm 2 is $O(m\bar{r}t)$. Considering that $\bar{r} \leq m$, the algorithm has a quadratic time complexity w.r.t. training node number. This verifies the time efficiency of our algorithm.

3.1.4 Experiments

We aim to answer the following research questions in experiments. **RQ1**: How efficient is BIND in estimating the influence of training nodes on the mode bias? **RQ2**: How well can BIND estimate the influence of training nodes on the model bias? **RQ3**: How well can we debias GNNs via deleting harmful training nodes based on our estimation? More details of experimental settings, supplementary experiments, and analysis are in the online version.

3.1.4.1 Experimental Setup

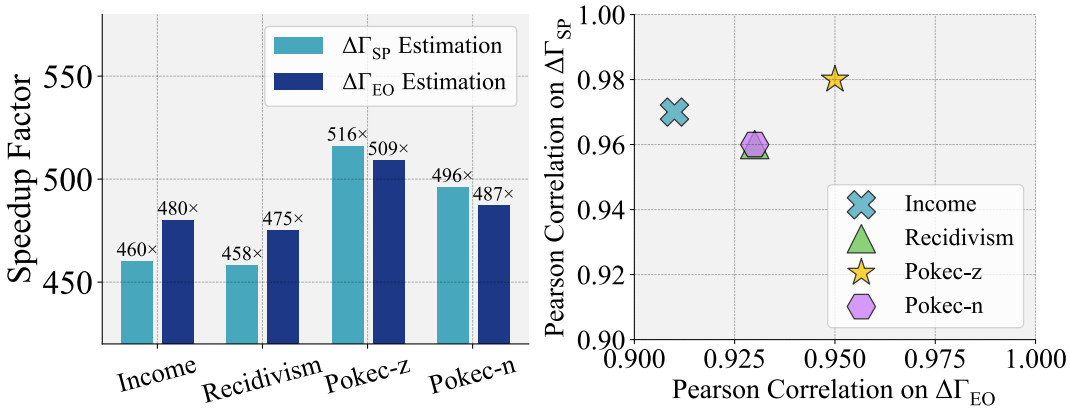
Downstream Task & Datasets. Here the downstream task is node classification. Four real-world datasets are adopted in our experiments, including *Income*, *Recidivism*, *Pokec-z*, and *Pokec-n*. Specifically, *Income* is collected from *Adult Data Set* [56]. Each individual is represented by a node, and we establish connections (i.e., edges) between individuals following a similar criterion adopted in [3]. The sensitive attribute is race, and the task is to classify whether the salary of a person is over \$50K per year or not. *Recidivism* is collected from [99]. A node represents a defendant released on bail, and defendants are connected based on their similarity. The sensitive attribute is race, and the task is to classify whether a defendant is on bail or not. *Pokec-z* and *Pokec-n* are collected from *Pokec*, which is a popular social network in Slovakia [182]. In both datasets, each user is a node, and each edge stands for the friendship relation between two users. The locating region of users is the sensitive attribute. The task is to classify the user working field. More details are in the online version.

Baselines & GNN Backbones. We compare our method with three state-of-the-art GNN debiasing baselines, namely FairGNN [37], NIFTY [3], and EDITS [47]. To perform GNN debiasing, FairGNN employs adversarial training to filter out the information of sensitive attributes from node embeddings; NIFTY maximizes the agreement between the predictions based on perturbed sensitive attributes and unperturbed ones; EDITS pre-processes the input graph data to be less biased via attribute and structural debiasing. We mainly present the results of using GCN [111] as the backbone GNN model, while experiments with other GNNs are discussed in the online version.

Evaluation Metrics. First, we employ running speedup factors to evaluate efficiency. Second, we use the widely adopted Pearson Correlation [113, 24] between the estimated and actual $\Delta\Gamma$ to evaluate the effectiveness of node influence estimation. Third, we adopt two traditional fairness metrics, namely Δ_{SP} (the metric for *Statistical Parity*) [58] and Δ_{EO} (the metric for *Equal Opportunity*) [77], to evaluate the effectiveness of debiasing GNNs via harmful nodes deletion. Additionally, the classification accuracy is also employed to evaluate the utility-fairness trade-off.

3.1.4.2 Efficiency of Node Influence Estimation

To answer RQ1, we evaluate the efficiency of $\Delta\Gamma$ estimation by comparing its running time with that of GNN re-training. The running time of GNN re-training is computed as follows.



(A) Evaluation of efficiency

(B) Evaluation of effectiveness

FIGURE 3.2. (A) Evaluation of efficiency: speedup factors of $\Delta\Gamma_{\text{SP}}$ and $\Delta\Gamma_{\text{EO}}$ estimation over GNN re-training. (B) Evaluation of effectiveness: correlation between estimated and actual $\Delta\Gamma_{\text{SP}}$ and $\Delta\Gamma_{\text{EO}}$.

We first delete the target node from the original input graph $\mathcal{G}$ and re-train the GCN to obtain $f_{\hat{\boldsymbol{W}}'}$. We then obtain $\Delta\Gamma$ based on the values of Γ given by $f_{\hat{\boldsymbol{W}}}$ and $f_{\hat{\boldsymbol{W}}'}$. The above running time is defined as the time cost of GNN re-training. The running time averaged across all training nodes is compared between GNN re-training and BIND, and we present the running speedup factors of BIND on the four real-world datasets in Fig. 3.2a. We observe that the running speedup factors are over 450× on all four real-world datasets, which corroborates the efficiency superiority of BIND in estimating the value of $\Delta\Gamma$. Additionally, we observe that the estimation on Pokec-z and Pokec-n datasets has higher speedup factors on both $\Delta\Gamma_{\text{SP}}$ and $\Delta\Gamma_{\text{EO}}$ compared with the other two datasets. A reason could be that nodes in Pokec-z and Pokec-n have lower average degrees (see online version for supplementary discussion and experimental results). This facilitates the computation of $\tilde{L}_{\mathcal{V}_i'}(\mathcal{G}_i, \hat{\boldsymbol{W}})$ (the term that characterizes non-i.i.d.) and corresponding derivatives.

3.1.4.3 Effectiveness of Node Influence Estimation

We now evaluate the effectiveness of $\Delta\Gamma$ estimation. It is worth noting that the numerical values of the estimated influence on model bias are small for most of the nodes (see online version for supplementary discussion and experimental results). Here we introduce a strategy to evaluate the estimation effectiveness across a wider value range of $\Delta\Gamma$. The basic intuition here is that we select node sets and evaluate how well their estimated $\Delta\Gamma$ summation aligns with the actual one. Specifically, we first follow the widely adopted routine [113, 24] to truncate the helpful and harmful nodes with top-ranked $\Delta\Gamma$ values. We then construct a series of node sets associated with the largest positive and negative estimated $\Delta\Gamma$ summations under different set size thresholds. The range of these thresholds is between zero and a maximum possible value (determined by the training set size). It is worth noting that only nodes with non-overlapping computation graphs are selected in constructing each node set. This ensures that these nodes result in an estimated $\Delta\Gamma$ equivalent to the summation of their estimated $\Delta\Gamma$[6]. We present the Pearson correlation of estimated $\Delta\Gamma_{\text{SP}}$ and $\Delta\Gamma_{\text{EO}}$ with the actual values on four

[6]See online version for supplementary discussion and experimental results.

TABLE 3.1. Comparison on GNN utility and bias mitigation between BIND and baselines. BIND 1% and BIND 10% denote the node deletion budget k being 1% and 10% of the training node set size, respectively. (↑) denotes the larger, the better; (↓) denotes the opposite. Numerical results are in percentages. Best ones and runner-ups are in bold and underline, respectively.

		Van. GCN	**FairGNN**	**NIFTY**	**EDITS**	**BIND 1%**	**BIND 10%**
Income	(↑) **Acc**	74.7 ± 1.4	69.1 ± 0.6	70.8 ± 0.9	68.3 ± 0.8	**75.2 ± 0.0**	71.7 ± 0.7
	(↓) Δ_{SP}	25.9 ± 1.9	**12.4 ± 4.7**	24.4 ± 1.6	24.0 ± 1.9	19.2 ± 0.6	14.7 ± 1.4
	(↓) Δ_{EO}	32.3 ± 0.8	**15.6 ± 6.8**	26.9 ± 3.7	24.9 ± 1.0	26.4 ± 0.4	16.2 ± 2.0
Recidivism	(↑) **Acc**	**89.8 ± 0.0**	89.7 ± 0.2	79.1 ± 0.9	89.6 ± 0.1	88.7 ± 0.0	88.5 ± 0.2
	(↓) Δ_{SP}	7.47 ± 0.2	7.31 ± 0.5	**1.82 ± 0.8**	5.02 ± 0.0	7.40 ± 0.0	6.57 ± 0.2
	(↓) Δ_{EO}	5.23 ± 0.1	5.17 ± 0.0	**1.28 ± 0.5**	2.89 ± 0.1	5.09 ± 0.1	4.23 ± 0.2
Pokec-z	(↑) **Acc**	63.2 ± 0.7	64.0 ± 0.7	**65.3 ± 0.2**	61.6 ± 0.9	63.5 ± 0.4	62.9 ± 0.4
	(↓) Δ_{SP}	7.32 ± 2.2	4.95 ± 0.8	2.34 ± 1.0	1.29 ± 0.8	6.75 ± 2.3	**1.02 ± 0.9**
	(↓) Δ_{EO}	7.60 ± 2.3	4.29 ± 0.7	**1.46 ± 1.3**	1.62 ± 1.6	5.41 ± 3.4	2.28 ± 1.5
Pokec-n	(↑) **Acc**	58.5 ± 0.8	60.3 ± 0.5	**61.1 ± 0.3**	56.8 ± 0.9	60.6 ± 0.8	58.8 ± 1.8
	(↓) Δ_{SP}	6.57 ± 2.6	5.30 ± 1.4	6.55 ± 0.7	2.75 ± 1.8	5.85 ± 2.0	**2.45 ± 0.9**
	(↓) Δ_{EO}	2.33 ± 0.5	1.67 ± 0.2	1.83 ± 0.6	2.24 ± 1.5	**1.15 ± 0.7**	2.22 ± 1.6

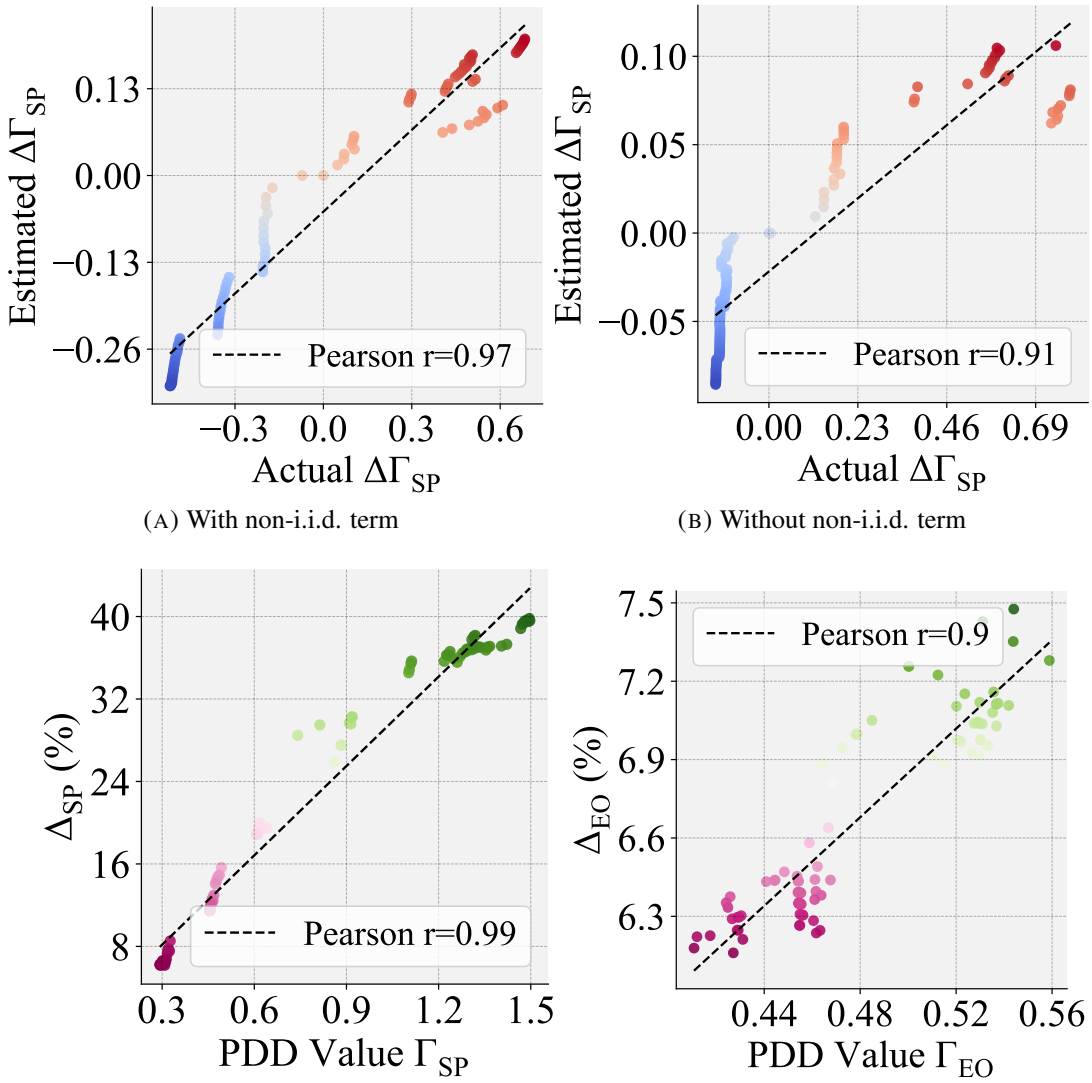
datasets in Fig. 3.2b. It is worth noting that achieving an exact linear correlation (i.e., Pearson correlation equals one) between the estimated and actual $\Delta\Gamma$ is almost impossible, since we only employ the first-order Taylor expansion in our estimation for $\Delta\Gamma$. From Fig. 3.2b, we observe that the estimation achieves Pearson correlation values over 0.9 on both Γ_{SP} and Γ_{EO} across all datasets. Such consistencies between estimated and actual values verify the effectiveness of BIND.

Additionally, to understand how the non-i.i.d. characterization benefits the estimation, we also estimate $\Delta\Gamma$ with BIND after the non-i.i.d. characterization being disabled (i.e., setting the $\tilde{L}_{\mathcal{V}_i'}(\mathcal{G}_i, \boldsymbol{W})$ term in Eq. (3.4) as 0). We present the estimated $\Delta\Gamma$ v.s. actual $\Delta\Gamma$ on Income dataset with non-i.i.d. characterization being enabled and disabled in Fig. 3.3a and 3.3b, respectively. We observe the correlation decreases between the estimated and actual $\Delta\Gamma$ after the non-i.i.d. characterization is disabled. Such a decrease is also observed on other datasets in terms of both statistical parity and equal opportunity. Such an observation verifies the contribution of non-i.i.d. characterization to the estimation of $\Delta\Gamma$.

Finally, we evaluate how well the values of the proposed PDD matches the values of traditional fairness metrics. We collect the value pairs of (Δ_{SP}, Γ_{SP}) and (Δ_{EO}, Γ_{EO}) during the GNN re-training process. The values of Δ_{SP} v.s. actual Γ_{SP} are presented in Fig. 3.3c, and the values of Δ_{EO} v.s. actual Γ_{EO} are shown in Fig. 3.3d. We observe a satisfying match between Γ and traditional metrics, which corroborates that PDD is a valid indicator of the fairness level depicted by traditional fairness metrics.

3.1.4.4 Debiasing via Harmful Nodes Deletion

In this subsection, we demonstrate how BIND could be employed for GNN debiasing. The basic intuition here is to identify and delete those harmful nodes according to the estimated node influence on model bias, and evaluate whether GNNs can be debiased when they are trained on this new graph. Specifically, we set $\Gamma = \lambda\Gamma_{\text{SP}} + (1-\lambda)\Gamma_{\text{EO}}$ and estimate the



(A) With non-i.i.d. term

(B) Without non-i.i.d. term

(C) Δ_{SP} v.s. Γ_{SP} (Income)

(D) Δ_{EO} v.s. Γ_{EO} (Recid.)

FIGURE 3.3. In (a) and (b), we compare the estimation effectiveness of $\Delta\Gamma_{SP}$ with and without characterizing non-i.i.d.; in (c) and (d), we present the consistency between Γ and traditional fairness metrics (Δ_{SP} for statistical parity and Δ_{EO} for equal opportunity) under different node deletion budgets.

node influence on Γ to consider both statistical parity and equal opportunity. We then set a budget k, and follow the strategy adopted in Section 3.1.4.3 to select and delete a set of training nodes with the largest positive influence summation on Γ under this budget. We set $\lambda = 0.5$ to assign statistical parity and equal opportunity the same weight, and perform experiments with k being 1% (denoted as BIND 1%) and 10% (denoted as BIND 10%) of the total number of training nodes. We present the results on the four adopted datasets in Table 3.1. The following observations are made: (1) compared with other baselines, BIND achieves competitive performance (i.e., lower values) on both Δ_{SP} and Δ_{EO}. Hence training GNNs on a new graph after deleting harmful nodes (to fairness) is an effective approach for GNN debiasing; (2) there is no obvious performance decrease on the model utility of BIND

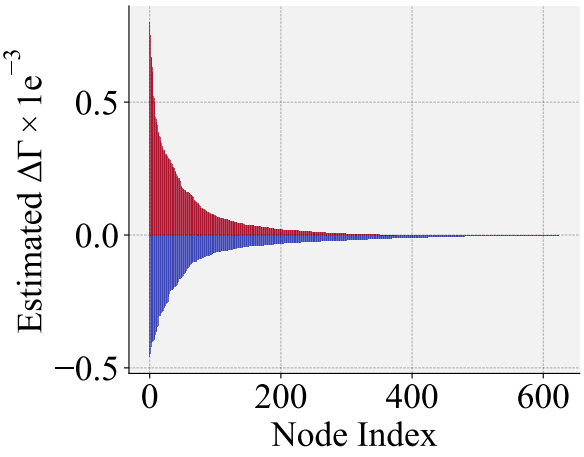
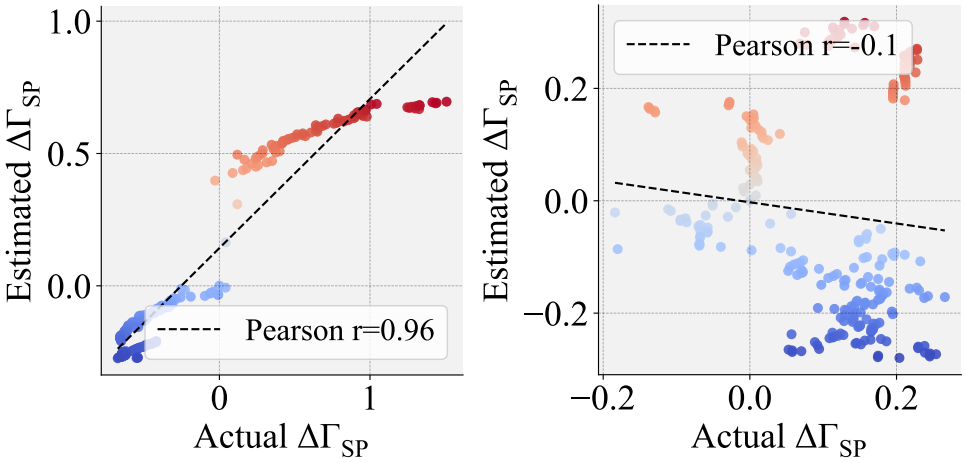


FIGURE 3.4. The estimated $\Delta\Gamma_{\text{SP}}$ values follow a long-tail distribution. Here, the color red and blue represent those helpful and harmful nodes, respectively.



(A) $\Delta\Gamma_{\text{SP}}$ on GIN (B) Shuffling on GCN

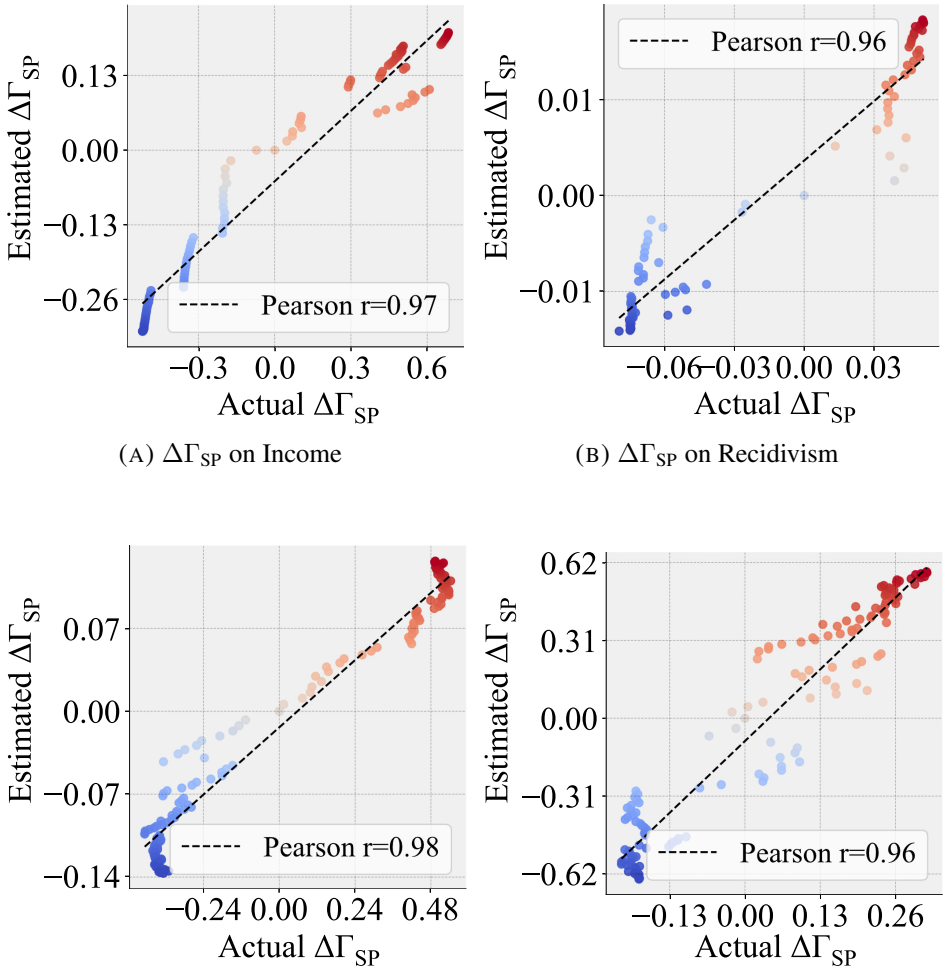
FIGURE 3.5. In (a), the estimated $\Delta\Gamma_{\text{SP}}$ v.s. the actual $\Delta\Gamma_{\text{SP}}$ based on a GIN model and Income dataset is presented. In (b), we present the estimated $\Delta\Gamma_{\text{SP}}$ v.s. the actual $\Delta\Gamma_{\text{SP}}$ after randomly shuffling the order of estimated $\Delta\Gamma_{\text{SP}}$ values for the training nodes. The correlation value decrease further validates the effectiveness of estimation.

compared with other baselines. We thus argue that deleting harmful nodes can also lead to a satisfying fairness-utility trade-off.

3.1.5 Supplementary Discussion

Distribution of Estimated $\Delta\Gamma$. We present the estimated $\Delta\Gamma_{\text{SP}}$ on Recidivism dataset in Fig. 3.4. Generally, the values of the estimated node influence on bias follow a long-tail distribution, i.e., only a small amount of nodes have large positive/negative influence values on the model bias. Similar phenomena can also be observed on other datasets.

Effectiveness of $\Delta\Gamma$ Estimation. We follow the same strategy introduced in the Effectiveness of Node Influence Estimation section to obtain the estimated $\Delta\Gamma_{\text{SP}}$ values and their



(A) $\Delta\Gamma_{SP}$ on Income

(B) $\Delta\Gamma_{SP}$ on Recidivism

(C) $\Delta\Gamma_{SP}$ on Pokec-z

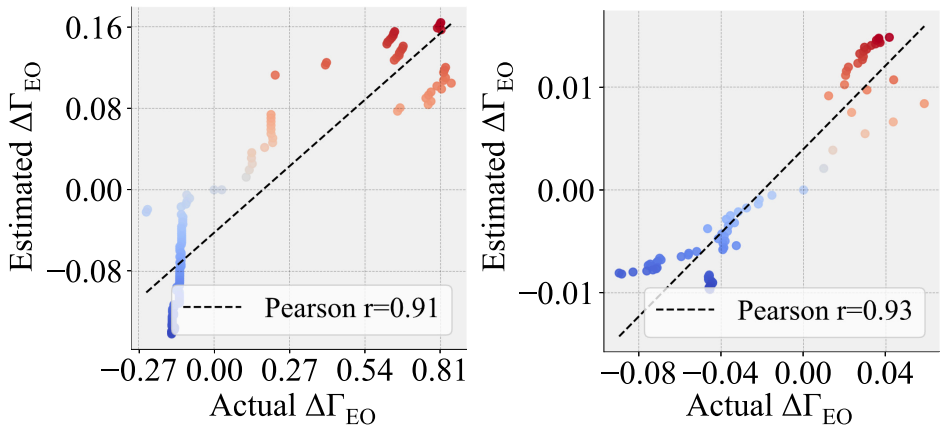
(D) $\Delta\Gamma_{SP}$ on Pokec-n

FIGURE 3.6. Estimated $\Delta\Gamma_{SP}$ v.s. actual $\Delta\Gamma_{SP}$ on four real-world datasets are presented for effectiveness analysis of $\Delta\Gamma$ estimation based on GCN. Helpful data points (marked in red) are with positive estimated $\Delta\Gamma_{SP}$ values, while harmful ones (marked in blue) are with negative estimated $\Delta\Gamma_{SP}$ values.

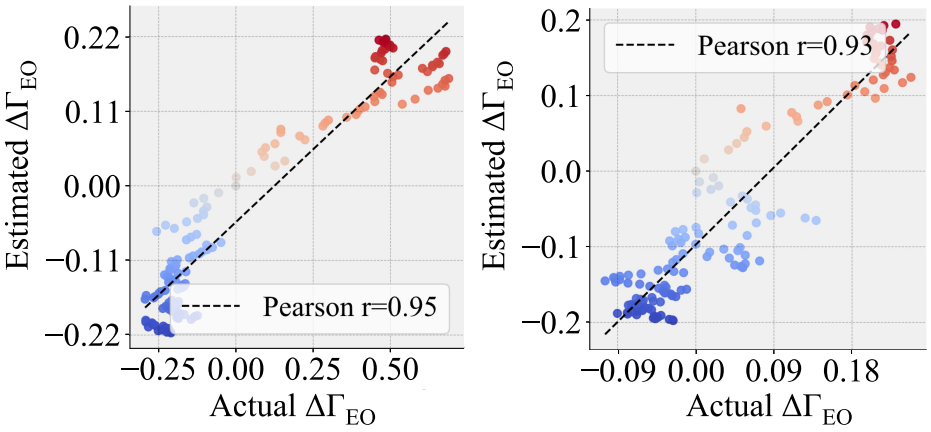
corresponding actual $\Delta\Gamma_{SP}$ values. We present the estimated $\Delta\Gamma_{SP}$ v.s. the actual $\Delta\Gamma_{SP}$ on the four datasets in Fig. 3.6a, 3.6b, 3.6c, and 3.6d, respectively; we also present the estimated $\Delta\Gamma_{EO}$ v.s. the actual $\Delta\Gamma_{EO}$ on the four datasets in Fig. 3.7a, 3.7b, 3.7c, and 3.7d, respectively. Experiments are carried out based on a random seed of 42. We draw the conclusion that on both $\Delta\Gamma_{SP}$ and $\Delta\Gamma_{EO}$, our estimation results show a satisfying match with the actual values across the four adopted datasets. This validates the effectiveness of $\Delta\Gamma$ estimation.

Generalization of $\Delta\Gamma$ Estimation to Different GNNs. We then test the generalization ability of our proposed estimation algorithm to different GNNs. Specifically, we present the estimated $\Delta\Gamma_{SP}$ v.s. the actual $\Delta\Gamma_{SP}$ based on a trained GIN model [222] and Income dataset in Fig. 3.5a. We have the observation that the estimated $\Delta\Gamma_{SP}$ also shows a satisfying match with the actual values based on the GIN model, which validates the generalization ability of the proposed estimation algorithm to GNNs other than GCN.

Shuffled Node Influence v.s. Actual PDD Difference. To further validate the effectiveness of the proposed estimation algorithm, we first perform node influence on bias estimation



(A) $\Delta\Gamma_{EO}$ on Income (B) $\Delta\Gamma_{EO}$ on Recidivism



(C) $\Delta\Gamma_{EO}$ on Pokec-z (D) $\Delta\Gamma_{EO}$ on Pokec-n

FIGURE 3.7. Estimated $\Delta\Gamma_{EO}$ v.s. actual $\Delta\Gamma_{EO}$ on four real-world datasets are presented for effectiveness analysis of $\Delta\Gamma_{EO}$ estimation based on GCN. Helpful data points (marked in red) are with positive estimated $\Delta\Gamma$ values, while harmful ones (marked in blue) are with negative estimated $\Delta\Gamma_{EO}$ values.

based on a trained GCN model and Income dataset. Then, we randomly shuffle the estimated influence values (i.e., estimated $\Delta\Gamma_{SP}$) for the training node set. This operation leads to a mismatch between training node indices and estimated $\Delta\Gamma_{SP}$. We follow the same strategy introduced in the Effectiveness of Node Influence Estimation section to obtain the estimated $\Delta\Gamma_{SP}$ values and their corresponding actual $\Delta\Gamma_{SP}$ values. We present the estimated $\Delta\Gamma_{SP}$ v.s. actual $\Delta\Gamma_{SP}$ in Fig. 3.5b. A decrease in Pearson correlation value is observed compared with those presented in Fig. 3.6. This observation further corroborates the effectiveness of the proposed estimation algorithm.

3.1.6 Related Work

Graph Neural Networks. GNNs can be divided into two mainstreams, including spectral-based and spatial-based ones [214, 245]. Specifically, spectral GNNs inherit the insights from Convolutional Neural Networks (CNNs) [15], and followed by many works [43, 122, 111]. Their goal is to design graph filters to extract task-related information from the input

graphs [32]. Differently, spatial GNNs design message-passing mechanisms in the spatial domain to extract information from each node's neighbors [214, 245]. Various aggregation strategies contribute to different tasks [188, 222, 181, 153].

Algorithmic Fairness. Algorithmic fairness can be defined from different perspectives [155, 134, 55, 22, 35, 142], where *Group Fairness* and *Individual Fairness* are two popular notions [58]. Generally, group fairness enforces similar statistics (e.g., positive prediction rate in binary classification tasks) across different demographic subgroups [58]. Typically, these demographic subgroups are described by certain sensitive attributes, such as gender, race, and religion. Individual fairness argues for similar outputs for similar individuals [58]. Algorithmic fairness can be considered in different stages of learning pipelines, including pre-processing [47], in-processing [46, 120, 38], and post-processing [104]. Particularly, re-weighting training samples to mitigate model bias is a popular fairness-enhancing method during in-processing stage [196, 76, 224, 90, 156]. However, most of these methods only yield a set of weights for training samples to mitigate bias [224, 196], while to what extent each sample influences the exhibited bias is still unclear. Different from them, this work aims to understand the influence of each training node on model bias. To the best of our knowledge, this is a first-of-its-kind study. Moreover, most of existing methods based on re-weighting training samples are developed under the IID assumption. However, in this paper, we also analyze the non-IID characteristic between nodes to understand how each training node influences model bias.

Interpretation of Deep Learning Models. Deep learning models have huge parameter size and high complexity [16, 168, 67, 218]. To make these models more trustworthy and controllable, many studies have been devoted to improving their transparency [67]. Generally, these works are divided into transparency design and post-hoc explanation [218]. The basic goal of transparency design is to understand the model in terms of model structure [127, 235] and training algorithms [158], while post-hoc explanation aims to explain specific prediction results via visualization [44] and explanatory examples [25]. In the realm of learning on graphs, some existing works aim to interpret GNNs [228, 132, 230], and they mainly focus on understanding the utility (e.g., node classification accuracy) of GNNs on the test set. Our work is different from them in two aspects: (1) we focus on interpreting the model bias instead of the utility for GNNs; (2) we aim to understand the model bias via attributing to the training set instead of only focusing on the test set.

3.1.7 Conclusion

In this paper, we study a novel problem of characterizing how each training node influences the bias exhibited in a trained GNN. We first propose a strategy named Probabilistic Distribution Disparity (PDD), which can be instantiated with different existing fairness notions, to quantify the node influence on the model bias. We then propose a novel framework named BIND to achieve an efficient influence estimation for each training node. We also develop a node deletion strategy to achieve GNN debiasing based on influence estimation. Extensive experiments verify (1) the consistency between the proposed PDD and traditional fairness metrics; (2) the efficiency and effectiveness of the influence estimation algorithm; and (3) the performance of the proposed strategy on GNN debiasing. We leave interpreting how the unfairness arises in other graph learning tasks as future works.

3.2 On Structural Explanation of Bias in Graph Neural Networks

3.2.1 Introduction

Graph Neural Networks (GNNs) have shown satisfying performance in various real-world applications, e.g., online recommendation [211], chemical reaction prediction [45], and complex physics simulation [169], to name a few. The success of GNNs is generally attributed to their message-passing mechanism [214, 245, 200]. Such a mechanism enables GNNs to capture the correlation between any node and its neighbors in a localized subgraph (i.e., the computation graph of the node [228]), which helps to extract information from both node attributes and network structure for node embedding learning [74]. Despite the remarkable success, most of the existing GNNs do not have fairness consideration [46, 47, 37, 105, 50, 138]. Consequently, GNN predictions could exhibit discrimination (i.e., bias) towards specific demographic subgroups that are described by sensitive features, e.g., age, gender, and race. Such discrimination has become one of the most critical societal concerns when GNNs are deployed in high-stake decision-making scenarios [103].

There is a rich body of literature on alleviating the bias of GNNs. Generally, these works aim to decouple the learned node embeddings from sensitive features [37, 47, 125, 179, 201]. However, they cannot provide explanations on how bias arises in GNNs. In fact, it is worth noting that in various high-stake decision-making scenarios, we not only need to alleviate bias in GNNs, but also need to understand how bias is introduced to the prediction of each individual data instance (e.g., a node in a graph). Such instance-level understanding is critical for the safe deployment of GNNs in decision-critical applications [228]. For example, GNNs have demonstrated superior performance in many financial applications, such as loan approval prediction for bank clients [193, 216]. In this scenario, different clients form a graph based on their transaction interactions, and the records of clients form their features. Here, the goal is to predict whether a client will be approved for a loan, and such a problem can be formulated as a node classification task that can be solved by GNNs. However, GNNs could lead to undesired discrimination against clients from certain demographic subgroups (e.g., rejecting a loan request mainly because the applicant belongs to an underprivileged group). In this example, understanding how bias is introduced to the prediction of each individual client enables bank managers to scrutinize each specific loan decision and take proactive actions to improve the algorithm and reduce potential discrimination.

In fact, biased GNN predictions can be attributed to a variety of factors. Among them, biased network structure has shown to be a critical source [125, 47, 179]. Additionally, bias in the network structure could be amplified by the core operation of GNNs – the *message-passing* mechanism [47]. Therefore, understanding which part of the network structure leads to biased GNN predictions for each node is vitally important. Towards this goal, we aim to provide an instance-level (i.e., node-level) structural explanation of bias in GNN predictions. More specifically, for any node in an input network for GNNs, we aim to understand and explain how different edges in its computation graph contribute to the level of bias for

its prediction[7]. Nevertheless, it remains a daunting task. Essentially, we mainly face the following three challenges: (1) **Fairness Notion Gap**: *how to measure the level of bias for the GNN prediction at the instance level*? For each node, understanding how the edges in its computation graph make its prediction biased requires a principled bias metric at the instance level. However, most of the existing bias metrics are defined over the whole population or the sub-population [58, 77], thus they cannot be directly grafted to our studied problem. In this regard, it is crucial to design a bias metric that can quantify the level of bias for the GNN prediction at the instance level. (2) **Usability Gap**: *is a single bias explainer sufficient*? It should be noted that our ultimate goal goes beyond explaining bias as we also aim to achieve fairer GNNs, which provide better model usability and enable ethical decision-making. Consequently, it is also critical to explain which edges in a node's computational graph contribute more to the fairness level of its prediction. However, edges that introduce the least bias cannot be simply regarded as the edges that maximally contribute to the fairness level of the prediction. This is because edges that introduce the least bias could also be those prediction-irrelevant edges — such edges could barely contribute any information to the GNN prediction. Therefore, only explaining how each edge in a computational graph contributes to the exhibited node-level bias is not sufficient. (3) **Faithfulness Gap**: *how to obtain bias (fairness) explanations that are faithful to the GNN prediction*? To ensure the obtained explanations reflect the true reasoning results based on the given GNN model, most existing works on the instance-level GNN explanation obtain explanations that encode as much critical information as possible for a given GNN prediction [228, 191, 132]. In this way, the obtained explanations are considered to be faithful to the given GNN model, as they generally reflect the critical information the GNN utilized to make the given prediction. Similarly, when explaining how the bias or the fairness level of the GNN prediction is achieved, we are also supposed to identify the critical information the GNN utilized to achieve such a level of bias or fairness for the given prediction.

As an attempt to tackle the challenges above, in this paper, we propose a principled framework named REFEREE (stRuctural Explanation oF biasEs in gRaph nEural nEtworks) for post-hoc explanation of bias in GNNs. Specifically, towards the goal of obtaining instance-level structural explanations of bias, we formulate a novel research problem of *Structural Explanation of Node-Level Bias in GNNs*. To tackle the first challenge, we propose a novel fairness notion together with the corresponding metric to measure the level of bias for a specific node in terms of GNN prediction. To tackle the second challenge, we design two explainers in the proposed framework REFEREE, namely bias explainer and fairness explainer. In any given computation graph, they are able to identify edges that maximally account for the exhibited bias in the prediction and edges that maximally contribute to the fairness level of the prediction, respectively. To tackle the third challenge, we design a constraint to enforce the faithfulness for the identified explanations, which can be incorporated into a unified objective function for the proposed framework. In this way, apart from the goal of explaining the exhibited bias and identifying edges that help with fairness, such a unified objective function also enforces the identified explanations to be faithful to the given GNN prediction. To better differentiate these two types of edges, the two explainers are designed to work in a contrastive manner. Finally, we evaluate the effectiveness of REFEREE on multiple real-world network

[7]Here, we only consider the edges in its corresponding computation graph. This is because the computation graph of a node fully encodes all information that GNN models leverage to generate its prediction [228].

datasets. The main contributions of this paper are as follows. (1) ***Problem Formulation.*** We formulate and study a novel problem of structural explanation of biases in GNNs given any instance-level GNN prediction. (2) ***Metric and Algorithmic Design.*** We propose a novel metric to measure how biased the GNN outcome prediction of a node is. We then propose a novel explanation framework named REFEREE to provide explanations on both fairness and bias, and maintain faithfulness to the given prediction. (3) ***Experimental Evaluations.*** We perform experimental evaluations on various real-world networks. Extensive experiments demonstrate the effectiveness of REFEREE and its superiority over other alternatives.

3.2.2 Problem Definition

In this section, we first present the notations used in this paper and some preliminaries. We then formulate a novel problem of *Structural Explanation of Bias in GNNs*.

Notations. We use bold uppercase letters (e.g., $\mathbf{A}$), bold lowercase letters (e.g., $\mathbf{x}$), and normal uppercase letters (e.g., N) to represent matrices, vectors, and scalars, respectively. Uppercase letters in math calligraphy font (e.g., $\mathcal{V}$) represent sets. The k-th entry of a vector, e.g., $\mathbf{x}$, is represented as $\mathbf{x}[k]$. For any number, $|\cdot|$ is the absolute value operator; for any set, $|\cdot|$ outputs its cardinality.

Preliminaries. We denote an attributed network as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$, where $\mathcal{V} = \{v_1, ..., v_N\}$ represents the set of N nodes; $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of all edges; $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ is the set of node attribute vectors. A trained GNN model $f_{\mathbf{\Theta}}$ maps each node to the outcome space, where $\mathbf{\Theta}$ denotes the parameters of the GNN model. Without loss of generality, we consider node classification as the downstream task. The GNN outcome for N nodes can be given as $\hat{\mathcal{Y}} = \{\hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_i, ..., \hat{\mathbf{y}}_N\}$, where $\hat{\mathbf{y}}_i \in \mathbb{R}^C$. Here C is the number of classes for node classification, and each dimension in $\hat{\mathbf{y}}_i$ represents the probability of the node belonging to the corresponding class. Based on $\hat{\mathcal{Y}}$, the predicted label set by GNN for these N nodes is denoted by $\{\hat{Y}_1, ..., \hat{Y}_i, ..., \hat{Y}_N\}$. Here $\hat{Y}_i$ is determined by the highest predicted probability across all C classes given by $\hat{\mathbf{y}}_i$. For GNN explanation, we consider the most widely studied instance-level explanation problem in this paper, i.e., we aim to explain the given prediction of a node based on its computation graph [228, 191, 132]. At the instance level, the explanations can be provided from different perspectives. Here we focus on the structural explanation, i.e., the explanation is given as an edge set $\tilde{\mathcal{E}}_i$ by any GNN explanation model $h_{\mathbf{\Phi}}$. Specifically, given a specific node v_i, its computation graph $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i, \mathcal{X}_i\}$ (i.e., the L-hop subgraph centered on node v_i [228], where L is the total layer number of the studied GNN), and the corresponding outcome $\hat{\mathbf{y}}_i$, the GNN structural explanation model $h_{\mathbf{\Phi}}$ with parameter $\mathbf{\Phi}$ identifies an explanation as an edge set $\tilde{\mathcal{E}}_i$ corresponding to the outcome $\hat{\mathbf{y}}_i$. $\tilde{\mathcal{E}}_i$ is identified through learning a weighted mask matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}_i| \times |\mathcal{V}_i|}$ that indicates the importance score of each edge in $\mathcal{E}_i$. Edges in $\tilde{\mathcal{E}}_i$ are selected from $\mathcal{E}_i$ based on such importance score. We denote the computation graph with the identified edge set $\tilde{\mathcal{E}}_i$ as a new subgraph $\tilde{\mathcal{G}}_i = \{\mathcal{V}_i, \tilde{\mathcal{E}}_i, \mathcal{X}_i\}$. Based on the new subgraph $\tilde{\mathcal{G}}_i$ with the identified edge set $\tilde{\mathcal{E}}_i$, the given GNN yields a different probabilistic outcome $\tilde{\mathbf{y}}_i = f_{\mathbf{\Theta}}(\tilde{\mathcal{G}}_i)$ compared with the vanilla outcome $\hat{\mathbf{y}}_i$.

Based on the above notations and preliminaries, we formulate the problem of *Structural Explanation of Bias in GNNs* as follows.

PROBLEM 3.2.1. ***Structural Explanation of Node-Level Bias in GNNs.*** *Given a trained GNN $f_{\mathbf{\Theta}}$, a node v_i to be explained, and its computation graph $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i, \mathcal{X}_i\}$, our goal is*

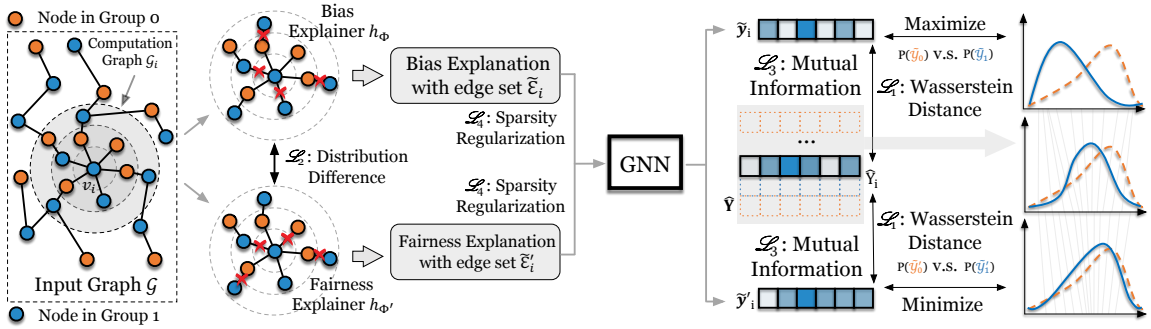


FIGURE 3.8. Framework structure of REFEREE: the edges in the edge set given by Bias Explainer maximally account for the node-level bias, while the edges in the edge set given by Fairness Explainer maximally alleviates the node-level bias.

to: (1) identify edges that are faithful to the prediction of v_i (based on f_Θ) and maximally account for the bias exhibited in the GNN outcome of v_i; (2) identify edges that are faithful to the prediction of v_i (based on f_Θ) and maximally contribute to the fairness level of the GNN outcome of v_i.

Intuitively, Problem 3.2.1 aims to identify two edge sets as two structural explanations: the bias explanation that accounts for the exhibited bias, and the fairness explanation that contributes to the fairness level of the given prediction. From the perspective of usability, the first explanation aims to identify edges that introduce the most bias to the instance-level GNN prediction, while the second explanation aims to identify edges that maximally contribute to the fairness level of the GNN prediction for any given node.

3.2.3 The Proposed Framework

In this section, we first present a principled metric to quantify the node-level bias for any given GNN prediction. Then we provide an overview of REFEREE, which is the proposed bias explanation framework for GNNs. Finally, we design a unified objective function for the proposed bias explanation framework REFEREE.

3.2.3.1 Node-Level Bias Modeling

To tackle the challenge of *Fairness Notion Gap*, here we aim to formulate a novel metric to quantify the bias for the node-level GNN prediction. Here we propose to formulate such a bias metric in the probabilistic outcome space of GNN predictions. The reason is that the information about the exhibited bias in the node-level prediction could be lost when the probabilistic outcomes are transformed into discrete predicted labels. In this regard, a bias metric based on the probabilistic outcome can better reflect the exhibited bias in the node-level predictions. This is also in align with some existing bias measures [37, 47, 63]. However, although these existing bias metrics are defined in the probabilistic outcome space, they can only measure the level of bias for the predictions over the whole population, and thus cannot be directly grafted to our problem.

We introduce the rationale of our proposed bias metric for node-level GNN predictions as follows. Intuitively, by measuring how much a node's outcome contributes to the overall

bias on the whole population, we can have a better understanding of the bias level of this node's outcome. More specifically, assume the nodes can be divided into two sensitive subgroups based on the values of their sensitive features[8]. The GNN outcome of nodes in the two sensitive subgroups forms two distributions, where the distance between the two distributions generally reflects the overall bias [63, 47]. For any specific node, if we change the probabilistic outcome of this node, the distribution distance between the outcome sets of the two sensitive subgroups will also change accordingly. Ideally, if the outcome of a node has no contribution to the outcome distribution distance between the two sensitive subgroups, then the distribution distance cannot be further reduced no matter how the outcome of this node is changed. In other words, a node that does not contribute to the overall bias should have an outcome with which the outcome distribution distance between the two sensitive subgroups is minimized. Meanwhile, we can also employ the potential distribution distance reduction to measure the contribution of a node's outcome to the overall bias. Based on such intuition, we then define *Node-Level Bias in GNNs* as follows.

DEFINITION 3.2.1. ***Node-Level Bias in GNNs.*** *Denote a probabilistic GNN outcome set as $\hat{\mathcal{Y}}$. Divide $\hat{\mathcal{Y}}$ into $\hat{\mathcal{Y}}_0$ and $\hat{\mathcal{Y}}_1$ as the outcome sets of the two demographic subgroups based on the sensitive feature. Denote D as the distance between the distributions of $\hat{\mathcal{Y}}_0$ and $\hat{\mathcal{Y}}_1$. For node v_i, denote $D_{min}(i)$ as the minimum distance between the distributions of $\hat{\mathcal{Y}}_0$ and $\hat{\mathcal{Y}}_1$ by changing the value of $\hat{\mathbf{y}}_i \in \hat{\mathcal{Y}}$ while maintaining $\sum_{k=1}^{C} \hat{\mathbf{y}}_i[k] = 1$. We define $B_i = D - D_{min}(i)$ as the node-level bias of node v_i for the GNN prediction.*

Definition 1 introduces how to measure the bias exhibited in the node-level prediction given a trained GNN. Clearly, the minimum value of B_i is 0, i.e., if no change on the value of $\hat{\mathbf{y}}_i$ can be found to further reduce the distance between the distribution of $\hat{\mathcal{Y}}_0$ and $\hat{\mathcal{Y}}_1$, we say that node v_i does not exhibit any node-level bias in the GNN outcome. In this paper, we adopt the Wasserstein distance as the metric for distribution distance measurement, considering its superior sensitivity over other distance metrics [5]. We will validate the consistency between Definition 1 and traditional fairness notions (e.g., *Statistical Parity* and *Equal Opportunity*) in Section 3.2.4.8.

3.2.3.2 Overview of Proposed Framework

Here we present an overview of the proposed bias explanation framework for node-level GNN predictions. In particular, to tackle the challenge of *Usability Gap*, REFEREE is designed with two different explainers, i.e., a bias explainer $h_{\mathbf{\Phi}}$ and a fairness explainer $h_{\mathbf{\Phi}'}$. The two explainers aim to identify two different edge sets in the given computation graph as two structural explanations, i.e., the bias explanation and the fairness explanation. The two explanations are learned in a contrastive manner, in which way edges that account for different explanations can be better distinguished. The basic goal of the bias explainer is to identify the edges that maximally account for the exhibited node-level bias, while the goal of the fairness explainer is to identify the edges whose existence can maximally alleviate the node-level bias for the instance-level GNN prediction. Different GNN explanation models that are able to identify edge sets as the node-level explanations can be the backbone of the two explainers. Besides, to reflect the true reasoning result in the given GNN model, both identified explanations should be faithful to the given GNN prediction. This leads

[8]Without loss of generality, we focus on binary sensitive attribute here.

to the challenge of *Faithfulness Gap*: how to achieve the bias(fairness)-related goal of each explanation and maintain faithfulness to the given GNN prediction at the same time? To tackle this challenge, we design a constraint to enforce the faithfulness of the identified explanations, and incorporate such constraint into a unified objective function for the proposed framework. Optimizing such a unified objective function helps to achieve two goals: (1) the bias(fairness)-related explanation goals of both explainers; and (2) the goal of faithfulness through end-to-end training. The overall structure of REFEREE is presented in Fig. 4.7. Given a trained GNN $f_{\mathbf{\Theta}}$, a node v_i, and its computation graph $\mathcal{G}_i$, the goal of *Bias Explainer* is to identify an edge set $\tilde{\mathcal{E}}_i$ that maximally accounts for the exhibited node-level bias of v_i as the bias explanation, while the goal of *Fairness Explainer* is to identify an edge set $\tilde{\mathcal{E}}_i'$ as the fairness explanation, where the edges in $\tilde{\mathcal{E}}_i'$ maximally alleviate the node-level bias of v_i.

3.2.3.3 Objective Function

In this subsection, we introduce the unified objective function formulation of our proposed framework REFEREE. Generally, the unified objective function includes three components, namely explaining bias (fairness), enforcing fidelity, and refining explanation.

3.2.3.4 Explaining Bias (Fairness).

Here we first introduce the bias (fairness)-related constraints to enable the two explainers to identify the edges that maximally account for the node-level bias and the edges whose existence maximally alleviate the node-level bias for a given GNN prediction, respectively. We start from the constraint for the *Bias Explainer*. Given any computation graph, the basic goal of Bias Explainer is to identify the edges that maximally account for the node-level bias as an obtained edge set for the explanation. Intuitively, if only edges that maximally account for the exhibited node-level bias are identified and preserved in such edge set, the probabilistic outcome based on such edge set will exhibit more node-level bias. This is because some edges whose existence help to alleviate the node-level bias in the vanilla computation graph are not involved in the obtained edge set anymore. As such, we develop the first component of our unified objective function towards the goal of explaining bias (fairness) below.

We denote the identified edge set given by Bias Explainer as $\tilde{\mathcal{E}}_i$. Here $\tilde{\mathcal{E}}_i \in \tilde{\mathcal{G}}_i$, and $\tilde{\mathcal{G}}_i$ is the computation graph with the obtained edge set $\tilde{\mathcal{E}}_i$. We represent the probabilistic outcome of the GNN model based on the computation graph $\tilde{\mathcal{G}}_i$ for node v_i as $\tilde{\mathbf{y}}_i = f_{\mathbf{\Theta}}(\tilde{\mathcal{G}}_i)$, where $f_{\mathbf{\Theta}}$ is a given trained GNN model with fixed parameter $\mathbf{\Theta}$. We utilize $\tilde{\mathcal{Y}}$ to denote the GNN outcome set $\hat{\mathcal{Y}}$ with the original element $\hat{\mathbf{y}}_i$ being replaced by $\tilde{\mathbf{y}}_i$, i.e., $\tilde{\mathcal{Y}} = \hat{\mathcal{Y}} \backslash \{\hat{\mathbf{y}}_i\} \cup \{\tilde{\mathbf{y}}_i\}$. According to the sensitive feature, we split $\tilde{\mathcal{Y}}$ into two outcome sets as $\tilde{\mathcal{Y}}_0$ and $\tilde{\mathcal{Y}}_1$ ($\tilde{\mathcal{Y}}_0 \cup \tilde{\mathcal{Y}}_1 = \tilde{\mathcal{Y}}$). We denote the distribution of $\tilde{\mathcal{Y}}_0$ and $\tilde{\mathcal{Y}}_1$ as $P(\tilde{\mathcal{Y}}_0)$ and $P(\tilde{\mathcal{Y}}_1)$, respectively. Generally, if the vanilla probabilistic outcome $\hat{\mathbf{y}}_i$ is replaced with $\tilde{\mathbf{y}}_i$, the outcome distribution distance between the two sensitive subgroups will also be changed accordingly. Considering that $\tilde{\mathbf{y}}_i$ is derived based on the input computation graph $\tilde{\mathcal{G}}_i$, the identified edges in $\tilde{\mathcal{E}}_i \in \tilde{\mathcal{G}}_i$ then determine how the distribution distance changes. As discussed above, the probabilistic outcome based on $\tilde{\mathcal{E}}_i$ will exhibit more node-level bias. From Definition 3.2.1, we know that the identified edges in $\tilde{\mathcal{E}}_i$ are supposed to lead to a larger distribution distance between $P(\tilde{\mathcal{Y}}_0)$ and $P(\tilde{\mathcal{Y}}_1)$. Hence we

formulate the goal of bias explanation based on Wasserstein-1 distance as

$$\max_{\tilde{\mathcal{E}}_i} \mathrm{W}_1(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1)), \tag{3.9}$$

where $\mathrm{W}_1(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1))$ is formally presented as

$$W_1(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1)) = \inf \mathbb{E}_{(\tilde{\mathbf{y}}_{(0)}, \tilde{\mathbf{y}}_{(1)}) \sim \kappa}[\|\tilde{\mathbf{y}}_{(0)} - \tilde{\mathbf{y}}_{(1)}\|_1]. \tag{3.10}$$

Here $\kappa \in \Pi(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1))$; $\Pi(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1))$ is the set including all possible joint distributions of $\kappa(\tilde{\mathbf{y}}_{(0)}, \tilde{\mathbf{y}}_{(1)})$ whose marginals are $P(\tilde{\mathcal{Y}}_0)$ and $P(\tilde{\mathcal{Y}}_1)$, respectively. Generally, Eq. (3.9) encourages the Bias Explainer to identify edges that maximally account for the Wasserstein-1 distance between $P(\tilde{\mathcal{Y}}_0)$ and $P(\tilde{\mathcal{Y}}_1)$. Nevertheless, the infimum in Eq. (3.10) is intractable. To perform effective optimization with gradient-based optimizing techniques (e.g., stochastic gradient descent), we adopted a widely used approximation strategy [36] for the Wasserstein distance and the corresponding gradients during optimization, which has been empirically proved to be effective [71].

We follow a similar approach to set up the other goal for Fairness Explainer to encourage the identification of edges whose existence can maximally alleviate the node-level bias for the given GNN prediction. We assume the edge set given by Fairness Explainer as $\tilde{\mathcal{E}}_i'$, where $\tilde{\mathcal{E}}_i' \in \tilde{\mathcal{G}}_i'$. Here $\tilde{\mathcal{G}}_i'$ is the computation graph with the identified $\tilde{\mathcal{E}}_i'$. We denote the outcome of the GNN model based on $\tilde{\mathcal{G}}_i'$ for node v_i as $\tilde{\mathbf{y}}_i' = f_{\mathbf{\Theta}}(\tilde{\mathcal{G}}_i')$. We use $\tilde{\mathcal{Y}}_0'$ and $\tilde{\mathcal{Y}}_1'$ to denote the subsets of $\tilde{\mathcal{Y}}' = \hat{\mathcal{Y}} \backslash \{\hat{\mathbf{y}}_i\} \cup \{\tilde{\mathbf{y}}_i'\}$ according to the sensitive feature. Correspondingly, $P(\tilde{\mathcal{Y}}_0')$ and $P(\tilde{\mathcal{Y}}_1')$ are the distributions of $\tilde{\mathcal{Y}}_0'$ and $\tilde{\mathcal{Y}}_1'$, respectively. We formulate the goal of Fairness Explainer as

$$\min_{\tilde{\mathcal{E}}_i'} \mathrm{W}_1(P(\tilde{\mathcal{Y}}_0'), P(\tilde{\mathcal{Y}}_1')), \tag{3.11}$$

where $\tilde{\mathcal{E}}_i'$ is the edge set given by the Fairness Explainer for explanation. To summarize, we formulate the objective function term towards explaining bias (fairness) as

$$\mathscr{L}_1(\mathbf{\Phi}, \mathbf{\Phi}') = \mathrm{W}_1(P(\tilde{\mathcal{Y}}_0'), P(\tilde{\mathcal{Y}}_1')) - \mathrm{W}_1(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1)). \tag{3.12}$$

The basic rationale is that when $\mathscr{L}_1$ is minimized, $\mathrm{W}_1(P(\tilde{\mathcal{Y}}_0), P(\tilde{\mathcal{Y}}_1))$ is maximized to encourage Bias Explainer to identify edges that account for the exhibited node-level bias; $\mathrm{W}_1(P(\tilde{\mathcal{Y}}_0'), P(\tilde{\mathcal{Y}}_1'))$ is minimized to encourage Fairness Explainer to identify edges whose existence can maximally alleviate the node-level bias.

Nevertheless, considering that the probabilistic outcome corresponding to only one explained node is changed during the optimization of Eq. (3.9) (or Eq. (3.11)), the numerical change of Wasserstein-1 distance could be small. Correspondingly, when using gradient-based techniques to optimize the two explainers in REFEREE, the gradients of $\mathscr{L}_1$ w.r.t. the learnable parameters in the two explainers could be similar. This could lead to a phenomenon that the two explainers tend to converge at similar solutions, which means that $\tilde{\mathcal{E}}_i \in \tilde{\mathcal{G}}_i$ and $\tilde{\mathcal{E}}_i' \in \tilde{\mathcal{G}}_i'$ could be close. To better differentiate the edges that are supposed to be separated into two different explanations, here we propose to add a contrastive loss between the two explainers. The intuition here is to encourage the Bias Explainer and the Fairness Explainer to identify different edges from each other. Specifically, the distribution difference between the edges in $\tilde{\mathcal{E}}_i$ and $\tilde{\mathcal{E}}_i'$ are maximized as an encouragement for identifying different edges. It should be noted that the edge sets given by both the two explainers (i.e., $\tilde{\mathcal{E}}_i$ and $\tilde{\mathcal{E}}_i'$) are

based on the edge set $\mathcal{E}_i$ in the given computation graph. Correspondingly, we denote the distribution of $\tilde{\mathcal{E}}_i$ and $\tilde{\mathcal{E}}_i'$ conditional on the given $\mathcal{E}_i$ as $P_{\mathbf{\Phi}}(\tilde{\mathcal{E}}_i|\mathcal{E}_i)$ and $P_{\mathbf{\Phi}'}(\tilde{\mathcal{E}}_i'|\mathcal{E}_i)$, respectively. We give the optimization problem as

$$\max_{\mathbf{\Phi},\mathbf{\Phi}'} \text{Dist\_Diff}(P_{\mathbf{\Phi}'}(\tilde{\mathcal{E}}_i'|\mathcal{E}_i)\|P_{\mathbf{\Phi}}(\tilde{\mathcal{E}}_i|\mathcal{E}_i)), \tag{3.13}$$

Various metrics can be adopted as the distribution difference operator Dist_Diff(.), e.g., Jensen–Shannon divergence and Wasserstein distance, etc. We give the second objective function term as

$$\mathscr{L}_2(\mathbf{\Phi},\mathbf{\Phi}') = -\text{Dist\_Diff}(P_{\mathbf{\Phi}'}(\tilde{\mathcal{E}}_i'|\mathcal{E}_i)\|P_{\mathbf{\Phi}}(\tilde{\mathcal{E}}_i|\mathcal{E}_i)). \tag{3.14}$$

Minimizing $\mathscr{L}_2$ helps to encourage the two explainers to yield different edge sets from each other as the identified explanations.

3.2.3.5 Enforcing Fidelity.

The explanations given by the two explainers should be able to reflect the true reasoning result given the node-level GNN prediction. Hence, for both explainers (i.e., the Bias Explainer and Fairness Explainer), the output explanation should be faithful to the given GNN prediction. In other words, given a node v_i, the structural explanations given by both the two explainers should lead to the same predicted label based on the given GNN $f_{\mathbf{\Theta}}$. Based on such intuition, here we leverage the mutual information between the original predicted label and the subgraph with the identified edge set for explanation to formulate fidelity enforcement. This also aligns with some existing works on GNN explanation [228]. We first introduce the fidelity enforcement formulation for the Bias Explainer. Specifically, for node v_i, the mutual information between the original GNN prediction $\hat{Y}_i \in \{1,...,C\}$ and the underlying subgraph $\tilde{\mathcal{G}}_i$ is maximized to ensure that $\tilde{\mathcal{E}}_i \in \tilde{\mathcal{G}}_i$ encodes the critical information of the given GNN prediction, which is formulated as

$$\max_{\tilde{\mathcal{G}}_i} \text{MI}(\hat{Y}_i, \tilde{\mathcal{G}}_i) = \text{H}(\hat{Y}_i) - \text{H}(\hat{Y}_i|\tilde{\mathcal{G}}_i). \tag{3.15}$$

Here $\text{MI}(\cdot,\cdot)$ denotes the mutual information computation operator, and $\text{H}(\cdot)$ represents the entropy function. It is worth mentioning that in Eq. (3.15), the value of the entropy term $\text{H}(\hat{Y}_i) = \text{H}(f_{\mathbf{\Theta}}(\mathcal{G}_i))$ is fixed, as the explanation model is post-hoc (i.e., the parameters in the given GNN model are fixed). Therefore, the optimization problem in Eq. (3.15) can be reduced to only minimizing the second entropy term, where $\text{H}(\hat{Y}_i|\tilde{\mathcal{G}}_i)$ can be presented as

$$\text{H}(\hat{Y}_i|\tilde{\mathcal{G}}_i) = -\mathbb{E}_{\hat{Y}_i|\tilde{\mathcal{G}}_i}[\log P_{\mathbf{\Theta}}(\hat{Y}_i|\tilde{\mathcal{G}}_i)]. \tag{3.16}$$

Considering fidelity is necessary for both explainers, we give the fidelity constraint for Fairness Explainer similarly. The objective function term to enforce fidelity for both explainers is given as

$$\mathscr{L}_3(\mathbf{\Phi},\mathbf{\Phi}') = -\mathbb{E}_{\hat{Y}_i|\tilde{\mathcal{G}}_i}[\log P_{\mathbf{\Theta}}(\hat{Y}_i|\tilde{\mathcal{G}}_i)] - \mathbb{E}_{\hat{Y}_i|\tilde{\mathcal{G}}_i'}[\log P_{\mathbf{\Theta}}(\hat{Y}_i|\tilde{\mathcal{G}}_i')]. \tag{3.17}$$

Minimizing $\mathscr{L}_3$ enforces the identified edges in $\tilde{\mathcal{G}}_i$ and $\tilde{\mathcal{G}}_i'$ to encode as much critical information to $\hat{Y}_i$ as possible.

3.2.3.6 Refining Explanation.

As mentioned in Section 4.2.1, our proposed explanation framework should be able to identify two edge sets, where the edges in one set can maximally account for the exhibited node-level bias, and the existence of the edges in the other set can maximally alleviate the node-level bias in GNNs. Therefore, the identified explanations for both explainers should be maximally refined. Intuitively, to refine the learned explanations, those goal-irrelevant edges for the GNN outcome of node v_i should be maximally identified and removed from the structural explanations of both explainers, i.e., the learned edge sets from both explainers should be sparse. Here we propose to regularize the sparsity of the identified edge set to remove those goal-irrelevant edges maximally. We take the sparsity regularization of the Bias Explainer as an example. Note that the explanation of the identified edge set is identified via a weighted mask matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}_i|\times|\mathcal{V}_i|}$ which indicates the edge importance score with entry values. We propose to utilize the ℓ_1-norm of the mask matrix $\mathbf{M}$ for the Bias Explainer as the regularization, i.e., $\|\mathbf{M}\|_1$. Considering both explainers, the corresponding objective function term $\mathscr{L}_4$ is formulated as

$$\mathscr{L}_4(\mathbf{\Phi}, \mathbf{\Phi}') = \|\mathbf{M}\|_1 + \|\mathbf{M}'\|_1 \tag{3.18}$$

for the two explainers. Here $\mathbf{\Phi}$ is the parameter of Bias Explainer $h_{\mathbf{\Phi}}$, and $\mathbf{\Phi}'$ denotes the parameter of Fairness Explainer $h_{\mathbf{\Phi}'}$. $\mathbf{M}$ and $\mathbf{M}'$ are used to indicate the edge weights given by the explanations from the Bias Explainer and Fairness Explainer, respectively. Besides, there are also cases where people are only interested in a certain number of top-ranked critical edges. In other words, there could be a pre-assigned budget T for the explanation edge set $\tilde{\mathcal{E}}_i$, i.e., $|\tilde{\mathcal{E}}_i| \leq T$. In this case, we formulate the $\mathscr{L}_4$ as

$$\mathscr{L}_4(\mathbf{\Phi}, \mathbf{\Phi}', T, T') = \text{ReLU}(\|\mathbf{M}\|_1 - T) + \text{ReLU}(\|\mathbf{M}'\|_1 - T') \tag{3.19}$$

given pre-assigned budget T and T' for $\tilde{\mathcal{E}}_i$ and $\tilde{\mathcal{E}}_i'$, respectively. Intuitively, minimizing $\mathscr{L}_4$ helps to remove those goal-irrelevant edges maximally to refine the identified explanation.

3.2.3.7 Unified Objective Function Formulation.

Based on our discussions on enforcing fidelity, explaining bias (fairness), and refining explanation, we formally formulate the unified objective function for the proposed GNN explanation framework REFEREE as

$$\mathscr{L} = \mathscr{L}_1 + \alpha\mathscr{L}_2 + \beta\mathscr{L}_3 + \gamma\mathscr{L}_4. \tag{3.20}$$

Here α, β, and γ are hyper-parameters controlling the effect of the three constraining terms. For any specific node to be explained, minimizing the objective function in Eq. (3.20) aims to: (1) encourage the Bias Explainer to identify an edge set that maximally accounts for the node-level bias in the given GNN; and (2) encourage the Fairness Explainer to identify an edge set that maximally contributes to the fairness for the given GNN prediction.

3.2.4 Experimental Evaluations

In this section, we first introduce the downstream learning task and the real-world datasets adopted for evaluation. The experimental settings and the implementation details are then introduced. Next, we present the empirical evaluation results of our proposed framework from the perspective of *Effectiveness of Explaining Bias (Fairness)*, *Explanation Fidelity*, and

Debiasing GNN with Explanations. In particular, we aim to answer the following research questions: **RQ1:** How well can REFEREE identify edges to explain bias (fairness) in GNNs given the prediction of a specific node? **RQ2:** How well can the explanations given by the two explainers in REFEREE be faithful to the given GNN? **RQ3:** How will the obtained explanations from REFEREE help with GNN debiasing for the whole population?

3.2.4.1 Experimental Settings

3.2.4.2 Downstream Task & Real-world Dataset.

In this paper, we focus on the widely studied *node classification* as the downstream task. We adopt three real-world attributed networks for experiments – German Credit, Recidivism, and Credit Defaulter [3], where all node labels are binary. A detailed description, supplementary discussion, and experimental results are in the online version.

3.2.4.3 Explainer Backbones.

Different GNN explanation approaches that are able to identify edge sets as the node-level explanations can be adopted as the backbone of the two explainers in REFEREE. To evaluate how well the proposed framework can be generalized to different explanation backbones, we adopt GNN Explainer [228] and PGExplainer [132] as explainer backbones for evaluation.

3.2.4.4 Baselines.

To the best of our knowledge, no other work is able to give structural explanations for the exhibited node-level bias of GNNs. Therefore, we modify some existing GNN explanation approaches to adapt them to explain exhibited node-level bias in terms of the computation graph structure. The adopted existing GNN explanation approaches for adaptation include the attention-based GNN explanation [188], the gradient-based GNN explanation [188], and two state-of-the-art GNN explanation approaches (GNN Explainer [228] and PGExplainer [132]). We elaborate more details on how we achieve the adaptation for these approaches as follows.

First, we introduce how we adapt these approaches as the baselines to evaluate *Effectiveness of Explaining Bias (Fairness)*. For attention-based explanation, we directly add a bias(fairness)-related objective onto the vanilla loss function of a Graph Attention Network (GAT) model [188] to maximize (as Eq. (3.9)) or minimize (as Eq. (3.11)) the Wasserstein-1 distance between the outcome distributions of the two sensitive subgroups. This enables the GAT model to identify the two types of critical edges for bias and fairness explanation, i.e., edges that maximally account for the exhibited node-level bias and edges whose existence can maximally alleviate the node-level bias. The learned attention weights are regarded as the indicator of the final explanations. For gradient-based explanation, we utilize the same objective function as the objective function adopted by attention-based explanation. The two types of critical edges for bias and fairness explanation are identified through gradient ascend w.r.t. the adjacency matrix of the given computation graph. For GNN Explainer and PGExplainer, we modified their objective function in a similar way as the attention-based explanation. Specifically, a bias(fairness)-related objective is added onto the vanilla loss function for both explanation models. For any given computation graph, the two types of critical edges for bias and fairness explanation are identified through maximizing (as Eq. (3.9))

or minimizing (as Eq. (3.11)) the Wasserstein-1 distance between the outcome distributions of the two sensitive subgroups.

Second, for the evaluation of *Explanation Fidelity*, we aim to compare whether the GNN explanation backbones in REFEREE can still maintain their faithfulness to the given GNN prediction. Here the most widely-used GNN Explainer is adopted as the baseline model. Correspondingly, GNN Explainer is also adopted as the backbone of the two explainers in REFEREE for a fair comparison.

Third, for the evaluation of *Debiasing GNNs with Explanation*, we adopt the same baselines as those adopted in the evaluation of *Effectiveness of Explaining Bias (Fairness)*.

3.2.4.5 Evaluation Metrics.

We first introduce the metrics for the evaluation of *Effectiveness of Explaining Bias (Fairness)*. Specifically, we evaluate how much the node-level bias B_i is promoted or reduced between the two sensitive subgroups when only the identified edge set is utilized for the GNN prediction of the given node. Intuitively, this enables us to evaluate how well each explainer can identify those edges that maximally account for the exhibited bias and edges whose existence can maximally alleviate the node-level bias for the prediction, respectively. For the evaluation of *Explanation Fidelity*, a widely acknowledged metric is *Fidelity*$-$ score [229]. Traditionally, *Fidelity*$-$ score measures the ratio of the consistent pairs between the vanilla correct predictions and the correct predictions based on the identified edge set. Nevertheless, to reflect the true reasoning process in GNNs, we argue that the faithfulness of those incorrect predictions is also critical, as bias may also exhibit and need to be explained for those incorrect predictions from the perspective of the usability of the GNNs. As a consequence, we extend the *Fidelity*$-$ score to measure the ratio of the consistent pairs between all vanilla predictions and the predictions based on the identified edge set. Formally, the extended fidelity metric for M explained nodes can be measured with $\text{Fidelity} = \frac{1}{M}\sum_{i=1}^{M}\left(\mathbb{1}\left(\hat{Y}_i = \tilde{Y}_i\right)\right)$. Here $\hat{Y}_i$ represents the vanilla GNN prediction for node v_i. $\tilde{Y}_i$ denotes the prediction of the given GNN $f_{\mathbf{\Theta}}$ for node v_i, where only the identified edges for explanation are preserved in the corresponding computation graph. $\mathbb{1}(\cdot)$ is the indicator function, which returns 1 if $\hat{Y}_i = \tilde{Y}_i$ and 0 otherwise. Finally, for *Debiasing GNNs with Explanation*, we utilize two traditional fairness metrics Δ_{SP} and Δ_{EO} to quantitatively evaluate how much the predictions of a GNN are debiased in terms of the whole population. Here Δ_{SP} and Δ_{EO} measure the positive prediction rate difference between two sensitive subgroups over all nodes and nodes with only positive class labels, respectively. Additionally, we use node classification accuracy to evaluate the GNN utility.

3.2.4.6 Effectiveness of Explaining Bias (Fairness)

To answer **RQ1**, we compare our proposed framework REFEREE with other baselines to evaluate the effectiveness of explaining bias (fairness). Here we adopt the widely used model GAT as the trained GNN for experiments, and similar results can be observed based on other GNNs. Specifically, we first randomly sample 50 nodes to be explained (i.e., $M = 50$). Then for each node, we obtain the two predictions of the given GNN $f_{\mathbf{\Theta}}$ based on the computation graph corresponding to each of the two identified edge sets given by the two explainers. For obtained predictions, the normalized average value of how much the node-level bias B_i is

TABLE 3.2. Promoted ΔB_i, denoted as ΔB_i (P), and reduced ΔB_i, denoted as ΔB_i (R), present how much Wasserstein-1 distance between the outcome distribution of two sensitive subgroups improves and reduces on average, respectively. Absolute values of normalized promotion and reduction are given in $\times 10^{-4}$ scale. Larger values indicate better effectiveness in explaining bias (fairness). GE- and PGE- prefixes indicate the backbone of both explainers in REFEREE as GNN Explainer and PGExplainer, respectively. The best results are in Bold.

	German		**Recidivism**		**Credit**	
	ΔB_i **(P)**	ΔB_i **(R)**	ΔB_i **(P)**	ΔB_i **(R)**	ΔB_i **(P)**	ΔB_i **(R)**
Att.	6.11 ± 2.51	7.84 ± 3.48	4.58 ± 1.67	7.18 ± 2.24	6.72 ± 0.75	8.48 ± 3.29
Grad.	4.27 ± 0.98	5.60 ± 1.85	3.59 ± 2.02	4.42 ± 2.01	5.97 ± 1.07	9.79 ± 1.78
GNN Explainer	5.17 ± 1.20	3.37 ± 1.53	1.74 ± 0.72	3.55 ± 2.08	7.41 ± 1.75	9.24 ± 2.66
PGExplainer	8.73 ± 0.74	9.37 ± 1.87	6.36 ± 2.39	8.66 ± 1.82	7.48 ± 2.70	10.54 ± 3.22
GE-REFEREE	14.29 ± 2.73	$\mathbf{14.45 \pm 2.29}$	$\mathbf{13.94 \pm 3.74}$	12.05 ± 2.79	10.30 ± 2.64	$\mathbf{15.07 \pm 3.35}$
PGE-REFEREE	$\mathbf{15.72 \pm 2.31}$	11.97 ± 2.62	10.39 ± 4.08	$\mathbf{12.57 \pm 3.12}$	$\mathbf{11.57 \pm 2.91}$	14.67 ± 3.49

promoted (given by Bias Explainer) or reduced (given by Fairness Explainer) compared with the vanilla B_i based on the complete computation graph is presented in Table 3.2. For both promotion and reduction of B_i, a larger value indicates better results, as more biased or fairer node-level outcome can be obtained based on the identified structural explanation. We make the following observations from Table 3.2:

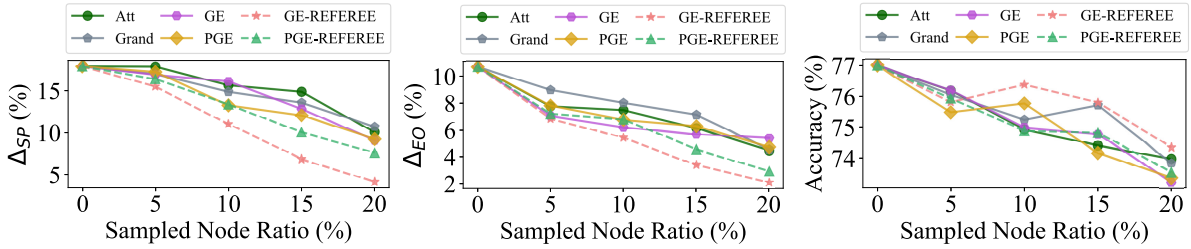
- Stable promotion and reduction of node-level bias is observed in all GNN explanation approaches. This indicates that the Wasserstein distance-based objective functions formulated in Eq. (3.9) and Eq. (3.11) effectively help to identify edges that account for the exhibited node-level bias and edges whose existence can alleviate the exhibited node-level bias.
- Existing GNN explanation models (e.g., GNN Explainer and PGExplainer) do not show any superior performance over other straightforward GNN explanation approaches such as Att and Grad. This observation implies that for these representative GNN explanation approaches, simply adding a constraint to explain bias (fairness) at the instance level only achieves limited effectiveness.
- Among all GNN explanation approaches, REFEREE yields the structural explanations that lead to the highest promotion and reduction of node-level bias in all datasets. Based on such observations, we argue that REFEREE achieves the best performance over other alternatives on identifying edges that account for the exhibited bias and whose existence can alleviate the exhibited node-level bias for the prediction.

3.2.4.7 Explanation Fidelity

We then answer **RQ2** in this subsection. Generally, it is necessary to ensure that the structural explanation results given by both explainers in REFEREE are faithful to the given trained GNN, i.e., the identified edge sets should encode critical information for the given GNN predictions. More specifically, in our experiments, the predicted labels given by the GNN model based on the computation graph with the identified edge sets should be the same as those based on the vanilla computation graph. To evaluate how well the proposed framework can maintain faithfulness when it is generalized to different GNNs, here we choose three widely used GNNs, namely GCN [111], GAT [188], and GIN [222] for explanation. Fidelity

TABLE 3.3. Explanation fidelity evaluation for different GNNs. Numerical results are in percentage. Vanilla denotes the explanation results given by the vanilla GNN Explainer. B. Explainer and F. Explainer represent the Bias Explainer and Fairness Explainer, respectively. The best results are in Bold.

		German	Recidivism	Credit
GCN	**Vanilla**	88.02 ± 1.48	90.04 ± 1.43	85.26 ± 1.67
	B. Explainer	**92.20 ± 1.39**	90.26 ± 3.24	87.60 ± 2.79
	F. Explainer	89.17 ± 0.85	**92.08** ± 2.44	**89.41** ± 4.08
GAT	**Vanilla**	83.65 ± 3.02	87.91 ± 2.04	**88.64 ± 3.41**
	B. Explainer	**85.71 ± 2.31**	90.51 ± 4.58	86.09 ± 2.07
	F. Explainer	84.40 ± 1.57	**91.98 ± 3.95**	87.04 ± 3.10
GIN	**Vanilla**	88.58 ± 2.50	**91.77 ± 1.42**	87.62 ± 2.60
	B. Explainer	88.11 ± 1.78	90.26 ± 4.13	86.47 ± 2.13
	F. Explainer	**89.67 ± 2.23**	91.45 ± 1.78	**88.17 ± 2.98**



(A) Δ_{SP} (B) Δ_{EO} (C) Accuracy

FIGURE 3.9. Debiasing GAT with explanations given by REFEREE with two different backbones and other baselines.

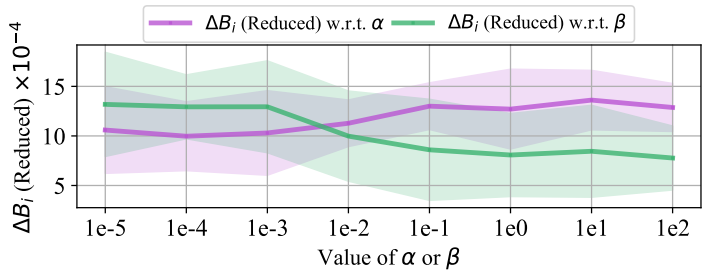
is adopted as the metric for evaluation. Intuitively, fidelity measures to what proportion the predicted labels based on the identified explanation are maintained to be the same as the vanilla ones. Here we adopt the GNN Explainer as the backbone of the two explainers in REFEREE. For both the baseline model and our proposed framework, we train and make predictions five times separately for 50 randomly selected nodes. We present the performance comparison between the two explainers in our framework and vanilla GNN Explainer on the average performance of fidelity in Table 3.3. We can make the observation that both Bias Explainer and Fairness Explainer achieve comparable performance on fidelity with the vanilla GNN Explainer across different datasets and GNNs. Consequently, we argue that the explanation given by REFEREE maintains faithfulness to the GNN predictions.

3.2.4.8 Debiasing GNNs with Explanations

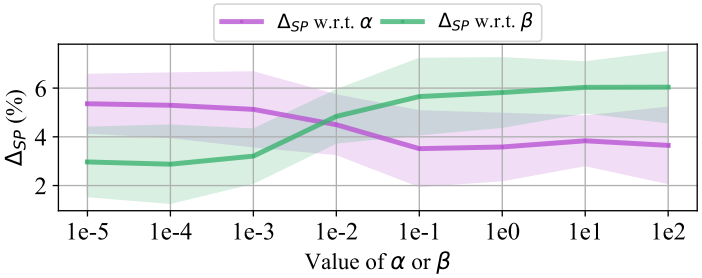
In this subsection, our goal is to answer **RQ3** to study how the instance-level explanations given by REFEREE help with GNN debiasing in terms of the whole population. A straightforward approach here is to first identify the edges that tend to introduce bias in the outcome of GNNs for some randomly sampled nodes, then remove such bias-introducing edges and input the network data into the GNN model to obtain less biased predictions. Nevertheless, the edges involved in the bias structural explanation (given by Bias Explainer) cannot be directly removed as a whole, as some edges could be critical for the GNN prediction of the explained

node. Besides, it is neither reasonable to only preserve the edges whose existence can maximally alleviate the node-level bias, as some removed non-critical edges for the explained node could be vital for the prediction of other nodes. Here we adopt an alternative strategy to study how the explanations help with GNN debiasing in terms of the whole population. Specifically, for those baseline explanation models, we randomly sample a subset of nodes for explanation. For each node, baselines are trained to learn structural explanations towards more biased and fairer predictions independently. Then edges that appear in the bias explanation but not in the fairness explanation are removed from the original input network. The intuition here is that if an edge only appears in the edge set that maximally accounts for the exhibited bias but not in the edge set whose existence can maximally alleviate the node-level bias of the prediction, such edge can be regarded as being more critical to the exhibited bias instead of being more critical to an accurate and fair prediction. Therefore, removing edges bearing such property has the potential to reduce the exhibited bias while maintaining the utility (i.e., yielding accurate and fair predictions) of the GNN. Correspondingly, for our proposed framework REFEREE, we also randomly sample nodes and remove edges that appear in the explanation given by Bias Explainer but not in the explanation from Fairness Explainer, i.e., removing edges in set $\tilde{\mathcal{E}}_i \backslash \tilde{\mathcal{E}}_i'$. In this way, edges are removed from the input network data towards the goal of debiasing the GNN and maintaining its usability at the same time. It is worth mentioning that such an edge removal strategy does not necessarily lead to graph structure modifications that are globally optimal for debiasing. However, if fairer GNNs can be achieved via removing edges that exhibit node-level bias defined in Definition 3.2.1, the consistency between Definition 3.2.1 and traditional fairness notions can be validated, i.e., reducing the node-level bias also helps to promote the overall fairness level of the GNN predictions in terms of traditional fairness metrics.

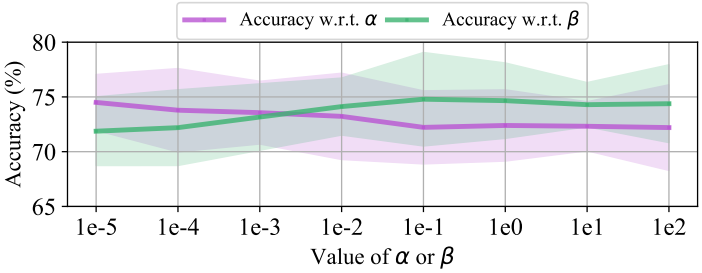
We adopt GAT as the explained GNN model here, and similar observations can also be found based on other GNNs. We vary the random sampling ratio of the number of explained nodes over the number of all nodes among {0%, 5%, 10%, 15%, 20%}. The changes of node classification accuracy, Δ_{SP}, and Δ_{EO} w.r.t. the sampled node ratio on German dataset is presented in Fig. 3.9. We make the following observations: (1) With more nodes being sampled and more edges that only appear in the bias explanations being removed, both Δ_{SP} and Δ_{EO} reduce significantly. This verifies that removing the edges that account for the node-level bias generally alleviates the exhibited bias in terms of the whole population. Besides, the reduction of both Δ_{SP} and Δ_{EO} also validates the consistency between traditional fairness notions and node-level bias given in Definition 3.2.1. (2) Removing the edges that only appear in the bias explanations generally reduces the GAT prediction accuracy. We argue that it is because some edges that lead to more biased results could also be critical for accurate predictions. However, the accuracy reduction is within an acceptable range. (3) Compared with other baseline approaches, REFEREE leads to limited accuracy reduction but achieves a more significant reduction on Δ_{SP} and Δ_{EO}. Such observation indicates that a fairer GNN is achieved (in terms of traditional fairness notions) based on the explanations identified by REFEREE compared with the explanations given by other alternatives. Considering our baselines also bear constraints for fidelity and explaining bias(fairness), it is safe to attribute such superiority to the designed contrastive mechanism of REFEREE. Consequently, we argue that REFEREE outperforms baselines in helping achieve fairer GNNs in terms of traditional fairness notions.



(A) The changes of relative ΔB_i (Reduced) w.r.t. α and β



(B) The changes of Δ_{SP} w.r.t. α and β



(C) Δ_{EO}

FIGURE 3.10. A parameter study of the proposed framework REFEREE based on hyper-parameter α and β. In (a), higher ΔB_i (Reduced) indicates better performance on explaining fairness for the Fairness Explainer in REFEREE. In (b), lower Δ_{SP} indicates a higher level of fairness is achieved based on the obtained explanations in terms of debiasing GNNs for the whole population. In (c), higher accuracy represents better GNN utility performance.

3.2.5 Parameter Sensitivity

We present the parameter sensitivity of our proposed framework REFEREE in this section. More specifically, we explore how the hyper-parameters α and β influence the performance of REFEREE on (1) explaining the bias (fairness) in GNNs and (2) debiasing the GNN across the whole population. Here α and β control the effect of the distribution difference constraint between the two explanations from the two explainers and the constraint to achieve better fidelity, respectively. In our experiments, we choose the widely used GAT model as the GNN to be explained, and we present the parameter study based on the performance of debiasing GNNs with explanations on German dataset. Similar observations can also be drawn on other GNN models and datasets.

Now we introduce the experimental settings for the parameter sensitivity study. Specifically, we fix the value for parameter γ as 1e-4 (the same as the setting in our implementation). First, for the parameter study of α, we set β =1e-4 (the same as the setting in our implementation), and we vary α from {1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2}. Second, for the parameter study of β, we set $\alpha = 1$ (the same as the setting in our implementation), and we also vary β from {1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2}. The performance changes of the proposed framework on explaining fairness (with ΔB_i (Reduced) being the node-level bias metric) and debiasing the GNN predictions over the whole population (with Δ_{SP} being the fairness metric and accuracy being the utility metric) are presented in Section 5.1.4.4. We can draw observations as below:

- From the perspective of explaining fairness (i.e., identifying the edges whose existence can maximally alleviate the exhibited node-level bias), we observe that a relatively larger α and a relatively smaller β help to achieve better performance, i.e., larger ΔB_i (Reduced), in Fig. 3.10a. This is because larger α and smaller β help the framework better differentiate the edges between the two types of explanations given by the two explainers. In this way, the fairness explainer is able to identify an edge set that leads to more significant node-level bias reduction, i.e., to give a fairness explanation that brings higher ΔB_i (Reduced) for any given node to be explained.
- From the perspective of debiasing the GNN predictions, we observe that a relatively larger α and a relatively smaller β help to achieve better debiasing performance in Fig. 3.10b. This is because: (1) Larger α helps to better differentiate the edges between bias explanation and the fair explanation. This makes it easier for the framework to distinguish the edges that account for the exhibited bias and edges whose existence can alleviate the node-level bias. (2) Smaller β means that the constraint strength on prediction fidelity is weak. This enables the framework to focus more on explaining bias (fairness) for edges in any given computation graph.
- From the perspective of maintaining GNN utility, we observe that a relatively smaller α and a relatively larger β help achieve higher prediction accuracy in Fig. 3.10c. This is because smaller α and larger β enforce the framework to focus more on the fidelity of the explanation. Therefore, more critical information could be encoded in the identified edges. Such an advantage leads to higher prediction accuracy based on the identified edges for any given node.
- Practically, it is necessary to balance the performance of bias reduction and model utility for any given GNN. In this regard, moderate values (e.g., values between 1e-4 and 1e0) for both α and β are recommended.

3.2.6 Related Work

Explanation of GNNs. Generally, existing GNN explanation approaches can be divided into data-level approaches and model-level ones [229]. For data-level approaches, the explanation models identify critical components in the input network data of GNNs, e.g., node features or edges. For example, squared gradient values are regarded as the importance scores of different input features in the node classification task [7]; interpretable surrogate models are leveraged to approximate the prediction of a certain GNN model, where the explanations from the surrogate model can be regarded as the explanation for the corresponding GNN

prediction [81, 191]. Another popular approach to identify important components of the input network data is to make perturbations on the input network, then observe the corresponding change in the output. The basic rationale is that if small perturbations lead to dramatic changes in the GNN prediction, then what has been perturbed is regarded as critical for the GNN prediction [228, 132, 231, 170, 197]. Despite its significance, this is a less studied topic. To provide model-level explanations for GNNs, graph generation can be leveraged to maximize the prediction of a GNN regarding a specific prediction (e.g., the probability of a class in graph classification) [230]. If the prediction probability of GNN regarding a specific prediction result can be maximized, then the generated input graph can be regarded as the explanation for this GNN that includes critical graph patterns. Different from the existing GNN explanation approaches, our proposed framework REFEREE not only explores critical edges for GNN predictions, but also identifies their contribution to the bias in GNNs. Hence, REFEREE is able to provide explanations for bias in GNNs, which helps understand how bias arises. This is with significance for GNN deployment in decision-critical scenarios and potentially facilitates the development of fairer GNNs.

Fairness of GNNs. With the increasing societal concerns on the fairness of GNNs [193], explorations have been made to alleviate the bias exhibited in GNNs. Generally, existing works focus either on *group fairness* [58] or *individual fairness* [233]. Group fairness requires that GNNs should not yield biased predictions against any specific demographic subgroups [134]. Among existing works, promoting group fairness through adversarial learning is one of the most popular GNN debiasing approaches [37]. Its goal is to train a discriminator to identify the sensitive information from the learned node embeddings. When the discriminator can barely distinguish the sensitive feature given any learned embedding, the sensitive feature can be regarded as being decoupled from the learned embeddings. Additionally, GNN debiasing can also be performed based on the input network data. For example, the network structure can be modified such that nodes in different demographic subgroups bear similar distributions on their neighbor node attribute values [47]. Moreover, edge dropout [179] is also proved to be effective in debiasing GNNs. On the other hand, individual fairness requires that similar individuals should be treated similarly [233, 104]. However, promoting individual fairness for GNNs remains under-explored. To the best of our knowledge, the only approach to fulfill such a goal is developed from a ranking perspective [46].

3.2.7 Conclusion

In this paper, we focus on a novel problem of structural explanation of node-level bias in GNNs. Specifically, we first propose to model node-level bias quantitatively, and then develop a principled post-hoc explanation framework named REFEREE with two different explainers: the bias explainer and the fairness explainer. Conditional on being faithful to the given GNN prediction, the two explainers aim to identify structural explanations that maximally account for the exhibited bias and that maximally contribute to the fairness level of the GNN prediction. Experiments on real-world network datasets demonstrate the effectiveness of REFEREE in identifying edges that maximally account for the exhibited node-level bias and edges whose existence can maximally alleviate the node-level bias for any given GNN prediction. Furthermore, REFEREE also shows superior performance over baselines on helping debias GNNs.

CHAPTER 4

Fairness Optimization for Graph Machine Learning

4.1 Individual Fairness for Graph Neural Networks: A Ranking-Based Approach

4.1.1 Introduction

Graph structured data is ubiquitous in today's increasingly connected world. Examples include social networks, biological networks, knowledge graphs, and critical infrastructure systems, to name a few. To gain deep insights from these graphs, a plethora of sophisticated graph mining algorithms have been developed in the past few decades [4, 68, 173, 199]. Among these efforts, Graph Neural Networks (GNNs) have emerged as a promising learning paradigm recently and demonstrated superior learning performance in diverse settings [111, 188, 74], which makes GNNs play an increasingly important role in various high-stake decision-making scenarios, e.g., credit scoring [176], recommendation [64], and medical diagnosis [64]. Nevertheless, close on the heels of the successful adoption of GNNs in various real-world scenarios has been the increasing societal concerns that these algorithms often do not have the fairness consideration, resulting in discriminatory actions toward specific groups or populations [133, 55, 77, 8]. For example, there is a growing practice of credit scoring using social network information [203], in which graph neural networks have become a de facto solution [193, 75]. Although these practices have shown to broaden opportunities for a larger portion of the population, they still yield unfair decisions for people in certain protected groups (e.g., low-income people) [133, 55].

To date, a wide spectrum of fairness measures has been developed to quantify and mitigate the bias of underlying learning algorithms [58, 233, 162, 118]. Existing fairness measures can be mainly divided into *group fairness measures* and *individual fairness measures* [133]. On the one hand, group fairness ensures that members of different protected groups (e.g., gender, race, and income) bear similar outcome statistics regarding model predictions [65, 77, 77, 232]. On the other hand, individual fairness scrutinizes potential bias and discrimination at a much finer granularity, and ensures similar individuals should yield similar prediction outcomes [58, 233]. Although much progress has been made in the field of algorithmic fairness, the studies of fairness issues in graph mining algorithms are fairly recent. Specifically, in the context of graph representation learning, a vast majority of existing works focus on the notion of group fairness, aiming to learn node embeddings that are independent of any protected attributes [162, 13, 18]. However, as graph data is naturally heterogeneous, different data modalities (i.e., graph structure and node attributes) are often coupled together. Thus bias and discrimination can exist in various shapes and formats. In this regard, beyond the notion

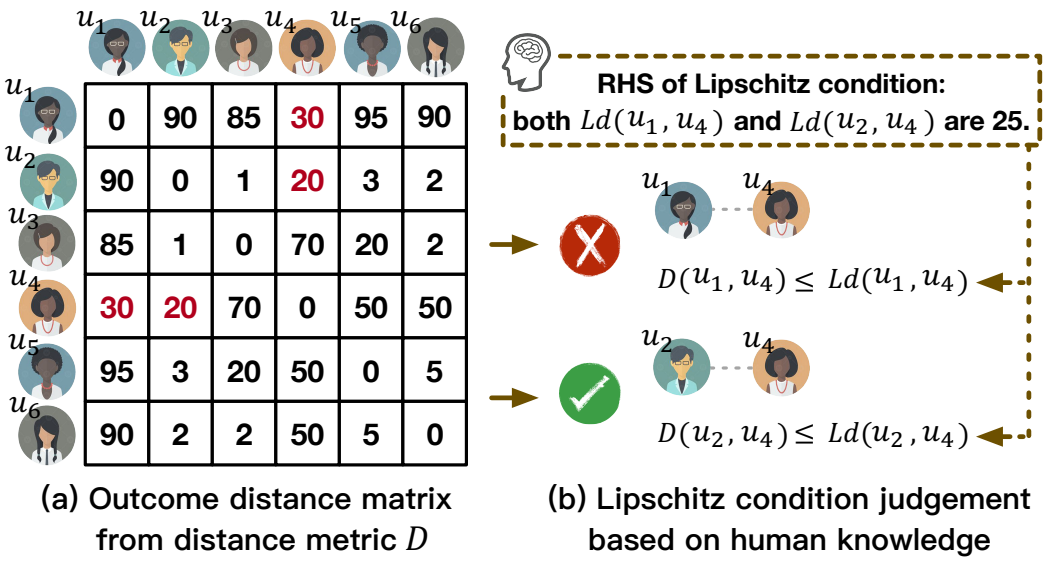


FIGURE 4.1. (a) Outcome distance for all instance pairs. (b) Lipschitz condition sanity check for pairs (u_1, u_4) and (u_2, u_4).

of group fairness regarding protected attributes, it is also desired to dig into the atomic components of graphs (i.e., a node) to ensure that graph representation learning renders similar results for similar individuals—in achieving *individual fairness*.

Despite the fundamental importance of achieving individual fairness for graph mining algorithms, the related studies are still in their infancy. In this work, we make an initial investigation to enhance the individual fairness of graph neural networks for decision-making. However, it remains a daunting problem mainly because of the following challenges: (1) ***Constraint Formulation.*** Formulating proper constraints to improve individual fairness is non-trivial. Traditionally, given a pair of instances, such constraint is achieved via the Lipschitz condition[1] [58, 104]. Nevertheless, the Lipschitz constant is often hard to be specified due to distance metric difference between the input and outcome space. (2) ***Distance Calibration.*** The absolute distance comparison in the Lipschitz condition fails to calibrate the differences between different instances. For example in Fig. 4.1, on the one hand for instance u_1, instance u_4 is the closest one to it in the outcome space. However, the distance between them violates the Lipschitz condition and thus we do not impose the individual fairness constraint between them (although we should since u_1 is much closer to u_4 than other instances). On the other hand for u_2, although u_4 is the second farthest one to it, Lipschitz condition is still satisfied and individual fairness constraint is imposed (in fact we may not need to do that since u_2 is much further to u_4 than u_2, u_5, and u_6). (3) ***End-to-End Learning Paradigm.*** A major advantage of GNNs over traditional unsupervised graph embedding algorithms is their end-to-end learning mechanism, i.e., the node embeddings are tailored for specific downstream learning tasks. In this regard, how to incorporate the individual fairness constraint seamlessly into the learning process without jeopardizing its end-to-end paradigm is another challenge to be tackled.

[1]Given a pair of instance x and y, their distance in the outcome space is upper bounded by their distance in the input space such that $D(f(x), f(y)) \leq Ld(x, y)$, where $f(.)$ maps instances to the output space, and L is the Lipschitz constant. $D(.,.)$ and $d(.,.)$ are two functions that measure the distance of instances in the output space and input space, respectively.

TABLE 4.1. Symbols and descriptions.

Symbols	Definitions or Descriptions
$\mathcal{M}$	backbone GNN model
$\mathcal{G}$	input graph
$\mathbf{A}$	adjacency matrix of graph $\mathcal{G}$
$\mathbf{X}$	node feature matrix of graph $\mathcal{G}$
$\mathbf{Y}$	ground truth of downstream learning task
$\hat{\mathbf{Y}}$	prediction of downstream learning task
$\mathbf{S}_{\mathcal{G}}$	oracle pairwise similarity matrix
$\mathbf{S}_{\hat{\mathbf{Y}}}$	pairwise similarity matrix from $\hat{\mathbf{Y}}$
n	number of nodes
d	number of node features
l	layer number in the backbone GNN model

In this paper, to tackle these challenges, we propose a principled framework REDRESS (short for Ranking basEd inDividual faiRnESS) to promote the individual fairness of graph neural networks. Specifically, to tackle the first two challenges, we refine the definition of individual fairness from a ranking perspective, and formulate the individual fairness constraint as "*for each instance u_i, the two ranking lists of other instances (based on their distances to u_i) in the input space and outcome space should be as similar as possible*". As such, we can avoid the delicate distance comparison between two different distance metrics in the Lipschitz condition, and the relative ranking comparison can also naturally alleviate the issue of uncalibrated distance. To tackle the third challenge, two optimization modules are encapsulated in REDRESS to improve model utility and individual fairness, respectively. To fit into the end-to-end training process, the two optimization modules are designed to adapt to the gradient-based optimization techniques. The main contributions of this paper can be summarized as follows.

- ***Problem Formulation.*** We study a novel problem of promoting individual fairness for graph neural networks from a ranking perspective.
- ***Algorithmic Design.*** We address the limitations of existing individual fairness constraints and propose a novel plug-and-play framework to mitigate the individual biases without jeopardizing the utility of underlying graph neural networks.
- ***Experimental Evaluations.*** We perform comprehensive experimental evaluations on real-world datasets to demonstrate the superiority of our proposed framework in terms of both bias mitigation and model utility maximization.

4.1.2 Problem Statement

In this section, we firstly present the notations used throughout this paper. Then we introduce the definition of *individual fairness from a ranking perspective*, followed by the problem formulation of *promoting ranking based individual fairness of GNNs*.

Notations. We use bold uppercase letters (e.g., $\mathbf{S}$), bold lowercase letters (e.g., $\mathbf{s}$), and normal lowercase letters (e.g., s) to denote matrices, vectors and scalars, respectively. Also, for any

matrix, e.g., $\mathbf{S}$, we represent its i-th row as $\mathbf{s}_i$, its (i,j)-th entry as $\mathbf{S}_{ij}$ or s_{ij}, and its transpose as $\mathbf{S}^\top$. For any scalar, $|\cdot|$ is the absolute value operator.

Let $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ be an input graph, where $\mathbf{A} \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix of the graph and $\mathbf{X} \in \mathbb{R}^{n \times d}$ denotes the matrix for node features (n nodes and d node features). $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ represent the ground truth and predictions for a specific downstream task, respectively. For example, if the downstream task is *node classification*, $\mathbf{Y} \in \{0,1\}^{n \times c}$ and $\hat{\mathbf{Y}} \in \mathbb{R}^{n \times c}$ are the ground truth and predicted class membership matrix (c classes), respectively.

To tackle the aforementioned challenges of *Constraint Formulation* and *Distance Calibration*, we refine the definition of individual fairness from a ranking perspective (Definition 4.1.1)[2]. We follow similar settings in [104, 120, 118], where the oracle pairwise similarity matrix $\mathbf{S}_\mathcal{G}$ is given apriori (e.g., assigned by specialists in [118]).

DEFINITION 4.1.1. ***Individual fairness from a ranking perspective.*** *Given the oracle pairwise similarity matrix $\mathbf{S}_\mathcal{G}$ of the input graph $\mathcal{G}$, and the similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ among instances in the outcome space (defined upon a similarity metric), we say the predictions are individually fair if for each instance i, the two ranking lists that encode the relative order of other instances (ranked based on the similarity between instance i and other instances in descending order) from $\mathbf{S}_\mathcal{G}$ and $\mathbf{S}_{\hat{\mathbf{Y}}}$ are consistent with each other.*

Example: Given a graph $\mathcal{G}$ with five nodes, suppose the ranking list that encodes the similarity between node u_1 and other nodes from $\mathbf{S}_\mathcal{G}$ is $\{u_4, u_3, u_2, u_5\}$, we say the predictions are are individually fair for node u_1 if the ranking list that encodes the similarity between u_1 and other nodes from $\mathbf{S}_{\hat{\mathbf{Y}}}$ is also $\{u_4, u_3, u_2, u_5\}$.

Based on Definition 4.1.1, we formulate the problem of *promoting ranking based individual fairness of GNNs* as follows.

PROBLEM 4.1.1. ***Promoting ranking based individual fairness of GNNs.*** *Given an input graph $\mathcal{G}$, a backbone GNN model $\mathcal{M}$ (e.g., GCN [111]), the ground truth $\mathbf{Y}$ and the predictions $\hat{\mathbf{Y}}$ corresponding to a specific downstream task (e.g., node classification), oracle pairwise similarity matrix $\mathbf{S}_\mathcal{G}$ from $\mathcal{G}$, and pairwise similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ obtained from $\hat{\mathbf{Y}}$, our goal is to promote the individual fairness of each node in the graph $\mathcal{G}$ according to Definition 4.1.1 without jeopardizing the utility of the model predictions (i.e., making $\hat{\mathbf{Y}}$ close to $\mathbf{Y}$).*

4.1.3 The Proposed Framework—REDRESS

In this section, we firstly introduce the overall structure of the proposed framework REDRESS. Then details of utility maximization and individual fairness promotion are presented, followed by the overall objective function formulation for training.

4.1.3.1 Overall Framework Structure

As the main focus of this work is to promote the individual fairness of GNNs during the decision-making process while maximally preserve the utility of the underlying learning models, we formulate these two desiderata as two separate modules and encapsulate them

[2]For the ease of presentation convenience, we use similarity measures instead of distance metrics. Similarity measure can be considered as an inverse distance metric.

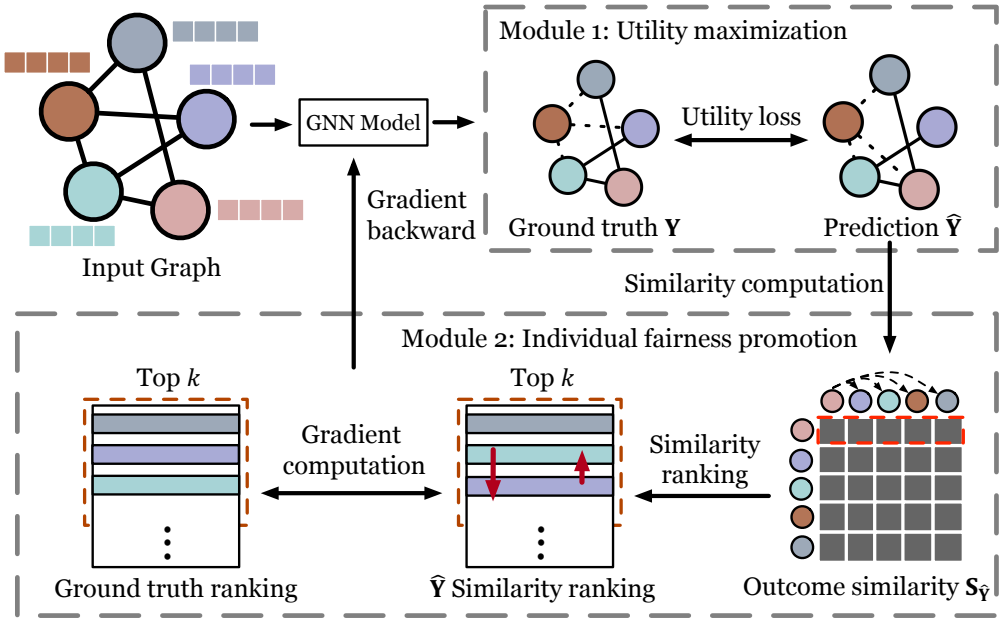


FIGURE 4.2. An illustration of the proposed REDRESS structure. Module 1 and 2 is utilized for model utility maximization and individual fairness promotion, respectively.

together with the GNN backbone into an end-to-end learning framework—REDRESS. The overall architecture of REDRESS is shown in Fig. 4.2. Essentially, it consists of three key parts: GNN backbone model, utility maximization (Module 1), and individual fairness promotion (Module 2).

- **GNN backbone model.** It provides a basic GNN architecture for downstream learning tasks. Some prevalent choices include GCN [111], GAE [110], and SGC [205].
- **Utility maximization.** To maximize the utility of the backbone model for a specific learning task, this module aims to minimize the prediction loss of the corresponding task.
- **Individual fairness optimization.** To relieve the individual bias toward fair decision-making, this module enforces the similarity ranking lists from $\mathbf{S}_{\hat{\mathbf{Y}}}$ and $\mathbf{S}_{\mathcal{G}}$ of each instance to be consistent according to Definition 1.

4.1.3.2 GNN Backbone Model

Acting as the backbone of the proposed framework, the GNN model takes the input $\mathcal{G}$ and outputs $\hat{\mathbf{Y}}$ as the predictions for a specific downstream learning task. The basic operation of GNN between l-th and $(l+1)$-th layer can be summarized as

$$\mathbf{h}_v^{(l+1)} = \sigma(\text{COMBINE}(\mathbf{h}_v^{(l)}, f(\{\mathbf{h}_u^{(l)} : u \in \mathcal{N}(v)\}))), \tag{4.1}$$

where $\mathbf{h}_v^{(l)}$ and $\mathbf{h}_v^{(l+1)}$ represent the embedding of node v at l-th and $(l+1)$-th layer, respectively. For a graph with node feature matrix $\mathbf{X}$, $\mathbf{h}_v^{(0)}$ can be initialized as the input node feature $\mathbf{x}_v$. $\mathcal{N}(v)$ indicates the neighbor set of node v according to the adjacency matrix $\mathbf{A}$. $f(.)$ denotes the aggregating function, e.g., weighted sum. COMBINE(.) indicates the combining function for output of $f(.)$ and $\mathbf{h}_v^{(l)}$, which combines the representation from the centering node and the

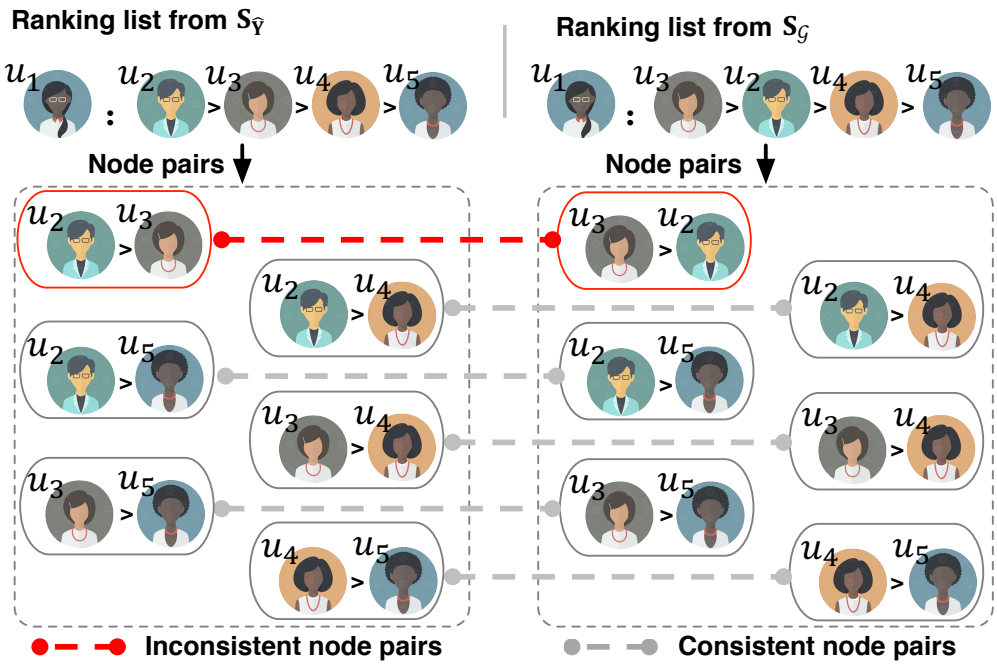


FIGURE 4.3. An illustrative example of the relative ranking order comparison for our proposed framework.

representations of its neighbors. $\sigma(.)$ represents the activation function (e.g., ReLU). Denote the output of the last GNN layer as matrix $\mathbf{Z} \in \mathbb{R}^{n \times c}$, then the predictions $\hat{\mathbf{Y}}$ of GNN can then be obtained as softmax($\mathbf{Z}$) $\in \mathbb{R}^{n \times c}$ for *node classification* [111] and sigmoid($\mathbf{Z}^\top \mathbf{Z}$) $\in \mathbb{R}^{n \times n}$ for *link prediction* [110].

4.1.3.3 Model Utility Maximization

To maximize the utility of the backbone GNN model in advancing downstream learning tasks, we need to enforce the predictions $\hat{\mathbf{Y}}$ to be closer to the ground truth $\mathbf{Y}$. To this end, a loss function corresponding to the specific learning task should be defined in Module 1 between $\mathbf{Y}$ and $\hat{\mathbf{Y}}$. For example, for the *node classification* and the *link prediction* tasks, the corresponding cross-entropy loss can be used to quantify the utility of the GNN model

$$\mathcal{L}_{\text{utility}} = -\sum_{(i,j) \in \mathcal{T}} \mathbf{Y}_{ij} \ln \hat{\mathbf{Y}}_{ij}. \tag{4.2}$$

Here $\mathcal{T}$ represents the (node, class) tuple set for training nodes in the *node classification* task and (node, node) tuple set for the vertices of training edges in the *link prediction* task. The utility maximization can be achieved by minimizing the cross-entropy loss in Eq. (4.2). Note that for this module, gradient-based optimization techniques can be directly applied for end-to-end training as the loss function is differentiable w.r.t. the model parameters.

4.1.3.4 Individual Fairness Promotion

As mentioned above, this module aims to promote the ranking based individual fairness for GNN such that for each node, the two ranking lists obtained from the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$ and the outcome similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ are consistent with each other. Since the ranking list

from the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$ is fixed, and $\mathbf{S}_{\hat{\mathbf{Y}}}$ is derived from the prediction outcome $\hat{\mathbf{Y}}$ via certain similarity metric, the ranking list from $\mathbf{S}_{\hat{\mathbf{Y}}}$ should be optimized via learning more appropriate $\hat{\mathbf{Y}}$—we refer it as *ranking optimization*. One straightforward solution to achieve this goal is to derive two ranking lists from $\mathbf{S}_{\mathcal{G}}$ and $\mathbf{S}_{\hat{\mathbf{Y}}}$ for each node, then define a loss function to quantify the difference between these two ranking lists. After that, we can combine the loss function over all nodes together and minimize the overall loss for a better $\hat{\mathbf{Y}}$ that can promote individual fairness. However, such a straightforward solution is often impractical as the ranking operations for the ranking lists will make the overall loss function not differentiable (w.r.t. the GNN model parameters) anymore, in a way the prevalent gradient-based optimization techniques cannot be directly applied. In other words, there is a gap between the involvement of ranking operations pertaining to Definition 1 and the need for gradient-based optimization techniques. To bridge the gap, we propose a novel ranking optimization strategy and we will elaborate more details in the following part.

4.1.3.5 Ranking Optimization

As mentioned above, minimizing the difference between two ranking lists (for each node) from $\mathbf{S}_{\hat{\mathbf{Y}}}$ and $\mathbf{S}_{\mathcal{G}}$ with gradient-based optimization techniques is difficult because of the non-differentiable ranking operations. Instead of formulating the loss based on ranking lists, here we propose a new loss formulation directly upon the outcome similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ and oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$. Since the new loss formulation does not rely on the ranking lists, gradient-based optimization techniques can then be applied. The new loss formulation is based on a probabilistic approach inspired by [161]. For each node, the new loss formulation will enforce the relative order of each node pair decided by $\mathbf{S}_{\hat{\mathbf{Y}}}$ and that decided by $\mathbf{S}_{\mathbf{G}}$ to be consistent. More specifically, for each node u_i, if it is with higher similarity value to u_j than u_m in $\mathbf{S}_{\hat{\mathbf{Y}}}$ (i.e., $\hat{s}_{i,j} > \hat{s}_{i,m}$, $i \neq j \neq m$, where $\hat{s}_{i,j}$ denotes the (i,j)-th entry of $\mathbf{S}_{\hat{\mathbf{Y}}}$), then it should also be with higher similarity value to u_j than u_m in the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$ (i.e., $s_{i,j} > s_{i,m}$, $i \neq j \neq m$, where $s_{i,j}$ denotes the (i,j)-th entry of $\mathbf{S}_{\mathcal{G}}$). In other words, the loss aims to penalize the node pairs whose relative similarity order are not consistent across the predicted similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$ and the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$ (as shown in Fig. 4.3). We then introduce the details of the loss formulation below.

To illustrate the formulation of the loss function, we take the loss computation of the node pair (u_j, u_m) centered on node u_i as an example. For node u_i, we define $\hat{P}_{j,m}(\hat{s}_{i,j}, \hat{s}_{i,m})$ as the predicted probability that the node u_i is more similar to node u_j than to node u_m. Here $\hat{s}_{i,j}$ and $\hat{s}_{i,m}$ represent the similarity score between node pairs (u_i, u_j) and (u_i, u_m) from the outcome similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$, respectively. To formulate it as a probability score between 0 and 1, we make use of the sigmoid function:

$$\hat{P}_{j,m}(\hat{s}_{i,j}, \hat{s}_{i,m}) = \frac{1}{1 + e^{-\alpha(\hat{s}_{i,j} - \hat{s}_{i,m})}}, \tag{4.3}$$

where α here is a scalar. Accordingly, define the known probability that the node u_i is more similar to node u_j than to node u_m as $P_{j,m}(s_{i,j}, s_{i,m})$, which can be formulated as follows:

$$P_{j,m}(s_{i,j}, s_{i,m}) = \begin{cases} 1, & s_{i,j} > s_{i,m}, \\ 0.5, & s_{i,j} = s_{i,m}, \\ 0, & s_{i,j} < s_{i,m}. \end{cases} \tag{4.4}$$

Here $s_{i,j}$ and $s_{i,m}$ denote the similarity between node pairs (u_i, u_j) and (u_i, u_m) from the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$, respectively. To promote individual fairness via ranking optimization, it is necessary to quantify and minimize the difference between the predicted probability distribution and the known one. Here, we make use of cross-entropy loss for the difference quantification between the two distributions. For example, the cross-entropy loss of node pair (u_j, u_m) centered on u_i can be expressed as

$$\mathcal{L}_{j,m}(i) = -P_{j,m}\log\hat{P}_{j,m} - (1 - P_{j,m})\log(1 - \hat{P}_{j,m}). \tag{4.5}$$

Then, the total loss function over all node pairs centered on node u_i can be formulated as

$$\mathcal{L}_{\text{fairness}}(i) = \sum_{j,m} \mathcal{L}_{j,m}(i)\,, \tag{4.6}$$

where $i \neq j \neq m$. By minimizing the above loss function, the relative order of all node pairs (centered on node u_i) decided by $\mathbf{S}_{\hat{\mathbf{Y}}}$ will be enforced to be consistent with the corresponding order decided by $\mathbf{S}_{\mathcal{G}}$. When the loss of all nodes is aggregated and minimized, the ranking based individual fairness can be achieved.

4.1.3.6 Training Facilitation.

Minimizing Eq. (4.6) for node u_i requires the ranking of all other nodes centered on node u_i being optimal (i.e., the ranking follows a descending order according to the similarity score from $\mathbf{S}_{\mathcal{G}}$). This is usually hard to achieve, especially for graphs with a large number of nodes. Consequently, for each node u_i, we propose to focus on the ranking optimization of the top-k nodes given by the outcome similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$. This strategy is motivated by the basic principle of individual fairness [58], which is only emphasizing the outcome of "similar people" to be similar. Motivated by existing research on learning to rank [161], we achieve this goal by developing a simple but effective approach. Specifically, we define $z_{@k}(\cdot,\cdot)$ as the similarity metric (e.g., $NDCG@k$ [88] or $ERR@k$ [23]) between the top-k ranking list derived from $\mathbf{S}_{\mathcal{G}}$ and the predicted top-k ranking list derived from $\mathbf{S}_{\hat{\mathbf{Y}}}$ for each node. For the loss of each node pair given by Eq. (4.5), we scale the loss term by the absolute value change of $z_{@k}$ if the ranking positions of the corresponding node pair u_j and u_m in the predicted ranking list are swapped. Then the loss for each node u_i can be presented as

$$\mathcal{L}_{\text{fairness}}(i) = \sum_{j,m} \mathcal{L}_{j,m}(i)|\Delta z_{@k}|_{j,m}\,, \tag{4.7}$$

where $i \neq j \neq m$. To illustrate the computation of $|\Delta z_{@k}|_{j,m}$, we take $k = 4, i = 1, j = 4$ and $m = 2$ as an example. Assume the top-4 ranking of node u_1 with other nodes in $\mathbf{S}_{\hat{\mathbf{Y}}}$ is $\hat{\textit{List}}_{u_i} = \{u_4, u_3, u_2, u_5\}$, and the corresponding ranking in $\mathbf{S}_{\mathcal{G}}$ is $\textit{List}_{u_i} = \{u_4, u_3, u_5, u_2\}$. Then the $|\Delta z_{@k}|_{4,2}$ corresponding to node pair (u_4, u_2) centered on node u_1 is $|z_{@k}(\textit{List}_{u_1}, \hat{\textit{List}}_{u_1}) - z_{@k}(\textit{List}_{u_1}, \hat{\textit{List}}'_{u_1})|$. Here $\hat{\textit{List}}'_{u_1} = \{u_2, u_3, u_4, u_5\}$, i.e., $\hat{\textit{List}}_{u_1}$ with the positions of node u_2 and u_4 swapped. With such method, the ranking optimization will be enforced to focus more on the top-k nodes for each node u_i. Here, k is often specified as a very number ($k \ll n$).

Besides, $|\Delta z_{@k}|_{j,m}$ is always zero if neither node u_j nor u_m is from the top-k nodes of u_i according to $\mathbf{S}_{\hat{\mathbf{Y}}}$. Consequently, this strategy also reduces the time complexity from $\mathcal{O}(n\cdot\binom{n-1}{2}) = \mathcal{O}(n^3)$ to $\mathcal{O}(n\cdot\binom{n-1}{1}\binom{k}{1}) = \mathcal{O}(n^2k)$ for the total loss computation. Nevertheless, the time complexity of $\mathcal{O}(n^2k)$ is still expensive for the training on large graphs. To

further reduce the time complexity, here we constrain that both nodes in the pair (u_j, u_m) are from the top-k ranked nodes of u_i according to $\mathbf{S}_{\hat{\mathbf{Y}}}$. Then the total fairness loss can be formulated as

$$\mathcal{L}_{\text{fairness}} = \sum_{i} \sum_{j,m:j,m\in\mathcal{K}(i)} \mathcal{L}_{j,m}(i)|\Delta z_{@k}|_{j,m}\,, \tag{4.8}$$

where $i \neq j \neq m$, and $\mathcal{K}(i)$ represents the top-k ranked node set for node u_i. In this way, the computational complexity can be further reduced from $\mathcal{O}(n^2k)$ to $O(n\cdot\binom{k}{2}) = O(nk^2)$, which facilitates the training process of REDRESS.

4.1.3.7 Overall Objective Function

Now we have $\mathcal{L}_{\text{utility}}$ for the model utility maximization (formulated in Section 4.1.3.3), and $\mathcal{L}_{\text{fairness}}$ for the model individual fairness promotion (formulated in Section 4.1.3.4). Then the overall objective function of the proposed framework REDRESS can be attained by combining the two formulations together:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{utility}} + \gamma\mathcal{L}_{\text{fairness}}. \tag{4.9}$$

Here γ is a tunable hyperparameter controlling the strength of individual fairness constraint. For training of the proposed framework, the gradient-based techniques can be directly adopted to minimize the total objective function $\mathcal{L}_{\text{total}}$.

4.1.4 Experimental Evaluations

In this section, we first introduce the adopted downstream learning tasks and the used datasets. Then we present the experimental settings and the implementation details. At last, we show the empirical evaluation results of REDRESS.

4.1.4.1 Downstream Tasks and Datasets

Downstream Tasks. To assess the performance of our proposed framework REDRESS, we choose the widely adopted *semi-supervised node classification* task [111, 193, 205] and *link prediction* task [110, 150] as the downstream learning tasks. Both of these two tasks are of high practical significance in many areas.

Datasets. To comprehensively explore how REDRESS promotes the individual fairness of GNNs from a ranking perspective, we adopt three different real-world datasets for each of the chosen downstream task. Specifically, for the *semi-supervised node classification* task, we adopt one citation network (ACM [183]) and two co-authorship networks (Co-author-CS and Co-author-Phy [174] from the KDD Cup 2016 challenge). For the *link prediction* task, three social networks (BlogCatalog [184], Flickr [82], and Facebook [121]) are used for evaluation. All of these datasets are publicly available. For citation networks, each node represents a paper, and an edge between two nodes denotes the citation relationship between two papers. For co-author networks, nodes represent authors, and an edge between two nodes indicates that the linked two authors have co-authored a paper together. Node attributes of both citation and co-author networks are generated by the bag-of-words model based on the abstract of the published paper. For social networks, each node represents a user, and links represent the corresponding interactions between users. The attributes here are constructed by the

profiles or descriptions of users. Here, CS and Phy are short for the datasets Co-author-CS and Co-author-Phy, respectively. The detailed statistics of these datasets are shown in Table 4.2.

4.1.4.2 Experiment Settings

GNN Backbone Models. As mentioned previously, our proposed REDRESS is a plug-and-play framework which can be easily generalized to any prevalent GNN architectures. Hence, we choose two different backbone GNN architectures for each downstream learning task in our experiments. For the *semi-supervised node classification* task, Graph Convolutional Network (GCN) [111] and Simplifying Graph Convolutional Network (SGC) [205] are adopted as our backbones. For the *link prediction* task, GCN and Variational Graph Auto-Encoders (GAE) [110] are employed.

Oracle Similarity Matrix. Following the settings of [104, 120, 118], the oracle similarity matrix $\mathbf{S}_{\mathcal{G}}$ of the input graph $\mathcal{G}$ is a given apriori. In practice, the oracle similarity matrix is often problem-specific and is determined by humans. To show the generalization capability of REDRESS in handling different types of oracle similarity matrix, we construct two different types of oracle similarity matrix from the feature perspective and the structural perspective. From the feature perspective, we compute the cosine similarity between input node features as the $\mathbf{S}_{\mathcal{G}}$; while from the structural perspective, we compute the Jaccard similarity between node pairs as the $\mathbf{S}_{\mathcal{G}}$. For the outcome similarity matrix $\mathbf{S}_{\hat{\mathbf{Y}}}$, we utilize the cosine distance, which is a widely adopted distance metric to measure similarity in the embedding space.

Baselines. To demonstrate the superiority of our proposed ranking-based individual fairness framework, we compare REDRESS with the following individual fairness promotion approaches on top of the backbone GNN models. It should be noted that the existing group fairness graph embedding methods (such as [162, 13]) cannot be used for comparison as they achieve fairness for subgroups determined by specific protected attributes while we focus on the notion of individual fairness without such protected attributes.

- **PFR** [118]: PFR aims to learn fair representations to achieve the notion of individual fairness. It has demonstrated to outperform traditional approaches such as [120, 233, 77] on individual fairness promotion. Since PFR can be considered as a pre-processing strategy and is not tailored for graph data, we employ it on the input node features to generate a new fair node feature representation and feed it into the backbone GNN models for learning.
- **InFoRM** [104]: InFoRM is a recently proposed individual fairness framework for conventional graph mining tasks (e.g., PageRank, Spectral Clustering) based on the Lipschitz condition. Here, we adapt InFoRM to different backbone GNN models by combing its individual fairness promotion loss and the unity loss of the backbone GNN model together, and then optimize the final loss in an end-to-end manner.

Evaluation Metrics. For model utility evaluation, we adopt classification accuracy (ACC) and area under receiver operating characteristic curve (AUC) for the node classification task and the link prediction task, respectively. For individual fairness evaluation, we adopt two widely used ranking metrics $NDCG@k$ [88] and $ERR@k$ [23] to measure the similarity between the ranking list from $\mathbf{S}_{\hat{\mathbf{Y}}}$ (outcome similarity matrix) and $\mathbf{S}_{\mathcal{G}}$ (oracle similarity matrix)

TABLE 4.2. Detailed statistics of the used datasets for node classification (short as NC) and link prediction (short as LP).

	Dataset	# Nodes	# Edges	# Features	# Classes
NC	**ACM**	16,484	71,980	8,337	9
	CS	18,333	81,894	6,805	15
	Phy	34,493	247,962	8,415	5
LP	**BlogCatalog**	5,196	171,743	8,189	N/A
	Flickr	7,575	239,738	12,047	N/A
	Facebook	4,039	88,234	1,406	N/A

for each node. The average value of $NDCG@k$ and $ERR@k$ across all nodes[3] are reported. $k = 10$ is adopted for quantitative performance comparison, but different choices of k are also studied. The quantitative performance and corresponding discussion based on $ERR@k$ is provided in the online version[4].

4.1.4.3 Implementation Details

REDRESS is implemented in Pytorch [154]. For all GNN backbones adopted in our experiments (i.e., GCN[5], SGC[6] and GAE[7]), we utilize their released implementations. We set the learning rate of all experiments as 0.01 for both the node classification task and the link prediction task. For GCN and SGC based models, the layer and hidden unit number is set as 2 and 16, respectively. For GAE based models, we set the graph convolutional layer number as 3, with the two hidden unit number being 32 and 16. For the training of REDRESS in all experiments, we set γ and k in the loss function as 1 and 10, respectively. All models are optimized with Adam optimizer [109]. For both of the two downstream tasks, datasets are randomly shuffled, and only training data is visible for all models. More details, including dataset split and hyper-parameter settings, are provided in the online version.

4.1.4.4 Effectiveness of REDRESS

Now we perform experiments on real-world networks to validate the effectiveness of the proposed REDRESS framework. We aim to answer the following research questions:

- **RQ1:** How well can REDRESS balance the GNN model utility and individual fairness compared with other baselines?
- **RQ2:** How will the individual fairness promotion hyperparameter γ affect the performance of REDRESS?
- **RQ3:** How will the choice of parameter k affect the performance of REDRESS?

RQ1: Performance on Balancing Utility and Fairness. First, we investigate the effectiveness of the proposed REDRESS framework by comparing its performance on balancing the

[3]all nodes in the test set for node classification and all nodes for link prediction

[4]See online version here: https://dl.acm.org/doi/abs/10.1145/3447548.3467266 for supplementary discussion and experimental results.

[5]https://github.com/tkipf/pygcn

[6]https://github.com/Tiiiger/SGC

[7]https://github.com/tkipf/gae

TABLE 4.3. Node classification results on ACM, Co-author-CS (CS) and Co-author-Phy (Phy) datasets. BB represents the backbone GNN model. Vanilla denotes the vanilla GNN. All values are reported in percentage. Best performance is marked in bold.

	BB	Model	Feature Similarity		Structural Similarity	
			ACC	NDCG@10	ACC	NDCG@10
ACM	GCN	**Vanilla**	**72.49 ± 0.6**	47.33 ± 1.0	**72.49 ± 0.6**	25.42 ± 0.6
		InFoRM	68.03 ± 0.3	39.79 ± 0.3	69.13 ± 0.5	12.02 ± 0.4
		PFR	67.88 ± 1.1	31.20 ± 0.2	69.00 ± 0.7	23.85 ± 1.3
		REDRESS (Ours)	71.75 ± 0.4	**49.13 ± 0.4**	72.03 ± 0.9	**29.09 ± 0.4**
	SGC	**Vanilla**	68.40 ± 1.0	55.75 ± 1.1	**68.40 ± 1.0**	37.18 ± 0.6
		InFoRM	**68.81 ± 0.5**	48.25 ± 0.5	66.71 ± 0.6	28.33 ± 0.6
		PFR	67.97 ± 0.7	34.71 ± 0.1	67.78 ± 0.1	37.15 ± 0.6
		REDRESS (Ours)	67.16 ± 0.2	**58.64 ± 0.4**	67.77 ± 0.4	**38.95 ± 0.1**
CS	GCN	**Vanilla**	90.59 ± 0.3	50.84 ± 1.2	**90.59 ± 0.3**	18.29 ± 0.8
		InFoRM	88.66 ± 1.1	53.38 ± 1.6	87.55 ± 0.9	19.18 ± 0.9
		PFR	87.51 ± 0.7	37.12 ± 0.9	86.16 ± 0.2	11.98 ± 1.3
		REDRESS (Ours)	**90.70 ± 0.2**	**55.01 ± 1.9**	89.16 ± 0.3	**21.28 ± 0.3**
	SGC	**Vanilla**	87.48 ± 0.8	74.00 ± 0.1	87.48 ± 0.8	32.36 ± 0.3
		InFoRM	88.07 ± 0.1	74.29 ± 0.1	88.65 ± 0.4	32.37 ± 0.4
		PFR	88.31 ± 0.1	48.40 ± 0.1	84.34 ± 0.3	28.87 ± 0.9
		REDRESS (Ours)	**90.01 ± 0.2**	**76.60 ± 0.1**	**89.35 ± 0.1**	**34.24 ± 0.2**
Phy	GCN	**Vanilla**	**94.81 ± 0.2**	34.83 ± 1.1	**94.81 ± 0.2**	1.57 ± 0.1
		InFoRM	89.33 ± 0.8	31.25 ± 0.0	94.46 ± 0.2	1.77 ± 0.0
		PFR	89.74 ± 0.5	24.16 ± 0.4	87.26 ± 0.2	1.20 ± 0.1
		REDRESS (Ours)	94.63 ± 0.7	**43.64 ± 0.5**	93.94 ± 0.3	**1.93 ± 0.1**
	SGC	**Vanilla**	**94.45 ± 0.2**	49.63 ± 0.1	**94.45 ± 0.2**	3.61 ± 0.1
		InFoRM	92.01 ± 0.1	43.87 ± 0.2	94.27 ± 0.3	3.64 ± 0.0
		PFR	89.74 ± 0.3	28.54 ± 0.1	89.73 ± 0.3	2.62 ± 0.1
		REDRESS (Ours)	94.30 ± 0.1	**53.40 ± 0.1**	93.94 ± 0.2	**4.03 ± 0.0**

model utility and individual fairness against state-of-the-art alternatives. For generalization purpose, the performance of REDRESS is compared with other baselines under different settings of oracle similarity matrices (i.e., feature similarity and structure similarity) and different GNN backbones (i.e., GCN, SGC, and GAE). Quantitative results for the *node classification* task and the *link prediction* task are shown in Table 4.3 and Table 4.4, respectively. In these two tables, higher ACC and AUC represents better performance on model utility, and higher $NDCG@10$ indicates better performance on individual fairness. We can make the following observations from these two tables:

- From the perspective of model utility (i.e., ACC in node classification and AUC in link prediction), our proposed framework REDRESS provides competitive performance compared with other state-of-the-art baselines. Besides, our proposed framework REDRESS achieves better utility performance compared with the vanilla

GNN backbones in some cases. We conjecture that this is partly because the individual fairness promotion term plays the role of regularization to prevent over-fitting of the backbone GNN models.

- From the perspective of ranking based individual fairness, our framework outperforms all baseline methods in all cases with different levels of improvement w.r.t. the fairness evaluation metric $NDCG@10$. This verifies the effectiveness of the individual fairness promotion of our proposed framework REDRESS. PFR and InFoRM do not improve $NDCG@10$ in some cases due to the fact that their algorithms are not designed for ranking based individual fairness optimization.
- From the perspective of balancing the model utility and individual fairness, our framework achieves both competitive model utility performance and superior individual fairness promotion in all cases compared with other baselines. Based on such observations, we argue that our framework achieves better performance on balancing the model utility and individual fairness compared with other alternatives.

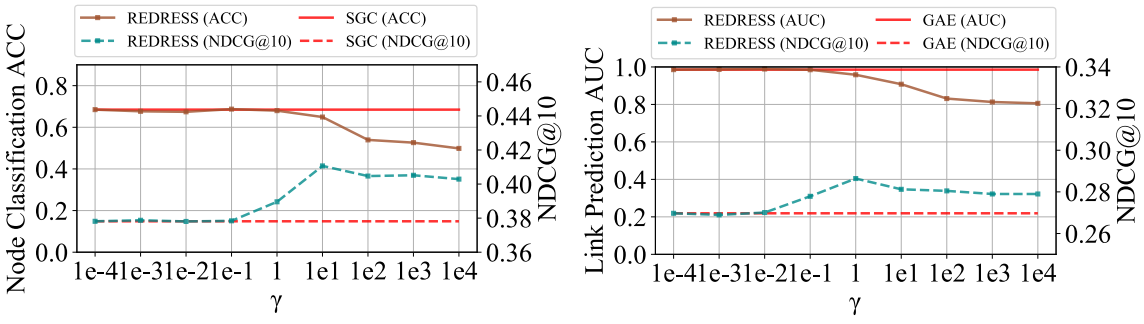
RQ2: Influence of Individual Fairness Promotion Hyperparameter γ. In our framework, the strength of individual fairness promotion is controlled by hyperparameter γ as defined in Eq. (4.9). To explore how γ affects the performance of REDRESS, we vary it among {1e-4, 1e-3, 1e-2, 1e-1, 1, 1e1, 1e2, 1e3, 1e4} and report the performance on utility and individual fairness within fixed training epochs. Due to space limit, here we only present the results from (a) ACM with SGC backbone based on Jaccard similarity, and (b) Facebook with GAE backbone based on cosine similarity as Fig. 4.4. We can make the following observations (we also have similar observations in other datasets):

- When γ is relatively small (e.g., smaller than 1e-1 for ACM and 1e-2 for Facebook), the individual fairness constraint makes little difference to the performance of REDRESS on the model utility and $NDCG@10$ for both tasks.
- When γ is a modest value (e.g., between 1e-1 and 1e1 for ACM or between 1e-2 and 1 for Facebook), $NDCG@10$ can be improved with little sacrifice on ACC or AUC. In other words, an appropriate γ helps to achieve better individual fairness performance without jeopardizing the model utility compared with the vanilla SGC and GAE. This shows that REDRESS achieves a proper balance between promoting individual fairness and maintaining the model utility.
- When γ is relatively large (e.g., larger than 1e1 for ACM and 1 for Facebook), ACC and AUC will be affected by the strength of individual fairness promotion. At the same time, $NDCG@10$ also drops as γ gets larger. This is because when γ falls in this area, the top-10 ranked nodes are far from optimal, and individual fairness promotion module can hardly achieve better performance within fixed epochs.

RQ3: Influence of the Choice of k. At last, we investigate what the performance of REDRESS will be like under different choices of k. We also present the model utility and individual fairness performance of REDRESS on: (a) ACM with SGC backbone based on Jaccard similarity, and (b) Facebook with GAE backbone based on cosine similarity in Fig. 4.5. Here we vary k among {2, 5, 10, 20, 50, 100}. Based on the tendencies presented, we can make the following observations (similar observations in other datasets).

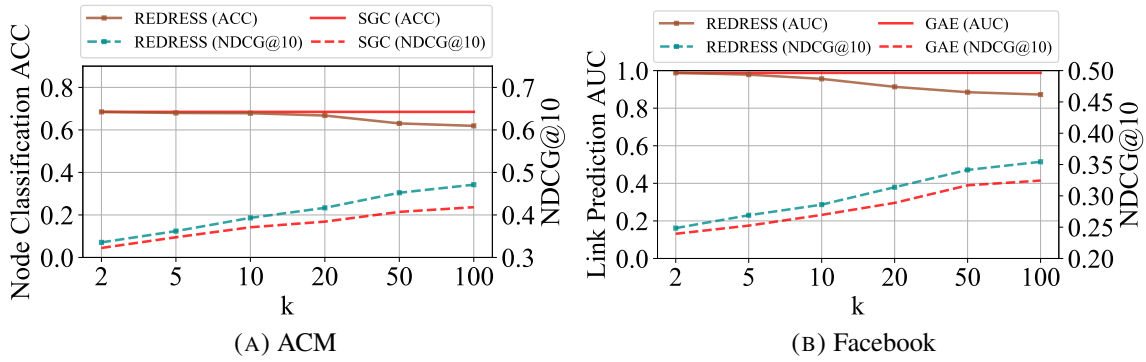
TABLE 4.4. Link prediction results on BlogCatalog (Blog), Flickr and Facebook (FB) datasets. BB represents the backbone GNN model. Vanilla denotes the vanilla GNN. All values are reported in percentage. Best performance is marked in bold.

	BB	Model	Feature Similarity		Structural Similarity	
			AUC	NDCG@10	AUC	NDCG@10
Blog	GCN	Vanilla	85.87 ± 0.1	16.73 ± 0.1	85.87 ± 0.1	32.47 ± 0.5
		InFoRM	79.85 ± 0.6	15.57 ± 0.2	84.00 ± 0.1	26.18 ± 0.3
		PFR	84.25 ± 0.2	16.37 ± 0.0	83.88 ± 0.0	29.60 ± 0.4
		REDRESS (Ours)	**86.49 ± 0.8**	**17.66 ± 0.2**	**86.25 ± 0.3**	**34.62 ± 0.7**
	GAE	Vanilla	**85.72 ± 0.1**	17.13 ± 0.1	85.72 ± 0.1	41.99 ± 0.4
		InFoRM	80.01 ± 0.2	16.12 ± 0.2	82.86 ± 0.0	27.29 ± 0.3
		PFR	83.83 ± 0.1	16.64 ± 0.0	83.87 ± 0.1	35.91 ± 0.4
		REDRESS (Ours)	84.67 ± 0.9	**18.19 ± 0.1**	**86.36 ± 1.5**	**43.51 ± 0.7**
Flickr	GCN	Vanilla	**92.20 ± 0.3**	13.10 ± 0.2	92.20 ± 0.3	22.35 ± 0.3
		InFoRM	91.39 ± 0.0	11.95 ± 0.1	91.73 ± 0.1	23.28 ± 0.6
		PFR	91.91 ± 0.1	12.94 ± 0.0	91.86 ± 0.2	19.80 ± 0.4
		REDRESS (Ours)	91.38 ± 0.1	**13.58 ± 0.3**	**92.67 ± 0.2**	**28.45 ± 0.5**
	GAE	Vanilla	89.98 ± 0.1	12.77 ± 0.0	89.98 ± 0.1	23.58 ± 0.2
		InFoRM	88.76 ± 0.7	12.07 ± 0.1	**91.51 ± 0.2**	15.78 ± 0.3
		PFR	**90.30 ± 0.1**	12.12 ± 0.1	90.10 ± 0.1	20.46 ± 0.3
		REDRESS (Ours)	89.45 ± 0.5	**14.24 ± 0.1**	89.52 ± 0.3	**29.83 ± 0.2**
FB	GCN	Vanilla	95.60 ± 1.7	23.07 ± 0.2	95.60 ± 1.7	16.55 ± 1.1
		InFoRM	90.26 ± 0.1	23.23 ± 0.3	**96.66 ± 0.6**	15.18 ± 0.7
		PFR	87.11 ± 1.2	21.83 ± 0.2	94.87 ± 1.9	19.53 ± 0.5
		REDRESS (Ours)	**96.49 ± 1.6**	**29.60 ± 0.1**	92.66 ± 0.4	**27.73 ± 1.1**
	GAE	Vanilla	**98.54 ± 0.0**	26.75 ± 0.1	**98.54 ± 0.0**	27.03 ± 0.1
		InFoRM	90.50 ± 0.4	22.77 ± 0.2	95.03 ± 0.1	15.38 ± 0.2
		PFR	96.91 ± 0.1	23.52 ± 0.1	98.28 ± 0.0	22.89 ± 0.3
		REDRESS (Ours)	95.98 ± 1.5	**28.43 ± 0.3**	94.07 ± 1.7	**33.53 ± 0.2**



(A) ACM (B) Facebook

FIGURE 4.4. Study on individual fairness constraint strength: (a) REDRESS with SGC backbone and Jaccard similarity on ACM; (b) REDRESS with GAE backbone and cosine similarity on Facebook.



(A) ACM (B) Facebook

FIGURE 4.5. Study on k choices: (a) REDRESS with SGC backbone and Jaccard similarity on ACM; (b) REDRESS with GAE backbone and cosine similarity on Facebook.

- As k goes larger, REDRESS achieves larger improvement on $NDCG@10$. This proves that larger k brings better optimization effectiveness on individual fairness promotion.
- Model utility performance is barely influenced when k gets larger. This implies REDRESS achieves a proper balance between maintaining the model utility and promoting individual fairness under different choices of k in the optimization. In practice, a modest k (e.g., 20 for ACM and 10 for Facebook) achieves best balancing performance.

4.1.5 Supplementary Discussion

Supplementary quantitative performance of REDRESS and other alternatives based on ERR (Expected Reciprocal Rank) are also provided in Table 4.5 and Table 4.6 for node classification and link prediction task, respectively. Similar to section 4.1.4.4, the performance of REDRESS is also compared with other baselines under different settings of oracle similarity matrix (i.e., feature similarity and structure similarity) and different GNN backbones (i.e., GCN, SGC, and GAE). It should be noted that higher ACC and AUC represents better performance on model utility, and higher $ERR@10$ indicates better performance on individual fairness. We can make the following observations from the two tables:

- From the perspective of model utility (i.e., ACC in node classification and AUC in link prediction), REDRESS provides competitive performance compared with other state-of-the-art baselines. Besides, similar to the $NDCG@10$ based experiments (Table 4.3 and Table 4.4), better utility performance from REDRESS can also be found compared with the vanilla GNN backbones. This further verifies that the individual fairness promotion term plays the role of regularization to prevent over-fitting of the backbone GNN models.
- From the perspective of ranking based individual fairness, our framework outperforms all baseline methods in all cases with different levels of improvement w.r.t. the fairness evaluation metric $ERR@10$. Considering that similar observation can also be found in $NDCG@10$ based experiments as in Table 4.3 and Table 4.4, the

TABLE 4.5. Node classification results on ACM, Co-author-CS (CS) and Co-author-Phy (Phy) datasets. BB represents the backbone GNN model. Vanilla denotes the vanilla GNN. All values are reported in percentage. Best performance is marked in bold.

BB	Model	Feature Similarity		Structural Similarity	
		ACC	ERR@10	ACC	ERR@10
ACM	GCN				
	Vanilla	72.49 ± 0.6	75.70 ± 0.6	**72.49** ± 0.6	37.55 ± 0.4
	InFoRM	67.65 ± 1.0	73.49 ± 0.5	65.91 ± 0.2	19.96 ± 0.6
	PFR	68.48 ± 0.6	76.28 ± 0.1	70.22 ± 0.7	36.54 ± 0.4
	REDRESS (Ours)	**73.46 ± 0.2**	**82.27 ± 0.1**	71.87 ± 0.4	**43.74 ± 0.0**
	SGC				
	Vanilla	**68.40 ± 1.0**	80.06 ± 0.1	**68.40 ± 1.0**	45.95 ± 0.3
	InFoRM	67.96 ± 0.5	75.63 ± 0.5	66.16 ± 0.6	39.79 ± 0.1
	PFR	67.69 ± 0.4	76.80 ± 0.1	66.69 ± 0.3	46.99 ± 0.5
	REDRESS (Ours)	66.51 ± 0.3	**82.32 ± 0.3**	67.10 ± 0.7	**49.02 ± 0.2**
CS	GCN				
	Vanilla	**90.59 ± 0.3**	80.41 ± 0.1	**90.59** ± 0.3	26.69 ± 1.3
	InFoRM	88.37 ± 0.9	80.63 ± 0.6	87.10 ± 0.9	29.68 ± 0.6
	PFR	87.62 ± 0.2	76.26 ± 0.1	85.66 ± 0.7	19.80 ± 1.4
	REDRESS (Ours)	90.06 ± 0.5	**83.24 ± 0.2**	89.81 ± 0.2	**32.42 ± 1.6**
	SGC				
	Vanilla	87.48 ± 0.8	90.58 ± 0.1	87.48 ± 0.8	43.28 ± 0.2
	InFoRM	87.31 ± 0.5	90.64 ± 0.1	88.21 ± 0.9	43.37 ± 0.1
	PFR	87.95 ± 0.2	79.85 ± 0.2	86.93 ± 0.1	38.83 ± 0.8
	REDRESS (Ours)	**90.48 ± 0.2**	**92.03 ± 0.1**	**90.39 ± 0.1**	**45.81 ± 0.0**
Phy	GCN				
	Vanilla	**94.81 ± 0.2**	73.25 ± 0.3	**94.81 ± 0.2**	2.58 ± 0.1
	InFoRM	88.67 ± 0.7	73.80 ± 0.6	94.68 ± 0.2	2.45 ± 0.1
	PFR	88.79 ± 0.2	73.32 ± 0.4	89.69 ± 1.0	1.67 ± 0.1
	REDRESS (Ours)	93.71 ± 0.1	**80.23 ± 0.1**	93.91 ± 0.4	**3.22 ± 0.3**
	SGC				
	Vanilla	94.45 ± 0.2	77.48 ± 0.2	94.45 ± 0.2	4.50 ± 0.1
	InFoRM	92.06 ± 0.2	75.13 ± 0.4	94.27 ± 0.1	4.44 ± 0.0
	PFR	87.39 ± 1.2	73.42 ± 0.2	89.16 ± 0.3	3.41 ± 0.2
	REDRESS (Ours)	**94.81 ± 0.2**	**79.57 ± 0.2**	**94.54 ± 0.1**	**4.98 ± 0.1**

generalization ability of REDRESS on individual fairness promotion can be further verified.

- From the perspective of balancing the model utility and individual fairness, our framework achieves both competitive model utility performance and superior individual fairness promotion in all $ERR@10$ based cases compared with other baselines. Similar observation can also be found in $NDCG@10$ based experiments as in Table 4.3 and Table 4.4. Based on these observations, we argue that our framework generally achieves better performance on balancing the model utility and individual fairness compared with other alternatives under different ranking similarity metrics.

4.1.6 Related Work

Individual Fairness. Dwork et al. [58] first argue that only emphasizing group fairness regarding protected attributes can barely treat each individual user in a fair manner, and

TABLE 4.6. Link prediction results on BlogCatalog (Blog), Flickr and Facebook (FB) datasets. BB represents the backbone GNN model. Vanilla denotes the vanilla GNN. All values are reported in percentage. Best performance is marked in bold.

	BB	Model	Feature Similarity		Structural Similarity	
			AUC	ERR@10	AUC	ERR@10
Blog	GCN	**Vanilla**	**85.87 ± 0.1**	67.95 ± 0.1	85.87 ± 0.1	38.63 ± 0.2
		InFoRM	80.14 ± 0.1	68.55 ± 0.1	83.68 ± 0.0	34.26 ± 0.9
		PFR	83.65 ± 0.0	68.04 ± 0.3	84.72 ± 0.1	37.28 ± 0.4
		REDRESS (Ours)	83.90 ± 0.2	**72.83 ± 0.2**	**86.44 ± 0.0**	**42.16 ± 0.1**
	GAE	**Vanilla**	**85.72 ± 0.1**	67.92 ± 0.1	85.72 ± 0.1	44.23 ± 0.2
		InFoRM	81.87 ± 0.1	68.36 ± 0.4	82.50 ± 0.1	33.98 ± 0.5
		PFR	83.49 ± 0.1	67.89 ± 0.0	84.31 ± 0.1	39.89 ± 0.2
		REDRESS (Ours)	85.30 ± 1.5	**69.62 ± 0.4**	**85.77 ± 2.0**	**47.44 ± 0.3**
Flickr	GCN	**Vanilla**	92.20 ± 0.3	70.39 ± 0.1	92.20 ± 0.3	38.44 ± 0.5
		InFoRM	91.28 ± 0.0	72.17 ± 0.0	**92.24 ± 0.0**	39.03 ± 0.4
		PFR	**92.43 ± 0.2**	71.36 ± 0.2	92.06 ± 0.2	37.29 ± 0.7
		REDRESS (Ours)	87.89 ± 0.4	**73.90 ± 0.3**	91.39 ± 0.0	**44.82 ± 0.5**
	GAE	**Vanilla**	89.98 ± 0.1	70.34 ± 0.2	89.98 ± 0.1	36.98 ± 0.3
		InFoRM	90.56 ± 1.4	71.54 ± 0.1	**91.55 ± 0.2**	35.58 ± 0.4
		PFR	90.44 ± 0.2	71.65 ± 0.2	90.09 ± 0.2	33.89 ± 0.3
		REDRESS (Ours)	**93.06 ± 0.3**	**72.41 ± 0.2**	87.96 ± 0.4	**44.00 ± 0.1**
FB	GCN	**Vanilla**	95.60 ± 1.7	61.52 ± 0.5	**95.60 ± 1.7**	32.18 ± 1.7
		InFoRM	90.66 ± 0.0	61.49 ± 0.2	94.65 ± 1.3	30.03 ± 1.7
		PFR	89.85 ± 2.0	62.02 ± 0.3	92.30 ± 0.5	30.62 ± 1.8
		REDRESS (Ours)	**95.99 ± 1.9**	**64.08 ± 0.1**	92.93 ± 0.8	**43.74 ± 1.5**
	GAE	**Vanilla**	**98.54 ± 0.0**	63.19 ± 0.1	**98.54 ± 0.0**	42.17 ± 0.4
		InFoRM	92.80 ± 0.1	62.29 ± 0.0	94.75 ± 0.2	31.93 ± 0.6
		PFR	96.85 ± 0.1	61.71 ± 0.1	98.18 ± 0.1	39.04 ± 0.3
		REDRESS (Ours)	95.10 ± 0.7	**64.40 ± 0.7**	92.35 ± 0.3	**44.54 ± 0.3**

propose the definition of individual fairness: *similar individuals should be treated similarly*. In their work, Lipschitz condition is utilized as the distance constrain for instance pairs between the input and outcome of the decision-making model. Zemel et al. [233] propose to emphasize the balance between the decision-making model utility and individual fairness. Individual fairness is promoted in their work via sharing the mapping function from the model input to corresponding outcome over all individuals. Lahoti et al. [120] point out that most individual fairness works are limited within binary classification problems. They firstly extend the problem setting to multi-class, and improve the model performance on individual fairness via learning low-rank representations for individuals in a model-agnostic way. Another work from Lahoti et al. [118] specifies similar individual pairs by human experts before training, and only emphasize the individual fairness optimization over these pre-assigned pairs. Different from these mentioned works, we define individual fairness from a ranking perspective, and promote individual fairness via ranking-based optimization. To

the best of our knowledge, we are the first to define and promote individual fairness from the ranking perspective.

Fairness in Graphs. Graph structured data has become ubiquitous in various high-impact areas. Nevertheless, most previous efforts achieving fairness in graphs focus on group fairness. Basically, group fairness emphasizes that all demographic groups (defined by sensitive features) should receive their fair share of interest. Among previous works, Rahman et al. [162] achieve the first-of-its-kind study to realize graph embedding learning with group fairness considerations. A modified random walk algorithm is proposed to ensure that the minority (according to sensitive features) bears the same appearing probability in the walk compared with other demographic groups. Bose et al. [13] propose to disentangle the learned embeddings from the sensitive features with an adversarial learning framework. A similar adversarial approach is also adopted by Dai et al. [37] for debiasing graph mining results. Palowitch et al. [147] promote group fairness via ensuring that the node embeddings are trained on a hyperplane orthogonal to sensitive features. Buyl et al. [18] disentangle the node embedding from sensitive features via enforcing the prior distribution to encode sensitive information as strongly as possible. Different from group fairness, individual fairness is much less studied on graphs. Kang et al. [104] propose to reduce bias in all three stages of a graph mining pipeline (i.e., pre-processing, processing, and post-processing [18]). However, their framework is for unattributed networks and does not allow end-to-end training. To our best knowledge, we are the first to study individual fairness for GNNs on attributed networks.

4.1.7 Conclusion

Due to the superior learning capability, GNNs have been widely adopted to handle graph-structured data for various decision-making scenarios. However, leaving more and more decisions and judgments to GNNs raises societal concerns as the GNNs often do not have fairness considerations. Although some recent works have aimed to improve the fairness of GNNs for certain subgroups defined by a protected attribute, the fairness notion of GNNs at a much finer granularity (i.e., individual fairness) remains under-explored. To promote individual fairness, existing studies often need to rely on the Lipschitz condition to guarantee similar individuals have similar outcomes. In this paper, we argue the conventional definition of individual fairness based on the Lipschitz condition may have some potential issues w.r.t. the subtle Lipschitz constant and the uncalibrated distance metrics. Thus, we refine the definition of individual fairness from a ranking perspective, such that for each individual, the two ranking lists that encode its similarity with other individuals in the input space and output space are consistent with each other. To achieve this goal, we develop a novel plug-and-play framework REDRESS, which encapsulates the GNN model utility optimization and ranking-based individual fairness optimization in a joint framework and enables end-to-end training. To demonstrate the effectiveness of our proposed framework REDRESS, we present empirical evaluations on different real-world graphs under two downstream tasks. The experimental results imply that REDRESS outperforms the state-of-the-art individual fairness promoting approaches without jeopardizing the prediction performance. REDRESS is not only restricted on GNNs but can be extended to other graph mining models and tasks as future works.

4.2 Modeling and Mitigating Data Bias for Graph Neural Networks

4.2.1 Introduction

Attributed networks are ubiquitous in a plethora of web-related applications including online social networking [184], web advertising [226], and news recommendation [160]. To better understand these networks, various graph mining algorithms have been proposed. In particular, the recently emerged Graph Neural Networks (GNNs) have demonstrated superior capability of analyzing attributed networks in various tasks, such as node classification [111, 188] and link prediction [234, 110]. Despite the superior performance of GNNs, they usually do not consider fairness issues in the learning process [37]. Extensive research efforts have shown that many recently proposed GNNs [37, 176, 216] could make biased decisions towards certain demographic groups determined by sensitive attributes such as gender [59] and political ideology [152]. For example, e-commerce platforms generate a huge amount of user activity data, and such data is often constructed as a large attributed network in which entities (e.g., buyers, sellers, and products) are nodes while activities between entities (e.g.., purchasing and reviewing) are edges. To prevent potential losses, fraud entities (e.g., manipulated reviews and fake buyers) need to be identified on these platforms, and GNNs have become the prevalent solution to achieve such goal [54, 129]. Nevertheless, GNNs may have the risk of using sensitive information (e.g., race and gender) to identify fraud entities, yielding inevitable discrimination. Therefore, it is a crucial problem to mitigate bias in these network-based applications.

Various efforts have been made to mitigate the bias exhibited in graph mining algorithms. For example, in online social networks, random walk algorithms can be modified via improving the appearance rate of minorities [162, 17]; adversarial learning is another popular approach, which aims to learn node embeddings that are not distinguishable on sensitive attributes [13, 141]. Some recent efforts have also been made to mitigate bias in the outcome of GNNs. For example, adversarial learning can also be adapted to GNNs for outcome bias mitigation [37]. Nevertheless, existing approaches to debias GNN outcomes are tailored for a specific GNN model on a certain downstream task. In practical scenarios, different applications could adopt different GNN variants [111, 74], and it is costly to train and fine-tune the debiasing approaches based on diverse GNN backbones. As a consequence, to mitigate bias more efficiently for different GNNs and tasks, developing a one-size-fits-all approach becomes highly desired. Then the question is: how can we perform debiasing regardless of specific GNNs and downstream tasks? Considering that a model trained on biased datasets also tends to be biased [233, 37, 9], directly debiasing the dataset itself can be a straightforward solution. There are already debiasing approaches modifying original datasets via perturbing data distributions or reweighting the data points in the dataset [195, 102, 19]. These approaches obtain less biased datasets, which help to mitigate bias in learning algorithms. In this regard, considering that debiasing for different GNNs is costly, it is also highly desired to mitigate the bias in attributed networks before they are fed into GNNs. Nevertheless, to the best of our knowledge, despite its fundamental importance, no existing literature has taken such a step.

In this paper, we make an initial investigation on debiasing attributed networks towards more fair GNNs. Specifically, we tackle the following challenges. (1) ***Data Bias Modeling***. Traditionally, bias modeling is coupled with the outcome of a specific GNN [37]. Based on the GNN outcome, bias can be modeled via different fairness notions, e.g., *Statistical Parity* [58] and *Equality of Opportunity* [77], to determine whether the outcome is discriminatory towards some specific demographic groups. Nevertheless, if debiasing is carried out directly based on the input attributed networks instead of the GNN outcome, the first and foremost challenge is how to appropriately model such data bias. (2) ***Multi-Modality Debiasing***. In fact, attributed networks contain both graph structure and node attribute information. Correspondingly, bias may exist with diverse formats across different data modalities. In this regard, how to debias attributed networks that have different data modalities is the second challenge that needs to be tackled. (3) ***Model-Agnostic Debiasing***. Existing GNN debiasing approaches require the outcome of a specific GNN for objective function optimization during training. Different from these approaches, model-agnostic debiasing for GNNs should not rely on any specific GNN, as our goal is to develop a one-size-fits-all data debiasing approach to benefit various GNNs. Clearly, such model-agnostic debiasing could have better generalization capability but becomes much more difficult compared with the model-oriented GNN debiasing approaches. Nevertheless, the ultimate goal of debiasing is still to ensure the GNN outcome does not exhibit any discrimination. Such a contradiction poses the challenge of how to properly formulate a debiasing objective that can be universally applied to different GNNs in downstream tasks.

To tackle the challenges above, we present novel data bias modeling approaches and a principled debiasing framework named EDITS (modEling anD mItigating daTa biaS) to achieve model-agnostic attributed network debiasing for GNNs. Specifically, we first carry out preliminary analysis to illustrate how bias exists in the two data modalities of an attributed network (i.e., node attributes and network structure) and affects each other in the information propagation of GNNs. Then, we formally define *attribute bias* and *structural bias*, together with the corresponding metrics for data bias modeling. Besides, we formulate the problem of debiasing attributed networks for GNNs, and propose a novel framework named EDITS for bias mitigation. It is worth mentioning that EDITS is model-agnostic for GNNs. In other words, our goal is to obtain less biased attributed networks for the input of any GNNs. Finally, empirical evaluations on both synthetic and real-world datasets corroborate the validity of the proposed bias metrics and the effectiveness of EDITS. Our contributions are summarized as:

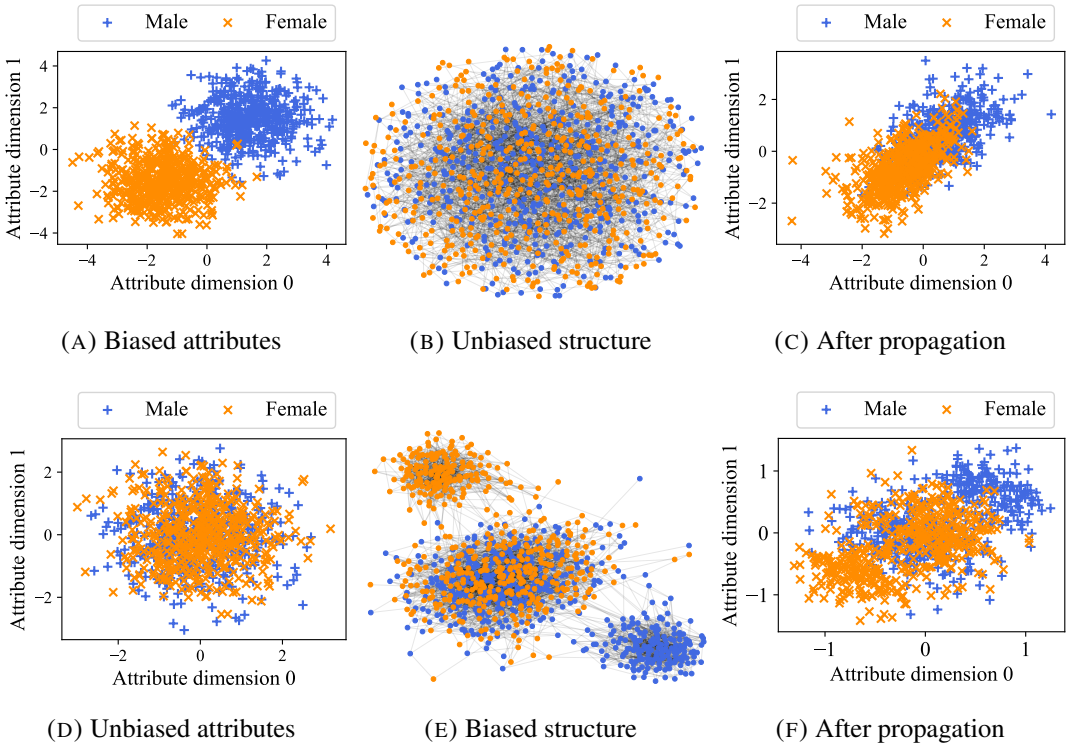
- **Problem Formulation.** We formulate and make an initial investigation on a novel research problem: debiasing attributed networks for GNNs based on the analysis of the information propagation mechanism.
- **Metric and Algorithm Design.** We design novel bias metrics for attributed networks, and propose a model-agnostic debiasing framework named EDITS to mitigate the bias in attributed networks before they are fed into GNNs.
- **Experimental Evaluation.** We conduct comprehensive experiments on both synthetic and real-world datasets to verify the validity of the proposed bias metrics and the effectiveness of the proposed framework EDITS.

4.2.2 Preliminary Analysis

We provide two exemplary cases to show how the two data modalities of an attributed network (i.e., node attribute and network structure) introduce bias in information propagation – the most common operation in GNNs. These two cases also bring insights on tackling the three challenges mentioned in Section 4.2.1. Specifically, two synthetic datasets are generated with either biased node attribute or network structure, and then attributes are propagated across the network structure to show how bias is introduced in GNNs. Here we consider the attribute distribution difference between different demographic groups as the bias in attribute, while the group membership distribution difference of the neighbors for nodes between different demographic groups is regarded as the bias in network structure. Such bias in attribute and structure can be regarded as the bias that existed in two data modalities in an attributed network. It should be noted that using distribution difference to define the level of bias is consistent with many algorithmic fairness studies [58, 233], Now we explain how the synthetic datasets are generated. We assume the *sensitive attribute* is gender, and 1,000 nodes are generated with half males (blue) and half females (orange) for both cases. In addition to the sensitive attribute, each node is with an extra two-dimensional attribute vector, which will be initialized and fed as input for information propagation. To introduce bias to either of the data modalities, different strategies are adopted to generate the attribute vector and the network structure. To study how the two data modalities introduce bias in information propagation, we compare the distribution difference of attributes between groups before and after the propagation mechanism in GCN [111].

Case 1: Biased attributes and unbiased structure. In this case, we generate biased two-dimensional attribute vectors for nodes from the two groups (i.e., males and females) and unbiased network structure. Specifically, biased attributes at each dimension is generated independently with Gaussian distribution $\mathcal{N}$(-1.5, 1^2) for female and $\mathcal{N}$(1.5, 1^2) for male. The distributions are shown in Fig. 4.6a. We then introduce how an unbiased network structure is generated. For each node in an unbiased network structure, the expected membership ratio of any group in its neighbor node set should be independent of the membership of the node itself. In this regard, we generate unbiased network structure via *random graph* model with edge formation probability as 2×10^{-3}. The visualization of the network is presented in Fig. 4.6b. The attribute distribution after information propagation according to the network structure is shown in Fig. 4.6c. Comparing Fig. 4.6a (attribute distribution before propagation) with 4.6c (attribute distribution after propagation), we can see the unbiased structure helps mitigate the original attribute bias after attributes are propagated according to the network structure. This not only implies that the attribute distribution difference between groups is a vital source of bias, but also demonstrates that unbiased structure helps mitigate bias in attributes after the information propagation process.

Case 2: Unbiased attributes and biased structure. In this case, unbiased attributes are generated independently at each dimension with $\mathcal{N}$(0, 1^2) for both males and females. The distributions are shown in Fig. 4.6d. The biased network structure is generated as follows. For each node, we sum up its attribute values. Then, we rank all nodes in descending order according to the summation of attribute values. After that, given a threshold integer t, for the top-ranked t males and bottom-ranked t females, we assume that they form two separated communities. The two communities are shown as the bottom right community (males) and



(A) Biased attributes (B) Unbiased structure (C) After propagation

(D) Unbiased attributes (E) Biased structure (F) After propagation

FIGURE 4.6. Two exemplary cases illustrating how bias in the two data modalities of an attributed network introduce bias in GNN information propagation. Here (c) is the node attribute distribution after propagation with biased node attributes (a) and unbiased network structure (b); while (f) is the attribute distribution after propagation with unbiased node attributes (d) and biased network structure (e).

the upper left community (females) in Fig. 4.6e. We generate edges via *random graph* model with edge formation probability as 5×10^{-2} within each community. Similarly, the rest nodes form the third community via *random graph* model with edge formation probability as 1×10^{-2}. We also generate edges between nodes from the male (or female) community and the third community with the probability of 2×10^{-4}. In this way, we introduce bias in network structure. The final network is presented in Fig. 4.6e. The attribute distribution after propagation according to the network structure is shown in Fig. 4.6f. Comparing Fig. 4.6d with 4.6f, we find that even if the original attributes are unbiased, the biased structure still turns the attributes into biased ones after information propagation. This implies that the bias contained in the network structure is also a significant source of bias.

Based on the discussions, we draw three preliminary conclusions to help us tackle the challenges in Section 4.2.1. (1) For ***Data Bias Modeling***, bias in attributes can be modeled based on the difference of attribute distribution between two groups. Also, bias in network structure can be modeled based on the difference of attribute distribution between two groups after information propagation. (2) For ***Multi-Modality Debiasing*** in an attributed network, at least two debiasing processes should be carried out targeting the two data modalities (i.e., attributes and structure). (3) For ***Model-Agnostic Debiasing***, if the attribute distributions between groups can be less biased both before and after information propagation, the learned

node representations tend to be indistinguishable between groups. Then GNNs trained on such data could also be less biased.

4.2.3 Modeling Data Bias for GNNs

In this section, we define *attribute bias* and *structural bias* in attributed networks together with their metrics. For the sake of simplicity, we focus on binary sensitive attribute and leave the generalization to non-binary cases in the online version[8]. Theoretical analysis of our proposed metrics is also presented in the online version.

4.2.3.1 Preliminaries

In this paper, without further specification, bold uppercase letters (e.g., $\mathbf{X}$), bold lowercase letters (e.g., $\mathbf{x}$), and normal lowercase letters (e.g., x) represent matrices, vectors, and scalars, respectively. For any matrix, e.g., $\mathbf{X}$, we use $\mathbf{X}_i$ denote its i-th row.

Let $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ be an undirected attributed network. Here $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix, and $\mathbf{X} \in \mathbb{R}^{N\times M}$ is the node attribute matrix, where N is the number of nodes and M is the attribute dimension. Let a diagonal matrix $\mathbf{D}$ be the degree matrix of $\mathbf{A}$, where its (i,i)-th entry $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{i,j}$, and $\mathbf{D}_{i,j} = 0$ $(i \neq j)$. $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the graph Laplacian matrix. Denote the normalized adjacency matrix and the normalized Laplacian matrix as $\mathbf{A}_{\text{norm}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ and $\mathbf{L}_{\text{norm}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$. $|.|$ is the absolute value operator.

4.2.3.2 Definitions of Bias

We consider two types of bias on attributed networks, i.e., attribute bias and structural bias. We first define attribute bias as follows.

DEFINITION 4.2.1. ***Attribute bias.*** *Given an undirected attributed network $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ and the group indicator (w.r.t. the sensitive attribute) for each node $\mathbf{s} = [s_1, s_2, ..., s_N]$, where $s_i \in \{0, 1\}$ $(1 \leq i \leq N)$. For any attribute, if its value distributions between different demographic groups are different, then attribute bias exists in $\mathcal{G}$.*

Besides, as shown in the second example in Section 4.2.2, bias can also emerge after attributes are propagated in the network even when original attributes are unbiased. Therefore, an intuitive idea to identify structural bias is to check whether information propagation in the network introduces or exacerbates bias [87]. Formally, we define structural bias on attributed networks as follows.

DEFINITION 4.2.2. ***Structural bias.*** *Given an undirected attributed network $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ and the corresponding group indicator (w.r.t. sensitive attribute) for each node $\mathbf{s} = [s_1, s_2, ..., s_N]$, where $s_i \in \{0, 1\}$ $(1 \leq i \leq N)$. For the attribute values propagated w.r.t. $\mathbf{A}$, if their distributions between different demographic groups are different at any attribute dimension, then structural bias exists in $\mathcal{G}$.*

Apart from these definitions, it is also necessary to quantitatively measure the attribute bias and structural bias. In the sequel, we introduce our proposed metrics for the two types of bias.

[8]See online version herehttps://dl.acm.org/doi/abs/10.1145/3485447.3512173 for supplementary discussion and experimental results.

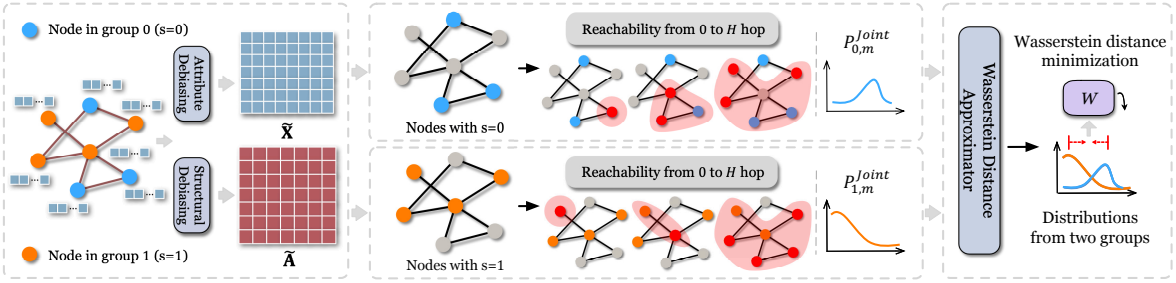


FIGURE 4.7. An illustration of EDITS with $H = 2$: Wasserstein Distance Approximator yields the approximated Wasserstein distance between $P^{Joint}_{0,m}$ and $P^{Joint}_{1,m}$; Attribute Debiasing and Structural Debiasing are optimized towards less biased $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{A}}$.

4.2.3.3 Bias Metrics

Here we take the first step to define metrics for both *attribute bias* and *structural bias* for an undirected attributed network $\mathcal{G}$.

Attribute bias metric. Let $\mathbf{X}_{\text{norm}} \in \mathbb{R}^{N \times M}$ be the normalized attribute matrix. For the m-th dimension ($1 \leq m \leq M$) of $\mathbf{X}_{\text{norm}}$, we use $\mathcal{X}^0_m$ and $\mathcal{X}^1_m$ to denote attribute value set for nodes with $s_i = 0$ and $s_i = 1$ ($1 \leq i \leq N$). Then, attributes of all nodes can be divided into tuples: $\mathcal{X}_{total} = \{(\mathcal{X}^0_1, \mathcal{X}^1_1), (\mathcal{X}^0_2, \mathcal{X}^1_2), ..., (\mathcal{X}^0_M, \mathcal{X}^1_M)\}$. We measure attribute bias with Wasserstein-1 distance [190] between the distributions of the two groups:

$$b_{\text{attr}} = \frac{1}{M} \sum_m W(pdf(\mathcal{X}^0_m), pdf(\mathcal{X}^1_m)). \tag{4.10}$$

Here $pdf(\cdot)$ is the probability density function for a set of values, and $W(.,.)$ is the Wasserstein distance between two distributions. Intuitively, b_{attr} describes the average Wasserstein-1 distance between attribute distributions of different groups across all dimensions. It should be noted that taking the distribution difference between demographic groups as the indication of bias is in align with many existing algorithmic fairness studies [233, 13, 37].

Structural bias metric. As illustrated in Section 4.2.2, the key mechanism of GNNs is information propagation, during which the structural bias could be introduced. Let $\mathbf{P}_{\text{norm}} = \alpha \mathbf{A}_{\text{norm}} + (1 - \alpha)\mathbf{I}$. Here $\mathbf{P}_{\text{norm}}$ can be regarded as a normalized adjacency matrix with re-weighted self-loops, where $\alpha \in [0, 1]$ is a hyper-parameter. Before measuring structural bias, we define the *propagation matrix* $\mathbf{M}_H \in \mathbb{R}^{N \times N}$ as:

$$\mathbf{M}_H = \beta_1 \mathbf{P}_{\text{norm}} + \beta_2 \mathbf{P}^2_{\text{norm}} + ... + \beta_H \mathbf{P}^H_{\text{norm}}, \tag{4.11}$$

where β_h ($1 \leq h \leq H$) is re-weighting parameters. The rationale behind the formulation above is to measure the aggregated reaching likelihood from each node to other nodes within a distance of H. To achieve localized effect for each node, a desired choice is to let $\beta_1 \geq \beta_2 \geq ... \geq \beta_H$, i.e., emphasizing short-distance terms and reducing the weights of long-distance terms. For example, assume $H = 3$, then the value $(\mathbf{M}_3)_{i,j}$ is the aggregated reaching likelihood from node i to node j within 3 hops with re-weighting parameters being β_1, β_2 and β_3. Also, given attributes $\mathbf{X}_{\text{norm}}$, we define the *reachability matrix* $\mathbf{R} \in \mathbb{R}^{N \times M}$ as $\mathbf{R} = \mathbf{M}_H \mathbf{X}_{\text{norm}}$. Intuitively, $\mathbf{R}_{i,m}$ is the aggregated reachable attribute value for attribute m of node i. We utilize $\mathcal{R}^0_m$ and $\mathcal{R}^1_m$ to represent the set of values of the m-th dimension in $\mathbf{R}$

for nodes with $s_i = 0$ and $s_i = 1$ ($1 \leq i \leq N$). The entries in $\mathbf{R}$ can also be divided into tuples according to attribute dimensions: $\mathcal{R}_{total} = \{(\mathcal{R}_1^0, \mathcal{R}_1^1), (\mathcal{R}_2^0, \mathcal{R}_2^1), ..., (\mathcal{R}_M^0, \mathcal{R}_M^1)\}$. We define structural bias as:

$$b_{\text{stru}} = \frac{1}{M} \sum_m W(pdf(\mathcal{R}_m^0), pdf(\mathcal{R}_m^1)). \tag{4.12}$$

Here b_{stru} is defined in a similar way as b_{attr}, except that the former uses $\mathcal{R}_m^0$ and $\mathcal{R}_m^1$ instead of $\mathcal{X}_m^0$ and $\mathcal{X}_m^1$. In this way, structural bias b_{stru} describes the average difference between aggregated attribute distributions of different groups after several rounds of propagation.

4.2.3.4 Problem Statement

Based on the definitions and metrics in Section 4.2.3.2 and 4.2.3.3, we argue that if both b_{attr} and b_{stru} are reduced, bias in an attributed network can be mitigated. As a result, if GNNs are trained on such data, the bias issues in downstream tasks could also be alleviated. Formally, we define the debiasing problem as follows.

PROBLEM 4.2.1. ***Debiasing attributed networks for GNNs.*** *Given an attributed network $\mathcal{G}$ = ($\mathbf{A}$, $\mathbf{X}$), our goal is to debias $\mathcal{G}$ by reducing b_{attr} and b_{stru} to obtain $\tilde{\mathcal{G}}$ = ($\tilde{\mathbf{A}}$, $\tilde{\mathbf{X}}$), so that the bias of GNNs trained on $\tilde{\mathcal{G}}$ is mitigated. The debiasing is independent of any specific GNNs.*

4.2.4 Mitigating Data Bias for GNNs

In this section, we discuss how to tackle Problem 4.2.1 with our proposed framework EDITS. We focus on the binary sensitive attribute for the sake of simplicity and discuss the extension later. We first present an overview of EDITS, followed by the formulation of the objective function. Finally, we present the optimization process.

4.2.4.1 Framework Overview

An overview of the proposed framework EDITS is shown in Fig. 4.7. Specifically, EDITS consists of three modules. The parameters of these three modules are optimized alternatively during training.

- ***Attribute Debiasing.*** This module learns a debiasing function $g_{\boldsymbol{\theta}}$ with learnable parameter $\boldsymbol{\theta} \in \mathbb{R}^M$. The debiased version of $\mathbf{X}$ is obtained as output where $\tilde{\mathbf{X}} = g_{\boldsymbol{\theta}}(\mathbf{X})$.
- ***Structural Debiasing.*** This module outputs $\tilde{\mathbf{A}}$ as the debiased $\mathbf{A}$. Specifically, $\tilde{\mathbf{A}}$ is initialized with $\mathbf{A}$ at the beginning of the optimization process. The entries in $\tilde{\mathbf{A}}$ are optimized via gradient descent with clipping and binarization.
- ***Wasserstein Distance Approximator.*** This module learns a function f for each attribute dimension. Here f is utilized to estimate the Wasserstein distance between the distributions of different groups for any attribute dimension.

4.2.4.2 Objective Function

In this subsection, we introduce the details of our framework. Following the Definition 4.2.1 and Definition 4.2.2, our goal is to reduce b_{attr} and b_{stru} simultaneously. For the ease of

understanding, we first only consider the m-th attribute dimension as an example, and then extend it to all M dimensions to obtain our final objective function.

Let $P_{0,m}$ and $P_{1,m}$ be the value distribution at the m-th attribute dimension in $\mathbf{X}$ for nodes with sensitive attribute $s = 0$ and $s = 1$, respectively. Denote $x_{0,m} \sim P_{0,m}^{(h)}$ and $x_{1,m} \sim P_{1,m}^{(h)}$ as two random variables drawn from the two distributions. Assume that we have a function $g_{\theta_m} : \mathbb{R} \to \mathbb{R}$ to mitigate attribute bias, where $1 \leq m \leq M$. For the m-th dimension, we denote $x_{0,m}^{(0)} = g_{\theta_m}(x_{0,m}) \sim P_{0,m}^{(0)}$ and $x_{1,m}^{(0)} = g_{\theta_m}(x_{1,m}) \sim P_{1,m}^{(0)}$ as the debiasing results for $x_{0,m}$ and $x_{1,m}$, respectively. Here the superscript (0) indicates that no information propagation is performed in the debaising process. Correspondingly, when such operation is extended to all M dimensions, we will have the debiased attribute matrix $\tilde{\mathbf{X}}$. Apart from the goal of mitigating attribute bias, we also want to mitigate structural bias. Let $\tilde{\mathbf{A}}$ be the adjacency matrix from the debiased network structure, and $\tilde{\mathbf{P}}_{\text{norm}}$ denotes the normalized $\tilde{\mathbf{A}}$ with re-weighted self-loops. Information propagation with h hops using the debiased adjacency matrix could be expressed as $\tilde{\mathbf{P}}_{\text{norm}}^{h}\tilde{\mathbf{X}}$, where $1 \leq h \leq H$. Let $P_{0,m}^{(h)}$ and $P_{1,m}^{(h)}$ be the value distribution at the m-th column of $\tilde{\mathbf{P}}_{\text{norm}}^{h}\tilde{\mathbf{X}}$ for nodes with sensitive attribute $s = 0$ and $s = 1$, respectively. Denote $x_{0,m}^{(h)} \sim P_{0,m}^{(h)}$ and $x_{1,m}^{(h)} \sim P_{1,m}^{(h)}$ as two random variables drawn from the two distributions. We hope that $\tilde{\mathbf{A}}$ could mitigate structural bias. We combine attribute and structural debiasing as below.

Based on the random variables $x_{0,m}^{(0)}$ to $x_{0,m}^{(H)}$ and $x_{1,m}^{(0)}$ to $x_{1,m}^{(H)}$, we have $(H+1)$-dimensional vectors $\mathbf{x}_{0,m} = [x_{0,m}^{(0)}, x_{0,m}^{(1)}, ..., x_{0,m}^{(H)}]$ and $\mathbf{x}_{1,m} = [x_{1,m}^{(0)}, x_{1,m}^{(1)}, ..., x_{1,m}^{(H)}]$ following the joint distribution $P_{0,m}^{Joint}$ and $P_{1,m}^{Joint}$, respectively. To reduce both b_{attr} and b_{stru} at the m-th dimension, our goal is to minimize the Wasserstein distance between $P_{0,m}^{Joint}$ and $P_{1,m}^{Joint}$, which is formulated as $\min_{\theta_m, \tilde{\mathbf{A}}} W(P_{0,m}^{Joint}, P_{1,m}^{Joint})$. $W(P_{0,m}^{Joint}, P_{1,m}^{Joint})$ can be expressed as

$$W(P_{0,m}^{Joint}, P_{1,m}^{Joint}) = \inf_{\gamma \in \Pi(P_{0,m}^{Joint}, P_{1,m}^{Joint})} \mathbb{E}_{(\mathbf{x}_{0,m}, \mathbf{x}_{1,m}) \sim \gamma}[\|\mathbf{x}_{0,m} - \mathbf{x}_{1,m}\|_1]. \tag{4.13}$$

Here $\Pi(P_{0,m}^{Joint}, P_{1,m}^{Joint})$ represents the set of all joint distributions $\gamma(\mathbf{x}_{0,m}, \mathbf{x}_{1,m})$ whose marginals are $P_{0,m}^{Joint}$ and $P_{1,m}^{Joint}$, respectively. After considering all the M dimensions, the overall objective is

$$\min_{\boldsymbol{\theta}, \tilde{\mathbf{A}}} \frac{1}{M} \sum_{1 \leq m \leq M} W(P_{0,m}^{Joint}, P_{1,m}^{Joint}). \tag{4.14}$$

It is non-trivial to optimize Eq. (4.14) as the infimum is intractable. Therefore, in the next subsection, we show how to convert it into a tractable optimization problem through approximation, which enables end-to-end gradient-based optimization.

4.2.4.3 Framework Optimization

In this subsection, we introduce our optimization algorithm. For simplicity, first we still use the m-th attribute dimension in $\mathbf{X}$ to illustrate the idea. Considering the infimum in Wasserstein distance computation is intractable, we apply the Kantorovich-Rubinstein duality [189] to

convert the problem of Eq. (4.13) as:

$$W(P_{0,m}^{Joint}, P_{1,m}^{Joint}) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{\mathbf{x}_{0,m} \sim P_{0,m}^{Joint}}[f(\mathbf{x}_{0,m})] - \mathbb{E}_{\mathbf{x}_{1,m} \sim P_{1,m}^{Joint}}[f(\mathbf{x}_{1,m})]. \tag{4.15}$$

Here $\|f\|_L \leq 1$ denotes that the supremum is taken over all 1-Lipschitz functions $f : \mathbb{R}^{H+1} \to \mathbb{R}$. The problem can be solved by learning a neural network as f. Nevertheless, it is worth noting that the 1-Lipschitz function is difficult to obtain during optimization. Therefore, here we relax $\|f\|_L \leq 1$ to $\|f\|_L \leq k$ (k is a constant). In this case, the left side of Eq. (4.15) also changes to $kW(P_{0,m}^{Joint}, P_{1,m}^{Joint})$. Then, the Wasserstein distance between $P_{0,m}^{Joint}$ and $P_{1,m}^{Joint}$ up to a multiplicative constant can be attained via:

$$\max_{f_m \in \mathcal{F}} \mathbb{E}_{\mathbf{x}_{0,m} \sim P_{0,m}^{Joint}}[f_m(\mathbf{x}_{0,m})] - \mathbb{E}_{\mathbf{x}_{1,m} \sim P_{1,m}^{Joint}}[f_m(\mathbf{x}_{1,m})], \tag{4.16}$$

where $\mathcal{F}$ denotes the set of all k-Lipschitz functions (i.e., $\|f_m\|_L \leq k$, $f_m \in \mathcal{F}$). Then, extending Eq. (4.16) to all M dimensions leads to our final objective function as:

$$\mathscr{L}_1 = \sum_{1 \leq m \leq M} \{\mathbb{E}_{\mathbf{x}_{0,m} \sim P_{0,m}^{Joint}}[f_m(\mathbf{x}_{0,m})] - \mathbb{E}_{\mathbf{x}_{1,m} \sim P_{1,m}^{Joint}}[f_m(\mathbf{x}_{1,m})]\}, \tag{4.17}$$

where $\{f_m : 1 \leq m \leq M\} \subset \mathcal{F}$. To model the function f in Eq. (4.17), a single-layered neural network serves as the *Wasserstein Distance Approximators* in Fig. 4.7 to approximate each f_m ($1 \leq m \leq M$), where the objective can be formulated as:

$$\max_{\{f_m : 1 \leq m \leq M\} \subset \mathcal{F}} \mathscr{L}_1 \ . \tag{4.18}$$

The weights of neural networks are clipped within $[-c, c]$ (c is a pre-defined constant), which has been proved to be a simple but effective way to enforce the Lipschitz constraint for every f_m [5]. For the *Attribute Debiasing* module in Fig. 4.7, we choose a linear function, i.e., $g_{\theta_m}(x_{s,m}) = \theta_m x_{s,m}$ ($s \in \{0, 1\}$). One advantage is that it acts as the role of feature re-weighting by assigning a feature weight for each attribute, which enables better interpretability for the debiased result. In matrix form, assume $\mathbf{\Theta}$ is a diagonal matrix with the m-th diagonal entry being θ_m, we have $\tilde{\mathbf{X}} = g_{\boldsymbol{\theta}}(\mathbf{X}) = \mathbf{X}\mathbf{\Theta}$. Then the optimization goal for *attribute debiasing* is:

$$\min_{\mathbf{\Theta}} \mathscr{L}_1 + \mu_1 \|\tilde{\mathbf{X}} - \mathbf{X}\|_F^2 + \mu_2 \|\mathbf{\Theta}\|_1, \tag{4.19}$$

where μ_1 and μ_2 are hyper-parameters. The second term ensures that the debiased attributes after feature re-weighting are close to the original ones (i.e., preserve as much information as possible). The third term controls the sparsity of re-weighting parameters. For the *Structural Debiasing* module in Fig. 4.7, $\tilde{\mathbf{A}}$ is optimized through:

$$\min_{\tilde{\mathbf{A}}} \mathscr{L}_1 + \mu_3 \|\tilde{\mathbf{A}} - \mathbf{A}\|_F^2 + \mu_4 \|\tilde{\mathbf{A}}\|_1 \quad s.t., \tilde{\mathbf{A}} = \tilde{\mathbf{A}}^\top. \tag{4.20}$$

where μ_3 and μ_4 are hyper-parameters. The second term ensures the debiased result $\tilde{\mathbf{A}}$ is close to the original structure $\mathbf{A}$. The third term enforces the debiased network structure is also sparse, which is aligned with the characteristics of real-world networks [96].

Optimization Strategy. To optimize function f, parameter $\mathbf{\Theta}$, and $\tilde{\mathbf{A}}$, we propose a gradient-based optimization approach for alternatively training[9]. First, for the optimization of f

[9]See online version for algorithmic routine.

w.r.t. Eq. (4.18), we directly utilize Stochastic Gradient Descent (SGD). Second, for the optimization of parameter $\mathbf{\Theta}$ w.r.t. Eq. (4.19), we adopt Proximal Gradient Descent (PGD). In the projection operation in PGD, we clip the parameters in $\mathbf{\Theta}$ within [0, 1]. Finally, to remove the most biased attribute channels, the z smallest weights in the diagonal of $\mathbf{\Theta}$ are masked with 0, where z is a pre-assigned hyper-parameter for attribute debiasing. Third, for the optimization of parameter $\tilde{\mathbf{A}}$ w.r.t. Eq. (4.20), we also adopt PGD with similar clipping strategy as the optimization of $\mathbf{\Theta}$. Finally, the proposed approach outputs $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{A}}$ after multiple epochs of optimization.

Edge Binarization. Here we introduce how we binarize the elements in $\tilde{\mathbf{A}}$ to indicate existence of edges. The basic intuition is to set a numerical threshold to determine the edge existence based on the entry-wise value change between $\tilde{\mathbf{A}}$ and $\mathbf{A}$. Specifically, for the "0" entries in $\mathbf{A}$, if the corresponding weight of any entry in $\tilde{\mathbf{A}}$ exceeds $r \cdot \max(\tilde{\mathbf{A}} - \mathbf{A})$, then we flip such entry from 0 to 1. Here r is a pre-set threshold for binarization, and $\max(\cdot)$ outputs the largest entry of a matrix. Similarly, for the "1" entries in $\mathbf{A}$, if the corresponding weight of any entry in $\tilde{\mathbf{A}}$ is reduced by a number exceeding $r \cdot |\min(\tilde{\mathbf{A}} - \mathbf{A})|$, then such entry should be flipped as 0. Here $\min(\cdot)$ gives the smallest entry of a matrix. To summarize, this operation aims to flip the entries with significant changes in value directly, and maintain other entries as their original values. Finally, the binarized matrix is assigned to $\tilde{\mathbf{A}}$ as the final outcome.

4.2.5 Experimental Evaluations

We conduct experiments on both real-world and synthetic datasets to evaluate the effectiveness of EDITS. In particular, we answer the following research questions. **RQ1:** How well can EDITS mitigate the bias in attributed networks together with the outcome of different GNN variants for the downstream task? **RQ2:** How well can EDITS balance utility maximization and bias mitigation compared with other debiasing baselines tailored for a specific GNN?

4.2.5.1 Downstream Task and Datasets

Downstream Task. We choose the widely adopted *node classification* task to assess the effectiveness of our proposed framework.

Datasets. We use two types of datasets in our experiments, including six real-world datasets and two synthetic datasets. Statistics of the real-world datasets can be found in the online version. We elaborate more details as follows: (1) *Real-world Datasets.* We use six real-world datasets, namely Pokec-z, Pokec-n [182, 37], UCSD34 [187], German Credit, Credit Defaulter, and Recidivism [3]. We first introduce the three web-related networks. *Pokec-z* and *Pokec-n* are collected from a popular social network in Slovakia. Here a node represents a user, and an edge denotes the friendship relation between two users [182]. We take "region" as the sensitive attribute, and the task is to predict the user working field. UCSD34 is a Facebook friendship network of the University of California San Diego [187]. Each node denotes a user, and edges represent the friendship relations between nodes. We take "gender" as the sensitive attribute, and the task is to predict whether a user belongs to a specific major. Users with incomplete information (e.g., missing attribute values) are filtered out from the three web networks above. Besides, we also adopt three networks beyond web-related data. In *German Credit*, nodes represent clients in a German bank, and edges are formed between clients if their

credit accounts are similar. With "gender" being the sensitive attribute, the task is to classify the credit risk of the clients as high or low. In *Recidivism*, nodes are defendants released on bail during 1990-2009. Nodes are connected based on the similarity of past criminal records and demographics. The task is to classify defendants into bail vs. no bail, with "race" being the sensitive attribute. In the *Credit Defaulter*, nodes are credit card users, and they are connected based on the pattern similarity of their purchases and payments. Here "age" is the sensitive attribute, and the task is to predict whether a user will default on credit card payment. (2) *Synthetic Datasets.* For the ablation study of EDITS, we use the two datasets generated in Section 4.2.2. One network has biased attributes and an unbiased structure, while the other network is on the opposite. We add eight extra attribute dimensions besides the two attribute dimensions for both datasets. The attribute values in the extra attribute dimensions are generated uniformly between 0 and 1. For labels, we compute the sum of the first two extra attribute dimensions. Then, we add Gaussian noise to the sum values, and rank them by the values in descending order. Labels of the top-ranked 50% individuals are set as 1, while the labels of the other 50% are set as 0. The task is to predict the labels.

4.2.5.2 Experimental settings

GNN Models. Here we adopt three popular GNN variants in our experiments: GCN [111], GraphSAGE [74], and GIN [222].

Baselines. Since there is no existing work directly debiasing network data for GNNs, here we choose two state-of-the-art GNN-based debiasing approaches for comparison, namely FairGNN [37] and NIFTY [3]. (1) *FairGNN*. It is a debiasing method based on adversarial training. A discriminator is trained to distinguish the representations between different demographic groups. The goal of FairGNN is to train a GNN that fools the discriminator for bias mitigation. (2) *NIFTY*. It is a recently proposed GNN-based debiasing framework. With counterfactual perturbation on the sensitive attribute, bias is mitigated via learning node representations that are invariant to the sensitive attribute. It should be noted that both of them take GNNs as their backbones in the downstream task. While on the other hand, EDITS attempts at debiasing attributed networks without referring to the output of downstream GNN models (i.e., EDITS is model-agnostic). The hyper-parameters of EDITS are tuned only based on our proposed bias metrics. Obviously, the debiasing performed by EDITS is with better generalization ability but more difficult compared with the model-oriented baselines.

Evaluation Metrics. We evaluate model performance from two perspectives: model utility and bias mitigation. Good performance means low bias and high model utility. We introduce the adopted metrics for model utility and bias mitigation: (1) *Model Utility Metrics*. For node classification, we use the area under the receiver operating characteristic curve (AUC) and F1 score as the indicator of model utility; (2) *Bias Mitigation Metrics*. We use two widely-adopted metrics Δ_{SP} and Δ_{EO} to show to what extent the bias in the output of different GNNs are mitigated [130, 9, 37]. For both metrics, a lower value means better bias mitigation.

4.2.5.3 Debiasing Attributed Network for GNNs

To answer **RQ1**, we first evaluate the effectiveness of EDITS in reducing the bias measured by the two proposed metrics and traditional bias metrics with different GNN backbones. The attribute and structural bias of the six real-world datasets before and after being debiased by

TABLE 4.7. Attribute and structural bias comparison between original networks and debiased ones from EDITS (in scale of $\times 10^{-3}$). The lower, the better. Best ones are marked in bold.

	Attribute Bias		Structural Bias	
	Vanilla	**EDITS**	**Vanilla**	**EDITS**
Pokec-z	0.43	**0.33** (-23.3%)	0.83	**0.75** (-9.64%)
Pokec-n	0.54	**0.42** (-22.2%)	1.03	**0.89** (-13.6%)
UCSD34	0.53	**0.48** (-9.43%)	0.68	**0.63** (-7.35%)
German	6.33	**2.38** (-62.4%)	10.4	**3.54** (-66.0%)
Credit	2.46	**0.56** (-77.2%)	4.45	**2.36** (-47.0%)
Recidivism	0.95	**0.39** (-58.9%)	1.10	**0.52** (-52.7%)

TABLE 4.8. Comparison on utility and bias mitigation between GNNs with original networks (denoted as Vanilla) and debiased networks (denoted as EDITS) as input. ↑ denotes the larger, the better; ↓ denotes the opposite. Best ones are in **bold**.

		GCN		GraphSAGE		GIN	
		Vanilla	**EDITS**	**Vanilla**	**EDITS**	**Vanilla**	**EDITS**
Pokec-z	**AUC** ↑	**67.83 ± 0.7%**	67.38 ± 0.3%	**68.00 ± 0.3%**	66.37 ± 0.7%	**66.74 ± 0.8%**	65.64 ± 0.5%
	F1 ↑	**61.95 ± 0.6%**	61.91 ± 0.1%	**61.58 ± 1.3%**	60.62 ± 0.6%	**61.55 ± 0.5%**	60.65 ± 1.2%
	Δ_{SP} ↓	5.70 ± 1.2%	**2.74 ± 0.9%**	7.10 ± 1.2%	**2.89 ± 0.4%**	5.20 ± 1.0%	**1.90 ± 1.3%**
	Δ_{EO} ↓	4.88 ± 1.3%	**2.87 ± 1.0%**	6.37 ± 0.8%	**2.54 ± 0.7%**	4.65 ± 1.1%	**2.09 ± 1.1%**
Pokec-n	**AUC** ↑	**63.24 ± 0.5%**	61.82 ± 0.9%	**64.07 ± 0.4%**	62.05 ± 0.6%	**62.53 ± 1.4%**	61.60 ± 1.4%
	F1 ↑	**54.32 ± 0.4%**	52.84 ± 0.3%	**53.45 ± 1.2%**	52.53 ± 0.1%	**52.62 ± 1.2%**	52.56 ± 1.0%
	Δ_{SP} ↓	3.36 ± 0.4%	**0.91 ± 0.9%**	3.85 ± 0.2%	**2.08 ± 1.2%**	5.90 ± 2.5%	**0.96 ± 0.5%**
	Δ_{EO} ↓	3.97 ± 1.6%	**1.10 ± 1.0%**	2.64 ± 0.3%	**1.82 ± 0.9%**	4.47 ± 3.7%	**0.47 ± 0.4%**
UCSD34	**AUC** ↑	**63.33 ± 0.3%**	62.43 ± 0.9%	62.62 ± 1.0%	**62.82 ± 2.4%**	62.57 ± 0.7%	**64.50 ± 0.9%**
	F1 ↑	94.16 ± 0.3%	**94.69 ± 0.1%**	94.00 ± 0.2%	**94.55 ± 0.1%**	92.24 ± 1.6%	**92.48 ± 0.5%**
	Δ_{SP} ↓	1.27 ± 0.4%	**0.27 ± 0.1%**	1.27 ± 0.5%	**0.35 ± 0.3%**	2.11 ± 1.3%	**0.36 ± 0.1%**
	Δ_{EO} ↓	1.40 ± 0.4%	**0.39 ± 0.1%**	1.40 ± 0.4%	**0.25 ± 0.3%**	2.32 ± 1.6%	**0.47 ± 0.4%**
German	**AUC** ↑	**74.46 ± 0.7%**	71.01 ± 1.3%	**75.28 ± 2.1%**	73.21 ± 0.5%	71.35 ± 1.7%	**71.51 ± 0.6%**
	F1 ↑	81.54 ± 0.9%	**82.43 ± 0.7%**	**81.52 ± 1.0%**	80.62 ± 1.5%	83.08 ± 0.9%	**83.78 ± 0.4%**
	Δ_{SP} ↓	43.14 ± 2.5%	**2.04 ± 1.3%**	26.83 ± 0.5%	**8.30 ± 3.1%**	18.55 ± 2.0%	**1.26 ± 0.7%**
	Δ_{EO} ↓	33.75 ± 0.4%	**0.63 ± 0.4%**	20.66 ± 3.0%	**3.75 ± 3.3%**	11.27 ± 3.5%	**2.87 ± 1.4%**
Credit	**AUC** ↑	**73.62 ± 0.3%**	70.16 ± 0.6%	74.99 ± 0.2%	**75.28 ± 0.5%**	**73.82 ± 0.4%**	72.06 ± 0.9%
	F1 ↑	**81.86 ± 0.1%**	81.44 ± 0.2%	82.31 ± 0.7%	**83.39 ± 0.3%**	82.11 ± 0.1%	**85.10 ± 0.7%**
	Δ_{SP} ↓	12.93 ± 0.1%	**9.13 ± 1.2%**	17.03 ± 3.3%	**12.25 ± 0.2%**	12.18 ± 0.3%	**8.79 ± 5.6%**
	Δ_{EO} ↓	10.65 ± 0.0%	**7.88 ± 1.0%**	15.31 ± 4.0%	**9.58 ± 0.1%**	9.48 ± 0.3%	**7.19 ± 3.8%**
Recidivism	**AUC** ↑	**86.91 ± 0.4%**	85.96 ± 0.3%	88.12 ± 1.4%	**88.15 ± 0.9%**	**82.40 ± 0.8%**	81.55 ± 1.5%
	F1 ↑	**78.30 ± 1.0%**	75.80 ± 0.5%	76.23 ± 2.8%	**76.30 ± 1.4%**	70.36 ± 1.9%	**71.09 ± 2.3%**
	Δ_{SP} ↓	7.89 ± 0.3%	**5.39 ± 0.2%**	2.42 ± 1.2%	**0.79 ± 0.5%**	9.97 ± 0.7%	**4.98 ± 0.9%**
	Δ_{EO} ↓	5.58 ± 0.2%	**3.36 ± 0.3%**	2.98 ± 2.2%	**1.01 ± 0.5%**	6.10 ± 1.2%	**5.47 ± 0.7%**

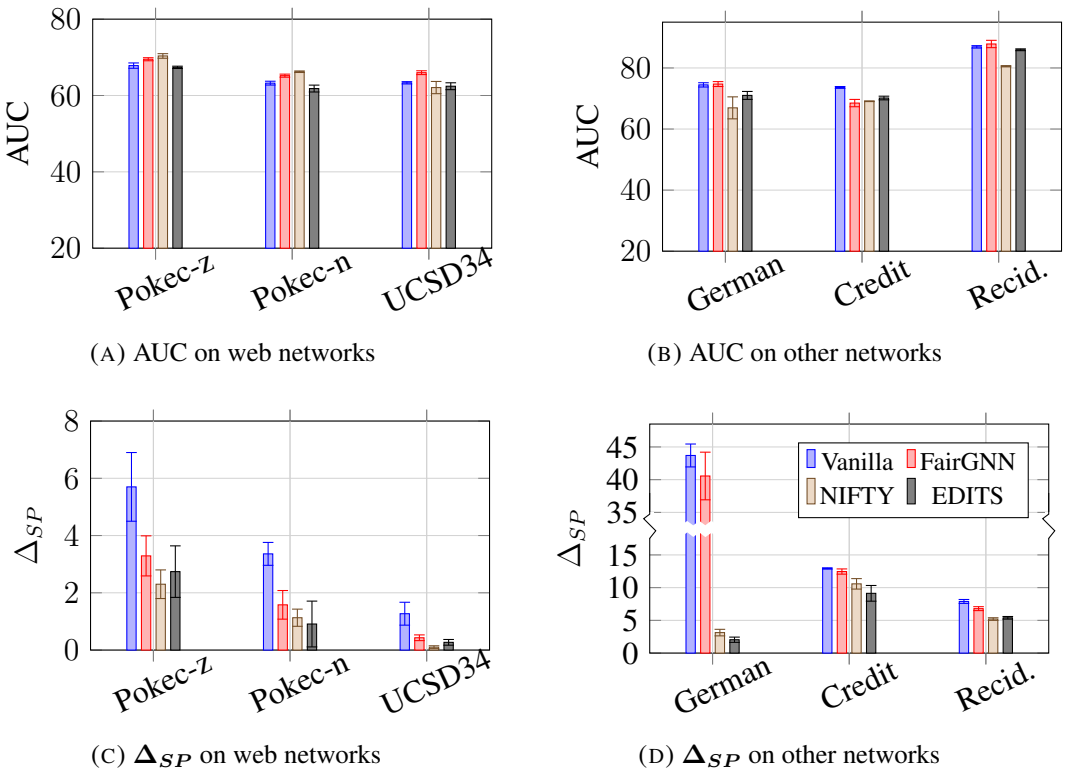
EDITS are shown in Table 4.7. The comparison on Δ_{SP} and Δ_{EO} between GNNs trained on debiased networks from EDITS and original networks is presented in Table 4.8. We make the following observations: (1) From the perspective of bias mitigation in the attributed network, EDITS demonstrates significant advantages over the vanilla approach as indicated by Table 4.7. This verifies the effectiveness of EDITS in reducing the bias existing in the attributed network data. (2) From the perspective of bias mitigation in the downstream task, we observe from Table 4.8 that EDITS achieves desirable bias mitigation performance with little utility sacrifice in all cases compared with GNNs with the original network as input (i.e., the vanilla one). This verifies that attributed networks debiased by EDITS can generally mitigate the bias in the outcome of different GNNs. (3) When comparing bias mitigation performance indicated by Table 4.7 and Table 4.8, we can find that the bias in the outcome of GNNs is also mitigated after EDITS mitigates attribute bias and structural bias in the attributed networks. Such consistency verifies the validity of our proposed metrics on measuring the bias that existed in the attributed networks.

4.2.5.4 Comparison with Other Debiasing Models

To answer **RQ2**, we then compare the balance between model utility and bias mitigation with other baselines based on a given GNN. Here we present the comparison of AUC and Δ_{SP} based on GCN in Fig. 4.8. Similar results can be obtained for other GNNs, which are omitted due to space limit. Experimental results include the performance of baselines and EDITS on the six real-world datasets. The following observations can be made: (1) From the perspective of model utility (indicated by Fig. 4.8a and Fig. 4.8b), EDITS and baselines achieve comparable results with the vanilla GCN. This implies that the debiasing process of EDITS preserves as much useful information for the downstream task as the original attributed network. (2) From the perspective of bias mitigation (indicated by Fig. 4.8c and Fig. 4.8d), all baselines achieve effective bias mitigation. Compared with debiasing in downstream tasks, debiasing the attributed network is more difficult due to the lack of supervision signals from GNN prediction. Observation can be drawn that the debiasing performance of EDITS is similar to or even better than that of the adopted baselines. This verifies the superior performance of EDITS on debiasing attributed networks for more fair GNNs. (3) From the perspective of balancing the model utility and bias mitigation, EDITS achieves comparable model utility with alternatives but exhibits better bias mitigation performance. Consequently, we argue that EDITS achieves superior performance on balancing the model utility and bias mitigation over other baselines.

4.2.5.5 Ablation Study

To evaluate the effectiveness of the two debiasing modules (i.e., attribute debiasing module and structural debiasing module) in EDITS, here we investigate how each of them individually contributes to bias mitigation under our proposed bias metrics and the traditional bias metrics in the downstream task. We choose GCN as the GNN model in our downstream task. For better visualization purposes, the two datasets showing large attribute bias and structural bias (i.e., *German* and *Credit*) are selected for experiments. Besides, to better demonstrate the functionality of the two debiasing modules, we also adopt the two synthetic datasets we mentioned in Section 4.2.2 (i.e., the network with only biased attributes and the network with only biased structure), which are further modified according to Section 4.2.5.1. Based on



(A) AUC on web networks

(B) AUC on other networks

(C) $\boldsymbol{\Delta_{SP}}$ on web networks

(D) $\boldsymbol{\Delta_{SP}}$ on other networks

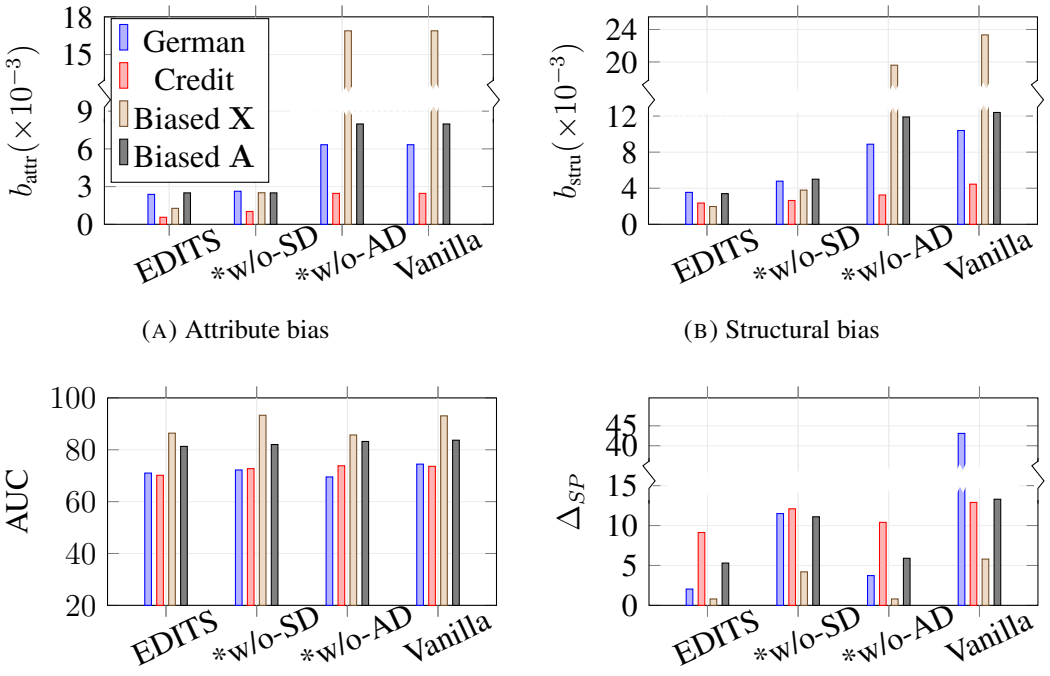
FIGURE 4.8. Performance comparison between EDITS and baselines on utility (AUC) and bias mitigation ($\boldsymbol{\Delta_{SP}}$).

the four selected datasets, four different variants of EDITS are tested, namely EDITS with both debiasing modules, EDITS without the structural debiasing module (i.e., *w/o-SD), EDITS without the attribute debiasing module (i.e., *w/o-AD), vanilla GCN model without debiased input (i.e., Vanilla). We present their performance of attribute bias, structural bias, AUC, and Δ_{SP} on the four datasets in Fig. 4.9. We make the following observations: (1) The value of *attribute bias* can be reduced with the attribute debiasing module of EDITS, which maintains the model utility (i.e., AUC) but reduces Δ_{SP} in the downstream task. (2) The value of *structural bias* can be reduced with both attribute debiasing and structural debiasing modules. With only structural debiasing, EDITS still maintains comparable model utility but reduces Δ_{SP} in the downstream task. (3) Although both attribute debiasing and structural debiasing module help mitigate *structural bias*, only debiasing the network structure achieves better bias mitigation performance on all four datasets compared with only debiasing the attributes as implied by Fig. 4.9d. This demonstrates the indispensability of the structural debiasing module in EDITS.

4.2.5.6 Extension to Non-Binary Sensitive Attributes

Here, we show how our proposed framework EDITS can be generalized to handle non-binary sensitive attributes. More specifically, we use a synthetic dataset to showcase the extension.

Synthetic Dataset Generation. Our goal here is to generate a synthetic attributed network with both biased node attributes and network structure, where nodes should come from at



(A) Attribute bias

(B) Structural bias

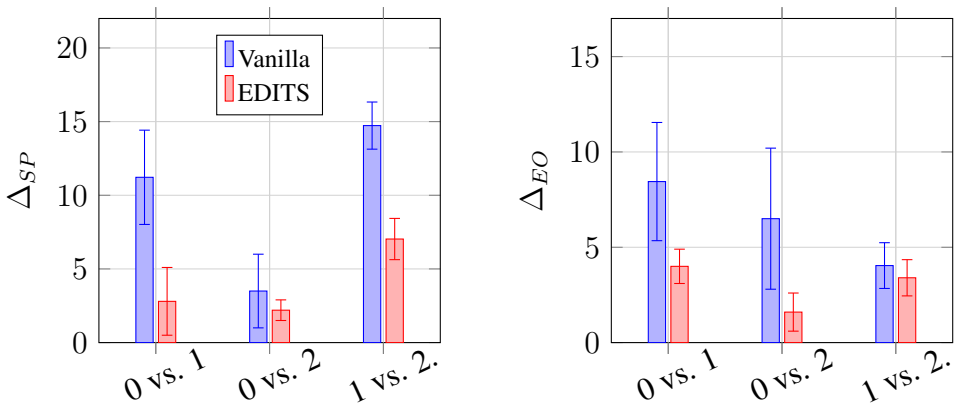
(C) Model utility performance

(D) Fairness performance

FIGURE 4.9. Performance EDITS and its variants on two real-world datasets and two synthetic datasets. EDITS denotes that both debiasing modules are included; *w/o-SD means EDITS without structural debiasing module; *w/o-AD means EDITS is without attribute debiasing module; Vanilla means applying GNN with the original attributed network as input.

least three different groups based on the sensitive attribute. We elaborate more details from three perspectives: biased network structure generation, biased node attribute generation, and node label generation. (1) *Biased Network Structure Generation.* We adopt a similar approach as presented in Fig. 4.6 to generate three communities with dense intra-community links but sparse inter-community links. (2) *Biased Node Attributes Generation.* We generate a ten-dimensional attribute vector for each node. The values at the first two dimensions are generated independently with Gaussian distribution $\mathcal{N}$(-1, 1^2), $\mathcal{N}$(0, 1^2), and $\mathcal{N}$(1, 1^2) for the nodes in the three communities, respectively. The attribute values for all other dimensions are generated with independent Gaussian Distribution $\mathcal{N}(0, 1^2)$. Besides, We generate a ternary variable $s \in \{0, 1, 2\}$ based on the node community membership for all nodes as an extra attribute dimension. Here the community membership is regarded as the sensitive attribute of nodes in this network. (3) *Node Label Generation.* We sum up the values at the first two unbiased attribute dimensions for all nodes, and then add Gaussian noise to the summation values. The summation values with noise are ranked in descending order. Labels of the top-ranked 50% nodes are set as 1, while the labels of the other 50% nodes are set as 0. The task is to predict the generated labels.

Framework Extension. To extend the proposed framework EDITS to handle non-binary sensitive attributes, the basic rationale is to encourage the function f_m introduced in Section 4.2.4.3 to help approximate the squared Wasserstein distance sum between all group



(A) Comparison of $\boldsymbol{\Delta_{SP}}$ (B) Comparison of $\boldsymbol{\Delta_{EO}}$

FIGURE 4.10. Comparison of $\boldsymbol{\Delta_{SP}}$ and $\boldsymbol{\Delta_{EO}}$ between vanilla and EDITS based on GCN for ternary sensitive attributes.

pairs based on ternary sensitive attribute. Therefore, we modify the $\mathscr{L}_1$ in Eq. (4.17) as

$$\tilde{\mathscr{L}_1} = \sum_{i,j} \sum_{m} \{\mathbb{E}_{\mathbf{x}_{i,m}}[f_m(\mathbf{x}_{i,m})] - \mathbb{E}_{\mathbf{x}_{j,m}}[f_m(\mathbf{x}_{j,m})]\}^2. \tag{4.21}$$

Here $1 \leq m \leq M$, and $i, j \in \{0, 1, 2\}$ $(i < j)$. $\mathbf{x}_{i,m}$ and $\mathbf{x}_{j,m}$ follows $P_{i,m}^{Joint}$ and $P_{j,m}^{Joint}$, respectively. The $\mathscr{L}_1$ in Eq. (4.18), (4.19), and (4.20) are repalced with $\tilde{\mathscr{L}_1}$. This helps approximate and minimize the squared Wasserstein distance sum between all group pairs.

Research Questions. Here we aim to answer two research questions. **RQ1:** Can EDITS mitigate the bias in the network dataset with ternary sensitive attributes? **RQ2:** Can EDITS achieve a good balance between mitigating bias and maintaining utility for GNN predictions with ternary sensitive attributes?

Evaluation Metrics. We introduce the metrics following the two research questions above. (1) For RQ1, to measure the bias in the network dataset, we adopt the b_{attr} and b_{stru} introduced in Sec. 4.2.3.3. (2) For RQ2, to measure the bias exhibited in GNN predictions, we adopt two traditional fairness metrics: Δ_{SP} and Δ_{EO}. Considering that these two metrics are designed only for binary sensitive attributes, Δ_{SP} and Δ_{EO} for each pair of groups are utilized to evaluate the fairness level of GNN predictions. Besides, AUC and F1 are adopted to evaluate the utility of GNN predictions.

Results Analysis. Results based on GCN are presented in Fig. 4.10 and Table 4.10, and similar observations can also be found on other GNN backbones. We evaluate the performance of EDITS from two perspectives. (1) *RQ1: the fairness level of the network dataset.* As presented in Table 4.10, b_{attr} and b_{stru} of the dataset are clearly reduced with EDITS. This verifies the effectiveness of EDITS on debiasing the attributed network data. (2) *RQ2: the balance between fairness and utility for GNN predictions.* As presented in Fig. 4.10, Δ_{SP} and Δ_{EO} for every group pair are reduced. This corroborates the effectiveness of EDITS on achieving more fair GNN predictions. At the same time, Table 4.10 indicates that the GNN with debiased input data still maintains similar utility performance compared with the GNN with vanilla input. This indicates that EDITS achieves a good balance between fairness and utility for GNN predictions.

TABLE 4.9. Parameter study for μ_1 and μ_3. The values of b_{attr} and b_{stru} are in scale of $\times 10^{-3}$.

μ_1	b_{attr}	F1(%)	$\Delta_{SP}(\%)$	μ_3	b_{stru}	F1(%)	$\Delta_{SP}(\%)$
1e2	6.33	81.69	35.3	**1e2**	10.2	82.26	34.2
1e1	5.02	80.69	19.9	**1e1**	9.97	80.89	25.0
1e0	3.74	80.28	7.76	**1e0**	9.81	79.77	14.1
1e-1	2.38	80.00	4.58	**1e-1**	4.89	79.46	3.96
1e-2	2.34	79.95	4.08	**1e-2**	3.53	78.93	3.26
1e-3	2.35	79.46	3.96	**1e-3**	3.34	78.89	2.76
1e-4	2.34	79.03	3.29	**1e-4**	3.29	78.37	2.06
1e-5	2.34	76.22	2.86	**1e-5**	3.22	78.06	2.00

TABLE 4.10. Comparison of fairness level and utility between the original synthetic network and the debiased one based on the ternary sensitive attributes. The values of b_{attr} and b_{stru} are in scale of $\times 10^{-3}$. Best ones are in bold.

Attribute Bias & Structural Bias Comparison						
	Group 0 v.s. 1		**Group 0 v.s. 2**		**Group 1 v.s. 2**	
	b_{attr}	b_{stru}	b_{attr}	b_{stru}	b_{attr}	b_{stru}
Vanilla	13.7	25.5	26.5	48.8	11.0	20.4
EDITS	**5.33**	**9.63**	**13.4**	**24.1**	**4.73**	**8.73**
Utility Comparison						
	AUC			**F1**		
Vanilla	**67.09 ± 0.3%**			**64.50 ± 0.6%**		
EDITS	67.05 ± 0.2%			62.91 ± 0.8%		

4.2.5.7 Parameter Study

Here we aim to study the sensitivity of EDITS w.r.t. hyper-parameters. Specifically, we show the parameter study of μ_1 and μ_3 on German dataset, but similar observations can also be found on other datasets. Here μ_1 and μ_3 control how much original information should be preserved from the original attributes and graph structure, respectively. We first vary μ_1 in the range of {1e2, 1e1, 1e0, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5} while fix other parameters as μ_2=1e-4, μ_3=1e-1, μ_4=1e-4; then we vary μ_1 in the same range with μ_1=1e-3, μ_2=1e-4, μ_4=1e-4. The results in Table 4.9 indicate that the trade-off between debiasing and utility performance is stable when μ_1 and μ_3 are in a wide range between 1e-3 and 1e-1. Therefore, it is safe to tune these parameters in a wide range without significantly affecting fairness and utility.

4.2.6 Related Work

Mitigating Bias in Machine Learning. Bias can be defined from a variety of perspectives in machine learning algorithms [55, 77, 8, 213, 133, 125]. Commonly used algorithmic bias notions can be broadly categorized into *group fairness* and *individual fairness* [58]. Group fairness emphasizes that algorithms should not yield discriminatory outcomes for any specific demographic groups [58]. Such groups are usually determined by sensitive attributes, e.g., gender or race [102]. Existing debiasing approaches work in one of the

three data flow stages, i.e., pre-processing, processing and post-processing stage. In pre-processing stage, a common method is to re-weight training samples from different groups to mitigate bias before model training [102]. Perturbing data distributions between groups is another popular approach to debias the data in the pre-processing stage [195]. In processing stage, a popular method is to add regularization terms to disentangle the outcome from sensitive attribute [164, 126] or minimize the outcome difference between groups [2]. Besides, utilizing adversarial learning to remove sensitive information from representations is also widely adopted [60]. In post-processing stage, bias in outcomes is usually mitigated by constraining the outcome to follow a less biased distribution [242, 77, 157, 119, 114]. Usually, all above-mentioned approaches are evaluated via measuring how much certain fairness notion is violated. *Statistical Parity* [58], *Equality of Opportunity*, *Equality of Odds* [77] and *Counterfactual Fairness* [115] are commonly studied fairness notions. Different from group fairness, individual fairness focuses on treating similar individuals similarly [58, 233]. The similarity can be given by oracle similarity scores from domain experts [118]. Most existing debiasing methods based on individual fairness work in the processing stage. For example, constraints can enforce similar predictions between similar instances [118, 101]. *Consistency* is a popular metric for individual fairness evaluation [118, 120].

Mitigating Bias in Graph Mining. Efforts have been made to mitigate bias in graph mining algorithms, where these works can be broadly categorized into either focusing on *group fairness* or *individual fairness*. For group fairness, adversarial learning can be adopted to learn less biased node representations that fool the discriminator [13, 37]. Rebalancing between groups is also a popular approach to mitigate bias [17, 62, 162, 185, 125]. For example, Rahman et al. mitigate bias via rebalancing the appearance rate of minority groups in random walks [162]. Projecting the embeddings onto a hyperplane orthogonal to the hyperplane of sensitive attributes is another approach for bias mitigation [147]. Compared with the vast amount of works on group fairness, only few works promote individual fairness in graphs. To the best of our knowledge, Kang et al. [104] first propose to systematically debias multiple graph mining algorithms based on individual fairness. Dong et al. [46] argue that for each individual, if the similarity ranking of others in the GNN outcome follows the same order of an oracle ranking given by domain experts, then people can get a stronger sense of fairness. Different from the above approaches, this paper proposes to directly debias the attributed networks in a model-agnostic way.

4.2.7 Conclusion

GNNs are increasingly critical in various applications. Nevertheless, there is an increasing societal concern that GNNs could yield discriminatory decisions towards certain demographic groups. Existing debiasing approaches are mainly tailored for a specific GNN. Adapting these methods to different GNNs can be costly, as they need to be fine-tuned. Different from them, in this paper, we propose to debias the attributed network for GNNs. With analysis of the source of bias existing in different data modalities, we define two kinds of bias with corresponding metrics, and formulate a novel problem of debiasing attributed networks for GNNs. To tackle this problem, we then propose a principled framework EDITS for model-agnostic debiasing. Experiments demonstrate the effectiveness of EDITS in mitigating the bias and maintaining the model utility.

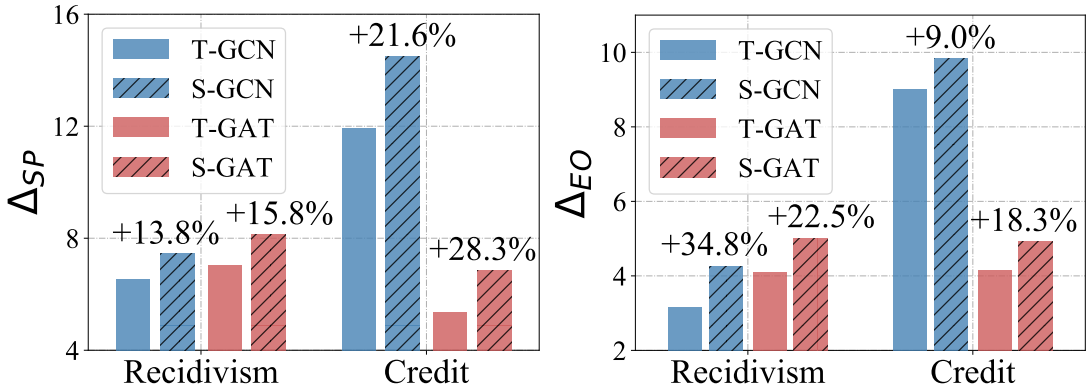
4.3 Fair Knowledge Distillation for Graph Neural Networks

4.3.1 Introduction

In recent years, Graph Neural Networks (GNNs) have shown satisfying performance in a plethora of real-world applications, e.g., medical diagnosis [166] and credit risk scoring [194], to name a few. In practice, the depth and the number of parameters of GNNs largely determine their expressive power [78], which directly influence their performances in various graph learning tasks [27]. Typically, deeper GNN layers enable the model to capture information that is multiple hops away from any node [111], while a larger number of learnable parameters enable GNN to fit more complex underlying data patterns [27]. However, in most cases, the inference efficiency of GNNs is inevitably degraded by the deep layers or the large number of parameters. Such low efficiency makes these GNNs inapplicable to be deployed on edge devices (e.g., mobile phones) with limited computational resources [78, 100].

Due to the problem above, it is necessary to compress those computationally expensive GNNs for deployment on edge devices. Knowledge Distillation (KD) is a common approach to compress GNNs but still maintains a similar level of prediction performance [225, 78, 100]. Here, the basic idea of KD is to let a light-weighted student model (as the compressed GNN) learn to mimic the behavior (e.g., output logits) of the teacher model (usually a computationally expensive GNN). However, most existing KD approaches do not have any fairness consideration over different demographic subgroups, and the optimized student model often preserves and even exaggerates the exhibited bias from the teacher GNN. Consequently, when the compressed model is deployed in real-world application scenarios, there could exist discrimination toward specific populations. Here we provide preliminary analysis based on two representative GNN knowledge distillation frameworks, namely CPF [225] and GraphAKD [78]. Specifically, we measure the exhibited bias in the widely-studied node classification task on two real-world datasets. Here Recidivism is a network of defendants [99, 3], while Credit is a network between bank clients [227, 3]. We adopt two traditional metrics, i.e., Δ_{SP} (measuring the level of bias under Statistical Parity [58]) and Δ_{EO} (measuring the level of bias under Equal Opportunity [77]), to measure the exhibited bias of GNN predictions. We present a comparison of the exhibited bias between teacher and student models in Fig. 4.11. Empirical results show that student models tend to yield more biased results compared with the teacher GNN model, which could be attributed to the biased guidance from the teacher GNN during training. It is worth noting that in most cases, directly retraining the teacher GNN for debiasing is infeasible, since retraining the teacher GNN with a large number of parameters is computationally expensive. Hence, mitigating the bias for the student model is an urgent need.

Despite the necessity of mitigating bias for the student model, existing exploration remains scarce. In this paper, we aim to make an initial step towards developing a debiasing framework that can be easily adapted to various existing GNN-based KD methods. However, this task is non-trivial mainly due to the following three challenges: (1) **Gap towards Fair Knowledge:** For most KD frameworks designed for compressing GNNs, the teacher GNN model usually serves as the sole source of supervision signal for the training of the student model. Therefore, if the teacher GNN exhibits any bias, such biased knowledge tends to be inherited by the student model. Hence, learning a fair student model with biased supervision from the teacher



(A) Bias under Δ_{SP} on CPF.

(B) Bias under Δ_{EO} on AKD.

FIGURE 4.11. A comparison of exhibited bias between teacher and student models based on two representative GNN knowledge distillation frameworks (CPF and GraphAKD). "T" and "S" represent the teacher and the student model, respectively. The names of GNNs mark out the teacher models.

GNN is our first challenge. (2) **Gap towards End-to-End Learning:** A critical advantage of existing KD models is the end-to-end learning paradigm, which enables the distilled knowledge to be tailored to specific downstream tasks. In such an end-to-end learning process, highly efficient gradient-based optimization techniques are widely adopted. However, widely-used fairness notions (e.g., Statistical Parity and Equal Opportunity) are defined on the predicted labels. Hence the corresponding bias metrics are naturally non-differentiable w.r.t. the student model parameters. Developing a debiasing framework suitable for gradient-based optimization techniques in the end-to-end learning paradigm is our second challenge. (3) **Gap towards Generalization:** Various KD models have been proposed for compressing GNNs to satisfy different application scenarios. In fact, most KD models are developed based on certain designs of student models. Developing a framework that is student-agnostic and easily adapted to different KD models is our third challenge.

To tackle the above challenges, in this paper, we propose a novel framework named RELIANT (faiR knowlEdge distiLlatIon for grAph Neural neTworks) to mitigate the bias learned by the student model. Specifically, we first formulate a novel research problem of *Fair Knowledge Distillation for GNN-based Teacher-Student Frameworks*. To tackle the first challenge, we incorporate a learnable proxy of the exhibited bias for the student model. In this way, despite the knowledge (from the teacher GNN) being biased, the student model still makes less biased predictions under proper manipulations on the proxy. To tackle the second challenge, we propose to approximate the bias level of the student model, where the approximation is differentiable (w.r.t. the student model parameters) manner. In this way, the highly efficient end-to-end learning paradigm is preserved, and the gradient-based optimization techniques are still applicable. To tackle the third challenge, we design the proposed framework RELIANT in a student-agnostic manner. In other words, the debiasing for the student model does not rely on any specific design tailored for the student model structure. Therefore, RELIANT can be easily adapted to different GNN-based knowledge distillation approaches. The main contributions of this paper are summarized as follows.

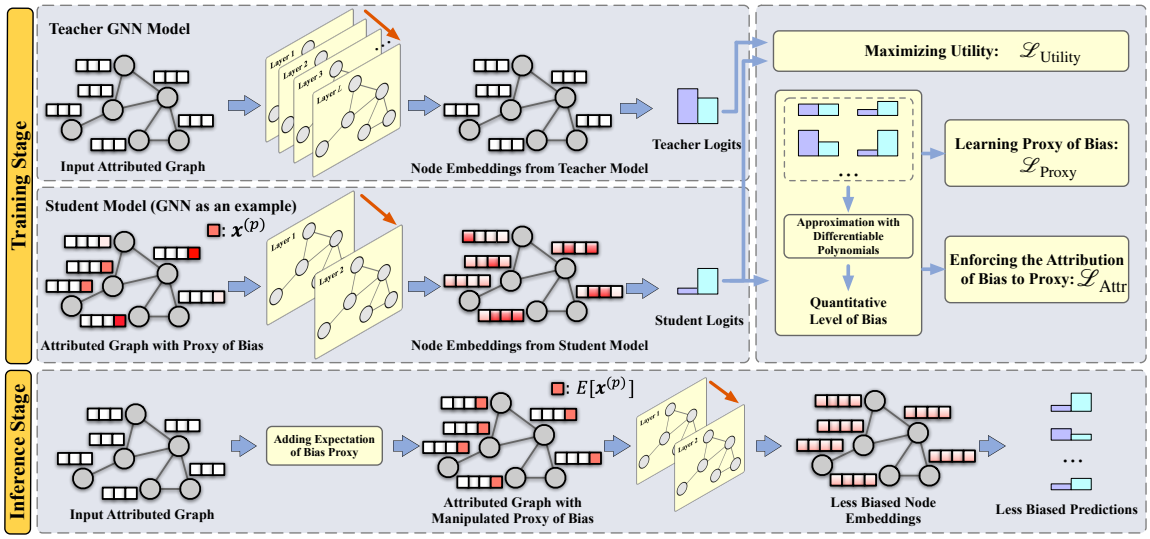


FIGURE 4.12. An overview of the proposed framework RELIANT including the training and inference stage.

- **Problem Formulation.** We formulate and make an initial investigation on a novel research problem of fair knowledge distillation for GNN-Based teacher-student frameworks.
- **Algorithmic Design.** We propose a principled framework named RELIANT that learns the proxy of bias for the student model during KD. RELIANT achieves student-agnostic debiasing via manipulating the proxy during inference.
- **Experimental Evaluation.** We conduct comprehensive experiments on multiple real-world datasets to verify the effectiveness of the proposed framework RELIANT in learning less biased student models.

4.3.2 Problem Definition

Notations. We denote matrices, vectors, and scalars by bold uppercase letters (e.g., $\mathbf{X}$), bold lowercase letters (e.g., $\mathbf{x}$), and regular lowercase letters (e.g., x), respectively. For any matrix, e.g., $\mathbf{X}$, we use $\mathbf{X}_{i,j}$ to indicate the element at the i-th row and j-th column.

Preliminaries. We utilize $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$ to denote an attributed network (graph). Here, $\mathcal{V} = \{v_1, ..., v_n\}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ ($\mathbf{x}_i \in \mathbb{R}^d$, $1 \leq i \leq n$) is the set of node attribute vectors. We use $\mathbf{A} \in \{0, 1\}^{n \times n}$ to denote the adjacency matrix of the graph. If there is an edge from the i-th node to the j-th node, $\mathbf{A}_{i,j} = 1$; otherwise $\mathbf{A}_{i,j} = 0$. Moreover, we denote the pre-trained teacher GNN model in a knowledge distillation framework as $f_{\hat{\boldsymbol{\theta}}}$ parameterized by $\hat{\boldsymbol{\theta}}$. Here $\hat{\boldsymbol{\theta}}$ denotes the optimized $\boldsymbol{\theta}$ of the pre-trained teacher model. Similarly, we denote the student model as $g_{\boldsymbol{\phi}}$ parameterized by $\boldsymbol{\phi}$. We represent the optimized $\boldsymbol{\phi}$ after the training of the student model as $\hat{\boldsymbol{\phi}}$. Without loss of generality, we consider the most widely studied node classification as the downstream task. For the teacher model $f_{\hat{\boldsymbol{\theta}}}(v)$, we denote the set of outcome logits, i.e., the continuous output vector corresponding to each node, as $\hat{\mathcal{Y}}^{(\text{t})} = \{\hat{\mathbf{y}}_1^{(\text{t})}, \hat{\mathbf{y}}_2^{(\text{t})}, ..., \hat{\mathbf{y}}_n^{(\text{t})}\}$, where $\hat{\mathbf{y}}_i^{(\text{t})} \in \mathbb{R}^c$. Here c is the total number of node classes. Correspondingly, we represent the set of outcome logits of the student model $g_{\boldsymbol{\phi}}(v)$ as $\hat{\mathcal{Y}}^{(\text{s})} = \{\hat{\mathbf{y}}_1^{(\text{s})}, \hat{\mathbf{y}}_2^{(\text{s})}, ..., \hat{\mathbf{y}}_n^{(\text{s})}\}$. For any node v_i, the predicted label

given by the student model (denoted as $\hat{Y}_i^{(s)}$ for the i-th node) is determined by the largest value across all c dimensions in $\hat{\mathbf{y}}_i^{(s)}$.

Based on the definitions above, we formulate the problem of *Fair Knowledge Distillation for GNN-based Teacher-Student Frameworks* as follows.

PROBLEM 4.3.1. ***Fair Knowledge Distillation for GNN-Based Teacher-Student Frameworks.*** *Given an attributed network $\mathcal{G}$ and a GNN-based teacher-student framework including a trained teacher GNN $f_{\hat{\boldsymbol{\theta}}}$ and a student model $g_{\boldsymbol{\phi}}$ to be trained, our goal is to achieve a more fair student model with similar prediction utility compared with $f_{\hat{\boldsymbol{\theta}}}$ in node classification.*

4.3.3 Methodology

In this section, we first present an overview of the proposed framework RELIANT, followed by the objective function formulation and optimization strategy.

4.3.3.1 Workflow of RELIANT

Here we first introduce the workflow of the proposed framework RELIANT. In general, we introduce the three main functionalities involved in the proposed framework RELIANT, namely *maximizing the utility*, *learning proxy of bias*, and *enforcing the attribution of bias to the proxy*. We present an overview of RELIANT in Fig. 4.12. Specifically, to tackle the first challenge (gap towards fair knowledge), we propose to first learn the proxy of bias as extra input attributes for the student model to account for the exhibited bias (on training nodes), and then exclude the information of proxy during test with manipulated pseudo proxy. To tackle the second challenge (gap towards end-to-end learning), we formulate our debiasing objectives in a differentiable (w.r.t. the parameters of the student model) manner. To tackle the third challenge (gap towards generalization), we achieve debiasing in a student-agnostic manner. In other words, the proposed framework RELIANT does not rely on any specific student model structure to achieve debiasing. We elaborate more details as follows.

Maximizing Utility. In general, existing GNN-based KD frameworks consider the GNNs with high computational costs as the teacher model, and the goal is to learn a student model with limited computational costs but similar prediction utility (e.g., accuracy in node classification tasks). To maintain the utility of the teacher model, it is necessary to utilize the knowledge from the teacher model as the supervision signal for the training of the student. In particular, a common approach is to utilize the output classification logits from the teacher model as the supervision signal, which we take as an example here. Specifically, we minimize the distance between the logits from the student model and the teacher model. We formally formulate the optimization goal as

$$\min_{\boldsymbol{\phi}} \sum_{v_i \in \mathcal{V}} \gamma_{\mathrm{d}} \left(\hat{\mathbf{y}}_i^{(t)}, \hat{\mathbf{y}}_i^{(s)} \right), \tag{4.22}$$

where $\hat{\mathbf{y}}_i^{(s)}$ and $\hat{\mathbf{y}}_i^{(t)}$ are the output logits from the student model $g_{\boldsymbol{\phi}}(v_i)$ and teacher model $f_{\hat{\boldsymbol{\theta}}}(v_i)$, respectively. The function $\gamma_{\mathrm{d}}(.,.)$ measures the distance between two logit vectors. Different choices can be adopted to measure the distance, e.g., cosine distance and Euclidean

distance. Correspondingly, to maximize the prediction utility, we minimize the objective

$$\mathscr{L}_{\text{Utility}}(\boldsymbol{\phi}) = \sum_{v_i \in \mathcal{V}} \gamma_{\text{d}} \left(\hat{\mathbf{y}}_i^{(t)}, \hat{\mathbf{y}}_i^{(s)} \right). \tag{4.23}$$

Learning Proxy of Bias. It is worth noting that even if the sensitive attributes are removed from the input data, the student model could still exhibit bias in its predictions. The main reason is that there could exist dependencies between those sensitive attributes and non-sensitive ones. Moreover, the information about sensitive attributes could also be encoded in the input network structure [47]. As a consequence, it is difficult to prevent the student model from leveraging information about sensitive attributes. To handle such a problem, we propose to learn the proxy of bias $\mathbf{x}_i^{(p)}$ as extra input attributes for each node v_i. Here, the rationale is that if much information about bias comes from the learned proxy instead of those encoded in the non-sensitive attributes or the network structures, then we are able to achieve less biased predictions by not using the information from such a proxy during test. As a consequence, such a proxy of bias should account for the exhibited bias of the student model as much as possible. In other words, the exhibited bias should be largely attributed to the proxy of bias rather than the sensitive information encoded in the network data. More specifically, to enforce the proxy of bias contributing to the exhibited bias in the student model, we propose to maximize the exhibited bias when these proxies are taken as input into the student model together with other attributes and the network structure. We formally formulate our goal as

$$\max_{\mathbf{X}^{(p)}} J_{\text{Bias}}(\{g_{\boldsymbol{\phi}}(\gamma(v_i, \mathbf{X}^{(p)})) : i \in \mathcal{V}\}), \tag{4.24}$$

where $\gamma(.,.)$ is a function that takes a node and the proxy of bias matrix as input, and outputs the node with a concatenated node attribute vector $[\mathbf{x}_i, \mathbf{x}_i^{(p)}]$. Here $\mathbf{x}_i^{(p)}$ is the i-th row of $\mathbf{X}^{(p)}$. $J_{\text{Bias}}(.)$ is a function that takes the set of logits from the student model as input and outputs a value indicating the level of exhibited bias. Nevertheless, the computation is non-differentiable under traditional fairness notions such as Statistical Parity and Equal Opportunity. Here we propose to utilize orthogonal polynomials (e.g., Legendre polynomials [42]) that are differentiable w.r.t. the output logits to approximate the level of bias under traditional fairness notions. This makes J_{Bias} differentiable w.r.t. the learnable parameter $\boldsymbol{\phi}$. Correspondingly, we formally give the objective function towards the goal above as

$$\mathscr{L}_{\text{Proxy}}(\mathbf{X}^{(p)}) = -J_{\text{Bias}}(\{g_{\boldsymbol{\phi}}(\gamma(v_i, \mathbf{X}^{(p)})) : i \in \mathcal{V}\}). \tag{4.25}$$

Enforcing the Attribution of Bias to the Proxy. Only achieving Eq. (4.24) is not enough to enforce the proxy of bias largely accounting for the exhibited bias of the student model. This is because the vanilla node attributes could still contribute to the exhibited bias. More specifically, we denote $P(\hat{Y}^{(s)})$ as the probability of a certain classification prediction given by the student model for any specific node. We assume that there are underlying unbiased and biased node attributes $\mathbf{X}^{(u)}$ and $\mathbf{X}^{(b)}$, respectively. When Eq. (4.24) is achieved, it is clear that $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)}, \mathbf{X}^{(b)}, \mathbf{X}^{(p)})$, i.e., $P(\hat{Y}^{(s)}|\mathbf{X}, \mathbf{X}^{(p)})$, is biased. However, both $\mathbf{X}^{(b)}$ and $\mathbf{X}^{(p)}$ could be the source of the exhibited bias. It is worth noting that our goal is to learn proxy $\mathbf{X}^{(p)}$ to account for as much of the exhibited bias as possible. Therefore, to enforce the effectiveness of the proxy, it is necessary to ensure that the exhibited bias is attributed to the biased information from $\mathbf{X}^{(p)}$ instead of $\mathbf{X}^{(b)}$. In other words, we need to enforce $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)}, \mathbf{X}^{(b)})$ being less biased, which ensures that $\mathbf{X}^{(p)}$ accounts for the exhibited bias

TABLE 4.11. The basic information about the real-world datasets adopted for experimental evaluation. Sens. denotes the semantic meaning of sensitive attribute.

Dataset	Recidivism	Credit Defaulter	DBLP	DBLP-L
# Nodes	18,876	30,000	39,424	129,726
# Edges	321,308	1,436,858	52,460	591,039
# Attributes	18	13	5,693	5,693
Avg. degree	34.0	95.8	1.3	4.6
Sens.	Race	Age	Continent of Affiliation	Continent of Affiliation
Label	Bail Decision	Future Default	Research Field	Research Field

as much as possible. Nevertheless, $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)}, \mathbf{X}^{(b)})$ is intractable considering that the number of the input dimension number for the student model is fixed. Hence we propose an alternative approach. Denote the learned proxy of bias and the underlying sensitive attribute vector of any node as $\mathbf{x}^{(p)}$ and $\mathbf{s}$, respectively. We propose to utilize a vector $\mathbb{E}[\mathbf{x}^{(p)}]$ to replace each row in $\mathbf{X}^{(p)}$ as the manipulated pseudo proxy $\tilde{\mathbf{X}}^{(p)}$. In this way, the rows in $\tilde{\mathbf{X}}^{(p)}$ are independent from $\mathbf{s}$, i.e., the information about sensitive attributes encoded in $\mathbf{X}^{(p)}$ is wiped out. To enforce the attribution of bias to the proxy $\mathbf{X}^{(p)}$, the predictions should be as fair as possible when the information about $\mathbf{X}^{(p)}$ is removed. Therefore, we formulate our last optimization goal as

$$\min_{\boldsymbol{\phi}} J_{\text{Bias}}(\{g_{\boldsymbol{\phi}}(\tilde{\gamma}(v_i, \tilde{\mathbf{X}}^{(p)})) : i \in \mathcal{V}\}), \tag{4.26}$$

where $\tilde{\gamma}(.,.)$ is a function that takes a node and the matrix $\tilde{\mathbf{X}}^{(p)}$ as input, and returns the input node with a concatenated node attribute vector $[\mathbf{x}_i, \tilde{\mathbf{x}}_i^{(p)}]$. Here $\tilde{\mathbf{x}}_i^{(p)}$ is the i-th row of matrix $\tilde{\mathbf{X}}^{(p)}$. We formally present the corresponding objective function as

$$\mathscr{L}_{\text{Attr}}(\boldsymbol{\phi}) = J_{\text{Bias}}(\{g_{\boldsymbol{\phi}}(\tilde{\gamma}(v_i, \tilde{\mathbf{X}}^{(p)})) : i \in \mathcal{V}\}). \tag{4.27}$$

Inference with Student Model. To achieve less biased inference, an ideal case is to make predictions with $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)})$. However, it is difficult to explicitly extract $\mathbf{X}^{(u)}$ from $\mathbf{X}$. Instead, we argue that $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)}, \mathbf{X}^{(b)}, \tilde{\mathbf{X}}^{(p)})$ exhibits similar level of bias compared with $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)})$. This is because (1) the bias exhibited by $P(\hat{Y}^{(s)}|\mathbf{X}^{(u)}, \mathbf{X}^{(b)}, \tilde{\mathbf{X}}^{(p)})$ minimally relies on $\mathbf{X}^{(b)}$ after enforcing Eq. (4.24) and Eq. (4.26); and (2) there is no further information about sensitive attributes encoded in $\tilde{\mathbf{X}}^{(p)}$ (as discussed above). Consequently, we propose to utilize $g_{\boldsymbol{\phi}}(\tilde{\gamma}(v_i, \tilde{\mathbf{X}}^{(p)}))$ to achieve less biased prediction for node v_i in the inference stage.

4.3.4 Optimization Objectives & Strategy

We present the optimization objectives of RELIANT followed by the training strategy in this section.

Optimization Objectives. Based on our discussions above, here we present a summary of the optimization objectives for the proposed RELIANT. First, to optimize the parameter $\boldsymbol{\phi}$, we formally formulate a unified objective function as

$$\mathscr{L}_{\boldsymbol{\phi}} = \mathscr{L}_{\text{Utility}}(\boldsymbol{\phi}) + \lambda \cdot \mathscr{L}_{\text{Attr}}(\boldsymbol{\phi}). \tag{4.28}$$

Here λ serves as a hyper-parameter controlling the effect of debiasing the student model. Second, to optimize the learnable proxy of bias $\mathbf{X}^{(p)}$, we formally present the objective function as

$$\mathscr{L}_{\mathbf{X}^{(\mathrm{P})}} = \mathscr{L}_{\text{Proxy}}(\mathbf{X}^{(p)}). \tag{4.29}$$

Optimization Strategy. To train the proposed framework RELIANT, we propose to optimize the parameter ϕ and learnable proxy of bias $\mathbf{X}^{(p)}$ in an alternating manner. We present the algorithmic routine of RELIANT in Algorithm 2.

Algorithm 2 Fair Knowledge Distillation for GNNs

Input: $\mathcal{G}$: the graph data; $f_{\hat{\boldsymbol{\theta}}}$: the trained teacher GNN model; $g_{\boldsymbol{\phi}}$: the student model;
Output: $g_{\hat{\boldsymbol{\phi}}}$: the optimized student model; $\mathbf{X}^{(p)}$: the proxy of bias matrix;
1: Randomly initialize $\mathbf{X}^{(p)}$;
2: **while** stop training condition not satisfied **do**
3: Compute $\mathscr{L}_{\boldsymbol{\phi}}$ according to Eq. (4.28);
4: Update $\boldsymbol{\phi}$ with $\frac{\partial \mathscr{L}_{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}$;
5: Compute $\mathscr{L}_{\mathbf{X}^{(p)}}$ according to Eq. (4.29);
6: Update $\mathbf{X}^{(p)}$ with $\frac{\partial \mathscr{L}_{\mathbf{X}^{(p)}}}{\partial \mathbf{X}^{(p)}}$;
7: **end while**
8: **Return:** $g_{\hat{\boldsymbol{\phi}}}$ and $\mathbf{X}^{(p)}$;

4.3.5 Experimental Evaluations

In this section, we will first introduce the downstream learning task and adopted real-world datasets, followed by the backbone models, baseline methods, and evaluation metrics. Next, we present the implementation details of the models. Finally, we discuss the evaluation results of the proposed RELIANT. In particular, we aim to answer the following research questions through experiments: **RQ1:** How well can RELIANT balance the utility and fairness of the student model compared with other baselines? **RQ2:** To what extent each component of RELIANT contributes to the overall debiasing performance? **RQ3:** How will the choice of the hyper-parameter λ affect the performance of RELIANT?

4.3.5.1 Experimental Settings

Here we introduce the settings for our experimental evaluation.

Downstream Task & Real-world Datasets. We adopt the widely studied node classification as the downstream task in this paper. We adopt four real-world datasets for the experimental evaluations, including two widely used network datasets (Recidivism [99, 3] and Credit Defaulter [227, 3]) and two newly constructed ones based on real-world data (DBLP and DBLP-L). In Recidivism, nodes are defendants released on bail, and edges denote the connections between defendants computed from their past criminal records. Here the sensitive feature is race, and we aim to classify if a certain defendant is unlikely to commit a crime after bail. In Credit Defaulter, nodes are credit card users, and edges are the connections between these users. Here we consider the age period of these users as their sensitive feature, and we aim to predict the future default of credit card payments. Additionally, we also construct two co-author networks, namely DBLP and DBLP-L based on AMiner network [183], which is a co-author network collected from computer science bibliography. Specifically, we first filter

TABLE 4.12. The experimental results based on node classification accuracy and Δ_{SP}. We use "(T)" and "(S)" suffixes to represent the teacher model and the student model, respectively. Here Vanilla(S) denotes the student model trained with the vanilla KD framework; One-Hot(S) represents the student model trained with the one-hot bias proxy; RELIANT(S) is the student model trained with our proposed model. ↑ denotes the larger, the better; while ↓ denotes the opposite. All quantitative results are presented in percentages. The best results are in **Bold**.

			DBLP	DBLP-L	Credit	Recidivism
CPF +GCN	**Accuracy (↑)**	**GCN(T)**	92.37 ± 0.06	94.20 ± 0.09	76.39 ± 0.48	**93.68 ± 0.21**
		Vanilla(S)	**93.14 ± 0.10**	**94.30 ± 0.04**	**77.85 ± 0.10**	89.41 ± 0.12
		One-Hot(S)	93.04 ± 0.34	94.16 ± 0.02	77.65 ± 0.10	89.15 ± 0.37
		RELIANT(S)	92.70 ± 0.40	94.07 ± 0.18	77.82 ± 0.45	88.88 ± 0.57
	Δ_{SP} (↓)	**GCN(T)**	7.66 ± 0.26	7.33 ±0.44	15.81 ±0.40	6.10 ±0.05
		Vanilla(S)	8.55 ± 0.50	7.16 ± 0.16	14.90 ± 0.89	6.85 ± 0.05
		One-Hot(S)	7.97 ± 0.63	7.46 ± 0.24	13.80 ± 0.32	6.78 ± 0.51
		RELIANT(S)	**2.27 ± 1.00**	**3.09 ± 0.36**	**10.28 ± 1.86**	**4.06 ± 0.64**
CPF +SAGE	**Accuracy (↑)**	**SAGE(T)**	92.57 ± 0.28	94.10 ± 0.25	77.88 ± 0.06	**89.71 ± 0.14**
		Vanilla(S)	**93.25 ± 0.15**	**94.97 ± 0.10**	77.97 ± 0.26	89.20 ± 0.11
		One-Hot(S)	93.07 ± 0.10	94.32 ± 0.07	78.01 ± 0.23	89.11 ± 0.29
		RELIANT(S)	92.91 ± 0.51	94.17 ± 0.93	**78.28 ± 0.36**	88.85 ± 0.27
	Δ_{SP} (↓)	**SAGE(T)**	8.32 ±0.24	7.81 ±0.08	14.08 ± 1.37	6.50 ±0.39
		Vanilla(S)	8.29 ± 0.85	7.02 ± 0.13	13.44 ± 5.23	4.41 ± 0.43
		One-Hot(S)	8.01 ± 0.25	7.52 ± 0.32	16.86 ± 3.86	6.62 ± 0.38
		RELIANT(S)	**2.01 ± 1.21**	**2.97 ± 0.61**	**10.06 ± 1.70**	**3.94 ± 0.60**
AKD +GCN	**Accuracy (↑)**	**GCN(T)**	**92.37 ± 0.06**	**94.20 ± 0.09**	**76.39 ± 0.48**	**93.68 ± 0.21**
		Vanilla(S)	92.06 ± 0.16	94.07 ± 0.11	76.35 ± 0.31	92.08 ± 0.29
		One-Hot(S)	91.55 ± 0.40	94.07 ± 0.04	75.65 ± 0.75	92.07 ± 0.03
		RELIANT(S)	91.39 ± 0.24	93.98 ± 0.08	75.64 ± 0.06	91.21 ± 0.14
	Δ_{SP} (↓)	**GCN(T)**	7.66 ±0.26	7.33 ±0.44	15.81±0.40	6.10 ±0.05
		Vanilla(S)	7.87 ± 0.25	6.79 ± 0.10	13.61 ± 2.00	6.54 ± 0.17
		One-Hot(S)	7.39 ± 0.35	6.72 ± 0.19	14.30 ± 0.24	6.44 ± 0.32
		RELIANT(S)	**3.66 ± 1.09**	**5.18 ± 0.16**	**8.47 ± 1.92**	**5.70 ± 0.18**
AKD +SAGE	**Accuracy (↑)**	**SAGE(T)**	**92.57 ± 0.28**	94.10 ± 0.25	77.88 ± 0.06	**89.71 ± 0.14**
		Vanilla(S)	92.23 ± 0.07	94.45 ± 0.03	78.10 ± 0.24	89.67 ± 0.07
		One-Hot(S)	92.31 ± 0.06	**94.52 ± 0.11**	78.24 ± 0.45	89.60 ± 0.12
		RELIANT(S)	92.07 ± 0.07	94.28 ± 0.06	**78.60 ± 0.33**	88.87 ± 0.31
	Δ_{SP} (↓)	**SAGE(T)**	8.32 ±0.24	7.81 ±0.08	14.08 ± 1.37	6.50 ±0.39
		Vanilla(S)	7.53 ± 0.29	7.34 ± 0.41	14.41 ± 0.15	6.24 ± 0.20
		One-Hot(S)	7.72 ± 0.44	7.26 ± 0.36	11.69 ± 0.93	6.18 ± 0.30
		RELIANT(S)	**4.91 ± 0.64**	**4.05 ± 0.14**	**5.00 ± 1.63**	**6.06 ± 0.26**

out the nodes in AMiner network with incomplete information. Then we adopt two different approaches to sample a connected network from the filtered dataset: DBLP is a subgraph sampled with random walk, while DBLP-L is the largest connected component of the filtered AMiner network. In both datasets, nodes represent the researchers in different fields, and edges denote the co-authorship between researchers. The sensitive attribute is the continent of the affiliation each researcher belongs to, and we aim to predict the primary research field of each researcher. The detailed statistics of these four datasets are in Table 4.11.

KD Framework Backbones & Teacher GNNs. To evaluate the capability of RELIANT in generalizing to different GNN-based KD backbones, here we adopt two representative KD frameworks designed for compressing GNNs, namely CPF [225] and GraphAKD [78]. In general, CPF minimizes the distribution distance between the logits from teacher and student to provide supervision information for the student, while GraphAKD utilizes adversarial training to achieve knowledge distillation for the student. The student model of CPF and GraphAKD is PLP [225] and SGC [205], respectively. For each KD framework, we adopt two types of GNNs (including GCN [111] and GraphSAGE [74]) as the teacher GNN.

Baselines. To the best of our knowledge, this is the first study on how to mitigate the bias exhibited in GNN-based KD frameworks. In experiments, we adopt the student model yielded by the vanilla KD framework as our first baseline. For our second baseline, we replace the learnable proxy of bias with a naive proxy for the input of the KD framework. Specifically, we utilize one-hot vectors as the naive proxy for different demographic subgroups during training, where the one-hot vector flags the membership of different nodes. We replace all proxy vectors during inference with an averaged proxy vector across all instances. Here, the rationale is that more distinguishable attributes are easier for deep learning models to learn during training, and these one-hot vectors serve as an "easier" indicator of biased information. In this way, if these one-hot proxy accounts for the exhibited bias of the student model after training, then the exhibited bias could also be mitigated during inference, where such information is wiped out.

Evaluation Metrics. We evaluate the performance of the compressed GNN models (i.e., the output student model of KD frameworks) from two perspectives, namely utility and fairness. Specifically, in terms of utility, we adopt the node classification accuracy as the corresponding metric; in terms of fairness, we adopt two traditional metrics Δ_{SP} and Δ_{EO}. Here Δ_{SP} measures the bias level (of predictions) under the fairness notion of Statistical Parity, while Δ_{EO} measures the bias level under the notion of Equal Opportunity. See online version of this paper for other results in the online version[10].

Implementation Details. RELIANT is implemented in PyTorch [154] and optimized with Adam optimizer [109]. In our experiments, the learning rate is chosen in $\{10^{-2}, 10^{-3}, 10^{-4}\}$ and the training epoch number is set as 1,000 for CPF and 600 for GraphAKD. Experiments are carried out on an Nvidia RTX A6000, and the reported numerical results are averaged across three different runs. We introduce more details in the online version.

4.3.5.2 Effectiveness of RELIANT

Here we aim to answer **RQ1**. Specifically, we evaluate our proposed framework RELIANT on two KD backbones, namely CPF and GraphAKD. For each KD backbone, we adopt two different GNNs (GCN and GraphSAGE) to evaluate the capability of our proposed framework in generalizing to different GNNs. We compare the corresponding performances between the teacher GNN model and the student models trained with three different frameworks (i.e., the vanilla KD framework, the KD framework with the one-hot proxy of bias, and our proposed RELIANT). We present quantitative results on node classification accuracy and Δ_{SP} in Table 4.12. In addition, we also perform experiments based on Equal Opportunity (see the

[10]See online version here https://epubs.siam.org/doi/abs/10.1137/1.9781611977653.ch18 for supplementary discussion and experimental results.

online version), where we have consistent observations. We make the following observations from Table 4.12.

- From the perspective of prediction utility, student models trained with all three KD frameworks achieve comparable performances with the teacher model. This implies that effective knowledge distillation can be achieved by all three KD frameworks.
- From the perspective of bias mitigation, the student models trained with the vanilla KD frameworks inherit and even exaggerate the exhibited bias from the teacher GNN model in all cases. Training the student models with the one-hot proxy can mitigate bias in most cases. Compared with the student models trained with the vanilla KD framework and the one-hot proxy, RELIANT consistently exhibits less bias w.r.t. Statistical Parity.
- Based on the performance of RELIANT in both perspectives, RELIANT achieves effective debiasing for the student model but still maintains comparable model utility with the teacher model. Therefore, we argue that RELIANT achieves a satisfying balance between debiasing and maintaining utility.

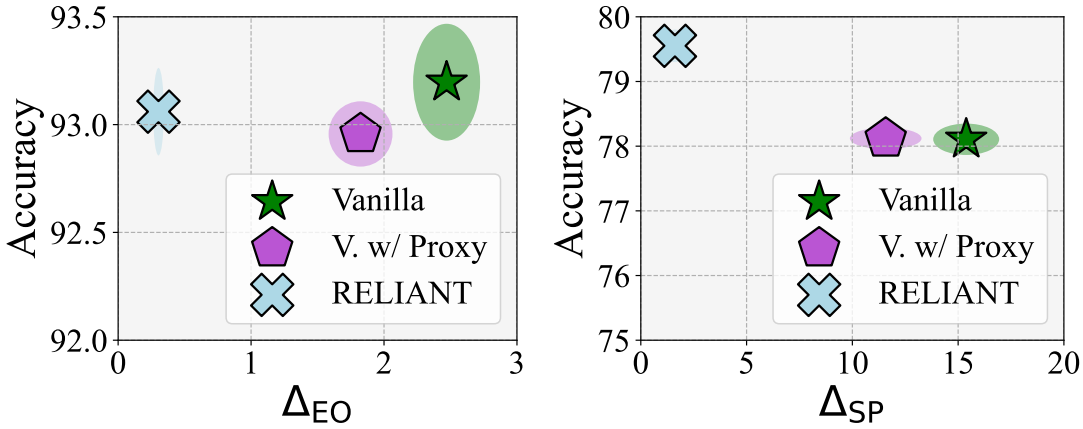
4.3.5.3 Ablation Study

We aim to answer **RQ2** in this subsection. Specifically, for each framework, we evaluate to what extent the two modules of RELIANT (including *learning proxy of bias* and *enforcing the attribution of bias to the proxy*) contribute to the performance of the student model. We present the results in Fig. 4.13. Here, Fig. 4.13a is the performance of accuracy vs. Δ_{SP} from CPF based on the DBLP-L dataset, while Fig. 4.13b is the performance of accuracy vs. Δ_{EO} from GraphAKD based on the Recidivism dataset. Notably, we also have similar observations under other settings. We make the following observations.

- From the perspective of prediction utility, we observe that the prediction utility is comparable among all three cases. This corroborates that both modules exert limited influence on the node classification accuracy.
- From the perspective of bias mitigation, adding the module of *learning proxy of bias* to the vanilla KD framework brings limited bias mitigation. This is because the bias could also come from the non-sensitive node attributes (as discussed in Section 4.3.3.1). After the module of *enforcing the attribution of bias to the proxy* is added together with *learning proxy of bias*, RELIANT is then able to achieve satisfying performance on bias mitigation.

4.3.5.4 Parameter Sensitivity

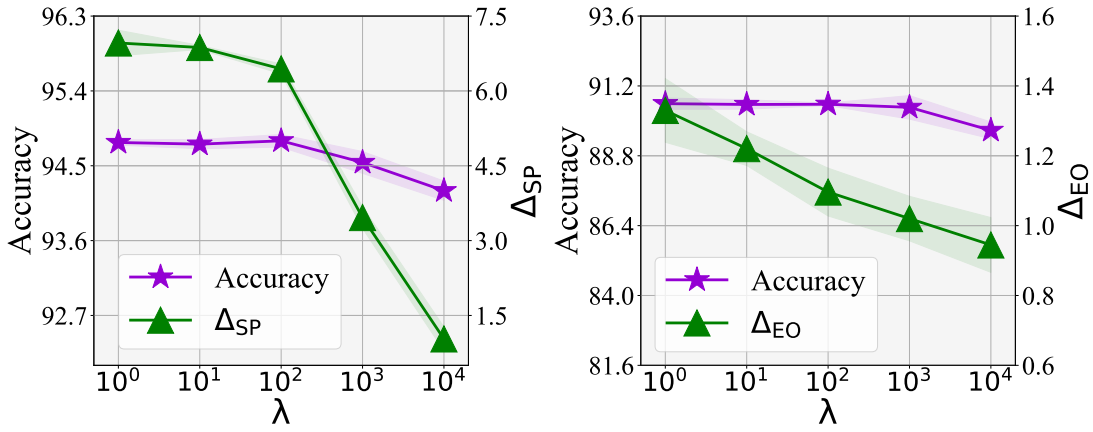
We answer **RQ3** by studying the tendency of model utility and exhibited bias w.r.t. the change of hyper-parameter λ. Here λ controls the effect of $\mathscr{L}_{\text{Attr}}$. More specifically, we vary λ in $\{10^0, 10^1, 10^2, 10^3, 10^4\}$, and we present the corresponding tendency of node classification accuracy and the exhibited bias of the trained student model with RELIANT in Fig. 4.14. Here, Fig. 4.14a is based on the Credit dataset under GraphAKD, while Fig. 4.14b is based on the DBLP dataset under CPF. We also have similar observations on other datasets. We make the following observations from Fig. 4.14.



(A) GraphAKD on Credit.

(B) CPF on DBLP.

FIGURE 4.13. Ablation study of RELIANT. "Vanilla" denotes the student model trained with the original KD framework, while "V. w/ Proxy" represents the student model trained under the KD framework with only learning the proxy of bias.

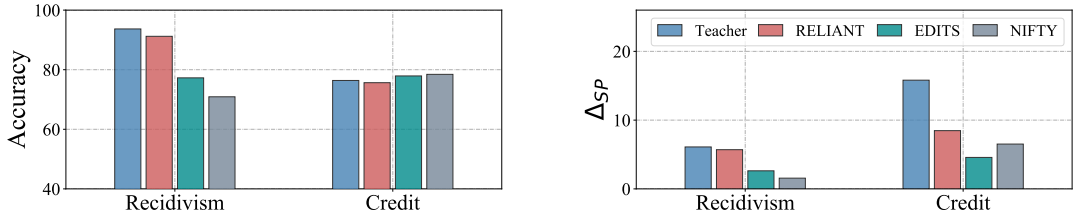


(A) GraphAKD on Credit.

(B) CPF on DBLP.

FIGURE 4.14. Parameter sensitivity of λ based on two different KD backbones on two real-world datasets. We also have similar observations on other datasets.

- From the perspective of prediction utility, the node classification accuracies on both datasets and KD backbones do not exhibit apparent reduction when the value of λ increases from 1 to 10^4. This verifies that the prediction utility is not sensitive to λ.
- From the perspective of bias mitigation, the student model exhibits less bias when λ increases from 1 to 10^4. Specifically, when λ is relatively small (e.g., 1), the learned proxy of bias only partially accounts for the exhibited bias; when the value of λ increases, more bias is then attributed to the learned proxy. Considering the balance between model utility and bias mitigation, a recommended range of λ is between 10^2 and 10^3.



(A) Comparison on the node classification accuracy.

(B) Comparison on Δ_{SP}.

FIGURE 4.15. Performance comparison on balancing prediction utility and bias mitigation between the proposed framework RELIANT and state-of-the-art methods that directly debias GNNs. All values are in percentage.

4.3.5.5 Effectiveness of RELIANT

Here we present complementary experiments to answer **RQ1**, where the fairness notion is instantiated with Equal Opportunity (measured with Δ_{EO}). Here we adopt the same settings as those in Section 4.3.5.2. We compare the performances between the teacher GNN model and the student models trained with three different framework variants, including the vanilla KD framework, the KD framework with the one-hot proxy of bias, and our proposed RELIANT. We present quantitative results on node classification accuracy and Δ_{EO} in Table 4.13. We make the following observations from Table 4.13.

- From the perspective of prediction utility, all student models trained with the adopted three KD frameworks are able to achieve comparable performances with the teacher model. This corroborates that all three KD frameworks are capable of achieving effective knowledge distillation.
- From the perspective of bias mitigation, the student models trained with the vanilla KD frameworks inherit or exaggerate the bias from the teacher GNN in all cases. Compared with the student models trained with the vanilla KD framework and the one-hot proxy, RELIANT consistently exhibits less bias under the fairness notion of Equal Opportunity. The student model trained with RELIANT even exhibits less bias than the teacher model in certain cases, which further corroborates the effectiveness of RELIANT in training less biased student models.
- According to the performance of RELIANT in both perspectives above, RELIANT is proved to achieve effective debiasing for the student model but maintains comparable prediction utility compared with the teacher model. Therefore, we argue that RELIANT achieves a satisfying balance between debiasing and maintaining the prediction utility.

4.3.5.6 RELIANT vs. GNN-Debiasing Methods

Here we perform experiments and compare the performance of RELIANT with other state-of-the-art GNN debiasing methods on balancing the prediction utility and bias mitigation.

Baselines. Here we choose two state-of-the-art GNN debiasing methods as our baselines, namely EDITS [47] and NIFTY [3]. EDITS is a recent GNN debiasing method that learns less

biased network data in the pre-processing stage. After debiasing, the network data will be fed into the GNN model for evaluation. NIFTY is another recent GNN-based debiasing framework that achieves bias mitigation in the processing stage. During training, node representations are learned to be invariant to the sensitive attributes after counterfactual perturbations.

Backbones. Here we choose the most widely used GCN as the backbone GNN model for all methods. We choose GraphAKD as the KD backbone of the proposed RELIANT. It is also worth noting that we also have similar observations with other GNN backbones.

Discussion. We present the performance comparison results on node classification accuracy and Δ_{SP} in Fig. 4.15. We make the following observations.

- From the perspective of prediction utility, RELIANT keeps comparable to the teacher GCN model, while other debiasing methods bear different levels of prediction utility corruption. Therefore, RELIANT achieves satisfying performance in maintaining the prediction utility among all methods.
- From the perspective of bias mitigation, RELIANT is able to achieve comparable debiasing performance with other baselines when all models bear similar prediction utility (e.g., on Credit dataset); when baselines outperform RELIANT on bias mitigation, there is also much more prediction utility sacrifice (e.g., on Recidivism dataset). Considering that debiasing the student model with biased supervision is much more difficult than directly debiasing GNNs, we argue that the performances of RELIANT in both cases should be considered satisfying.
- According to the performance of RELIANT in both perspectives above, we argue that RELIANT achieves comparable performance with other state-of-the-art GNN debiasing approaches, which further corroborates its satisfying performance on balancing the prediction utility and bias mitigation.

4.3.6 Related Works

Algorithmic Fairness in GNNs. Most existing works promoting the algorithmic fairness of GNNs focus either on *Group Fairness* [58] or *Individual Fairness* [233]. Specifically, group fairness is defined based on a set of pre-defined sensitive attributes (e.g., gender and race). These sensitive attributes divide the whole population into different demographic subgroups. Group fairness requires that each subgroup should receive their fair share of interest according to the output GNN predictions [134]. Various explorations have been made towards achieving a higher level of group fairness for GNNs [50]. Decoupling the output predictions from sensitive attributes via adversarial learning is one of the most popular approaches among existing works [201, 37]. Other common strategies include reformulating the objective function with fairness regularization [63, 144], rebalancing the number of intra-group edges between two demographic subgroups [47, 125], deleting nodes or edges that contribute the most to the exhibited bias [51, 48], etc. On the other hand, individual fairness does not rely on any sensitive attributes. Instead, individual fairness requires that similar nodes (in the input space) should be treated similarly (in the output space) [58]. To fulfill individual fairness in GNNs, adding fairness-aware regularization terms to the optimization objective is the most widely adopted approach [46, 177].

TABLE 4.13. The experimental results based on node classification accuracy and Δ_{EO}. We use "(T)" and "(S)" suffixes to represent the teacher model and the student model, respectively. Here Vanilla(S) denotes the student model trained with the vanilla KD framework; One-Hot(S) represents the student model trained with the one-hot bias proxy; RELIANT(S) is the student model trained with our proposed model. ↑ denotes the larger, the better; while ↓ denotes the opposite. All quantitative results are presented in percentages. The best results are in **Bold**.

			DBLP	DBLP-L	Credit	Recidivism
CPF +GCN	**Accuracy (↑)**	**GCN(T)**	92.37 ± 0.06	**94.20 ± 0.09**	76.39 ± 0.48	**93.68 ± 0.21**
		Vanilla(S)	**93.24 ± 0.18**	94.15 ± 0.04	77.58 ± 0.20	89.36 ± 0.16
		One-Hot(S))	93.07 ± 0.35	94.15 ± 0.07	**77.65 ± 0.10**	89.38 ± 0.15
		RELIANT(S)	93.20 ± 0.12	94.15 ± 0.08	77.00 ± 1.57	89.30 ± 0.19
	Δ_{EO} **(↓)**	**GCN(T)**	2.31 ± 0.13	2.29 ± 0.34	12.63 ± 0.24	0.52 ± 0.06
		Vanilla(S)	2.56 ± 0.11	2.34 ± 0.29	11.56 ± 0.38	1.25 ± 0.19
		One-Hot(S)	2.21 ± 0.32	2.17 ± 0.26	10.69 ± 0.31	0.96 ± 0.28
		RELIANT(S)	**0.42 ± 0.21**	**1.08 ± 0.10**	**6.02 ± 4.78**	**0.35 ± 0.12**
CPF +SAGE	**Accuracy (↑)**	**SAGE(T)**	92.57 ± 0.28	94.10 ± 0.25	77.88 ± 0.06	**89.71 ± 0.14**
		Vanilla(S)	93.22 ± 0.03	**94.37 ± 0.08**	**78.30 ± 0.23**	89.15 ± 0.27
		One-Hot(S)	93.13 ± 0.11	94.36 ± 0.06	78.01 ± 0.23	88.98 ± 0.55
		RELIANT(S)	**93.24 ± 0.09**	94.32 ± 0.06	78.11 ± 0.40	89.01 ± 0.26
	Δ_{EO} **(↓)**	**SAGE(T)**	2.51 ± 0.33	2.67 ± 0.19	11.05 ± 0.71	0.86 ± 0.03
		Vanilla(S)	2.83 ± 0.34	2.00 ± 0.18	11.07 ± 4.61	1.17 ± 0.11
		One-Hot(S)	2.16 ± 0.27	2.05 ± 0.21	12.73 ± 2.29	1.23 ± 0.08
		RELIANT(S)	**0.63 ± 0.42**	**0.86 ± 0.18**	**6.72 ± 4.49**	**0.51 ± 0.25**
AKD +GCN	**Accuracy (↑)**	**GCN(T)**	**92.37 ± 0.06**	**94.20 ± 0.09**	76.39 ± 0.48	**93.68 ± 0.21**
		Vanilla(S)	92.12 ± 0.09	94.06 ± 0.06	**78.12 ± 0.65**	92.29 ± 0.06
		One-Hot(S)	91.68 ± 0.28	93.98 ± 0.13	77.87 ± 0.48	92.28 ± 0.13
		RELIANT(S)	91.69 ± 0.19	94.09 ± 0.12	77.88 ± 0.82	92.46 ± 0.09
	Δ_{EO} **(↓)**	**GCN(T)**	2.31 ± 0.13	2.29 ± 0.34	12.63 ± 0.24	**0.52 ± 0.06**
		Vanilla(S)	2.76 ± 0.33	1.88 ± 0.08	8.26 ± 3.41	0.82 ± 0.17
		One-Hot(S)	2.69 ± 0.28	1.87 ± 0.17	8.43 ± 5.08	0.97 ± 0.45
		RELIANT(S)	**1.79 ± 0.31**	**1.43 ± 0.09**	**4.96 ± 3.77**	0.66 ± 0.21
AKD +SAGE	**Accuracy (↑)**	**SAGE(T)**	**92.57 ± 0.28**	94.10 ± 0.25	77.88 ± 0.06	89.71 ± 0.14
		Vanilla(S)	92.23 ± 0.07	94.45 ± 0.03	78.10 ± 0.24	90.56 ± 0.14
		One-Hot(S)	92.31 ± 0.06	**94.52 ± 0.11**	78.24 ± 0.45	**90.85 ± 0.20**
		RELIANT(S)	92.15 ± 0.16	94.42 ± 0.05	**79.08 ± 0.29**	90.00 ± 0.64
	Δ_{EO} **(↓)**	**SAGE(T)**	2.51 ± 0.33	2.67 ± 0.19	11.05 ± 0.71	**0.86 ± 0.03**
		Vanilla(S)	2.06 ± 0.06	2.23 ± 0.23	10.56 ± 0.43	1.61 ± 0.39
		One-Hot(S)	2.21 ± 0.39	2.11 ± 0.21	8.38 ± 0.73	1.10 ± 0.37
		RELIANT(S)	**1.60 ± 0.45**	**1.89 ± 0.21**	**2.33 ± 0.80**	0.91 ± 0.22

Knowledge Distillation. In recent years, knowledge distillation has been proven to be effective in compressing the model but still maintaining similar model prediction performance [69]. Correspondingly, it has been widely adopted in a plethora of applications, including visual recognition [212], natural language processing [73, 91], etc. The main idea of knowledge distillation is to transfer the knowledge of a computationally expensive teacher model to a light student model, and thus the student model is able to fit in platforms with limited computing resources [78, 100]. It is worth noting that such a strategy is also proved to be

effective in compressing GNNs [225, 78, 100]. Consequently, there is growing research attention on utilizing knowledge distillation to compress GNNs for more efficient inference. For example, encouraging the student model to yield similar output to the teacher GNN via regularization is proved to be effective [78]. In addition, adversarial learning is also a popular technique to obtain light-weighted but accurate student models [78]. However, most of these frameworks for GNNs do not have fairness consideration. Hence the student model tends to be influenced by biased knowledge from the teacher GNN. Different from existing works, we develop a generalizable knowledge distillation framework that explicitly considers fairness in GNNs but still maintains the utility of GNN predictions.

4.3.7 Conclusion

Despite the success of Knowledge Distillation (KD) in compressing GNNs, most existing works do not consider fairness. Hence the student model trained with the KD framework tends to inherit and even exaggerate the bias from the teacher GNN. In this paper, we take initial steps towards learning less biased student models for GNN-based KD frameworks. Specifically, we first formulate a novel problem of fair knowledge distillation for GNN-based teacher-student frameworks, then propose a framework named RELIANT to achieve a less biased student model. Notably, the design of RELIANT is agnostic to the specific structures of teacher and student models. Therefore, it can be easily adapted to different KD approaches for debiasing. Extensive experiments demonstrate the effectiveness of RELIANT in fulfilling fairness for GNN compression with KD.

CHAPTER 5

Fairness Certification for Graph Machine Learning

5.1 Certified Defense on the Fairness of Graph Neural Networks

5.1.1 Introduction

Graph Neural Networks (GNNs) have emerged to be one of the most popular models to handle learning tasks on graphs [111, 188] and made remarkable achievements in various domains [66, 124]. Nevertheless, as GNNs are increasingly deployed in real-world decision-making scenarios, there has been an increasing societal concern on the fairness of GNN predictions. A primary reason is that most traditional GNNs do not consider fairness, and thus could exhibit bias against certain demographic subgroups. Here the demographic subgroups are usually divided by certain sensitive attributes, such as gender and race. To prevent GNNs from biased predictions, multiple recent studies have proposed fairness-aware GNNs [3, 37, 105] such that potential bias could be mitigated.

Unfortunately, despite existing efforts towards fair GNNs, it remains difficult to prevent the corruption of their fairness level due to their common vulnerability of lacking adversarial robustness. In fact, malicious attackers can easily corrupt the fairness level of GNNs by perturbing the node attributes (i.e., changing the values of node attributes) and/or the graph structure (i.e., adding and deleting edges) [83], which could lead to serious consequences in the test phase [37, 83]. For example, GNNs are leveraged to determine the salary of employees over a network based on their relational information. Yet, by simply injecting adversarial links in such a network of employees, attackers can make GNNs deliver advantaged salary predictions for a subgroup (e.g., employees with a certain nationality) while damaging the interest of others [83]. Hence achieving defense over the fairness of GNNs is crucial for safe deployment.

It is worth noting that despite the abundant empirical defense strategies for GNNs [238, 61, 93, 96, 207], they are always subsequently defeated by novel attacking techniques [171, 21], and the defense over the fairness of GNNs also faces the same problem. Therefore, an ideal way is to achieve certifiable defense on fairness (i.e., certified fairness defense). A few recent works aim to certify the fairness for traditional deep learning models [108, 106, 95, 140, 12, 165]. Nevertheless, most of them require specially designed training strategies [108, 95, 165] and thus cannot be directly applied to optimized GNNs ready to be deployed. More importantly, they mostly rely on assumptions on the optimization results [108, 95, 12, 165] or data distributions [106, 140] over a continuous input space. Hence they can hardly be

generalized to GNNs due to the binary nature of the input graph topology. Several other works propose certifiable GNN defense approaches to achieve theoretical guarantee [192, 10, 11, 94, 247, 248]. However, they mainly focus on securing the GNN prediction for a certain individual node to ensure model utility, ignoring the fairness defense over the entire population. Despite the significance, the corresponding study still remains in its infancy.

It is worth noting that achieving certifiable defense on the fairness of GNNs is a daunting task due to the following key challenges: (1) **Generality:** different types of GNNs could be designed and optimized for different real-world applications [245]. Correspondingly, our first challenge is to design a plug-and-play framework that can achieve certified defense on fairness for any optimized GNN models that are ready to be deployed. (2) **Vulnerability:** a plethora of existing studies have empirically verified that most GNNs are sensitive to input data perturbations [238, 249, 220]. In other words, small input perturbations may cause significant changes in the GNN output. Hence our second challenge is to properly mitigate the common vulnerabilities of GNNs without changing its structure or re-training. (3) **Multi-Modality:** the input data of GNNs naturally bears multiple modalities. For example, there are node attributes and graph topology in the widely studied attributed networks. In practice, both data modalities may be perturbed by malicious attackers. Therefore, our third challenge is to achieve certified defenses of fairness on both data modalities for GNNs.

As an early attempt to address the aforementioned challenges, in this paper, we propose a principled framework named ELEGANT (c**E**tifiab**LE** **G**NNs over the f**A**ir**N**ess of Predic**T**ions). Specifically, we focus on the widely studied task of node classification and formulate a novel research problem of *Certifying GNN Classifiers on Fairness*. To handle the first challenge, we propose to develop ELEGANT on top of an optimized GNN model without any assumptions over its structure or parameters. Hence ELEGANT is able to serve as a plug-and-play framework for any optimized GNN model ready to be deployed. To handle the second challenge, we propose to leverage randomized smoothing [192, 33] to defend against malicious attacks, where most GNNs can then be more robust over the prediction fairness level. To handle the third challenge, we propose two different strategies working in a concurrent manner, such that certified defense against the attacks on both the node attributes (i.e., add and subtract attribute values) and graph topology (i.e., flip the existence of edges) can be realized. Finally, we evaluate the effectiveness of ELEGANT on multiple real-world network datasets. In summary, our contributions are three-fold: (1) **Problem Formulation.** We formulate and make an initial investigation on a novel research problem of *Certifying GNN Classifiers on Fairness*. (2) **Algorithm Design.** We propose a framework ELEGANT to achieve certified fairness defense against attacks on both node attributes and graph structure without relying on assumptions about any specific GNNs. (3) **Experimental Evaluation.** We perform comprehensive experiments on real-world datasets to verify the effectiveness of ELEGANT.

5.1.2 Problem Definition

Preliminaries. Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be an undirected attributed network, where $\mathcal{V} = \{v_1, ..., v_n\}$ is the set of n nodes; $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. Let $\boldsymbol{A} \in \{0, 1\}^{n \times n}$ and $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ be the adjacency matrix and attribute matrix of $\mathcal{G}$, respectively. Assume each node in $\mathcal{G}$ represents an individual, and sensitive attribute s divides the population into different demographic subgroups. We follow a widely studied setting [3, 37] to assume the sensitive attribute is

binary, i.e., $s \in \{0, 1\}$. We use s_i to denote the value of the sensitive attribute for node v_i. In node classification tasks, we use $\mathcal{V}_{\text{trn}}$ and $\mathcal{V}_{\text{tst}}$ ($\mathcal{V}_{\text{trn}}, \mathcal{V}_{\text{tst}} \in \mathcal{V}$) to represent the training and test node set, respectively. We denote the GNN classifier as $f_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$. $f_{\boldsymbol{\theta}}$ takes $\boldsymbol{A}$ and $\boldsymbol{X}$ as input, and outputs $\hat{\boldsymbol{Y}}$ as the predictions for the nodes in $\mathcal{G}$. Each row in $\hat{\boldsymbol{Y}}$ is a one-hot prediction. We use $f_{\boldsymbol{\theta}^*}$ to denote the GNN with optimal parameter $\boldsymbol{\theta}^*$.

Threat Model. We focus on the attacking scenario of model evasion, i.e., the attack happens in the test phase. In particular, we assume that the victim model under attack is an optimized GNN node classifier $f_{\boldsymbol{\theta}^*}$. We follow a widely adopted setting [10, 247, 136, 143] to assume that a subset of nodes $\mathcal{V}_{\text{vul}} \in \mathcal{V}_{\text{tst}}$ are vulnerable to attacks. Specifically, attackers may perturb their links (i.e., flip the edge existence) to other nodes and/or their node attributes (i.e., change their attribute values). We denote the perturbations on adjacency matrix as $\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}}$. Here $\oplus$ denotes the element-wise XOR operator; $\boldsymbol{\Delta}_{\boldsymbol{A}} \in \{0, 1\}^{n \times n}$ is the matrix representing the perturbations made by the attacker, where 1 only appears in rows and columns associated with the vulnerable nodes while 0 appears elsewhere. Correspondingly, in $\boldsymbol{\Delta}_{\boldsymbol{A}}$, 1 entries represent edges that attackers intend to flip, while 0 entries are associated with edges that are not attacked. Similarly, we denote the perturbations on node attribute matrix as $\boldsymbol{X} + \boldsymbol{\Delta}_{\boldsymbol{X}}$, where $\boldsymbol{\Delta}_{\boldsymbol{X}} \in \mathbb{R}^{n \times n}$ is the matrix representing the perturbations made by the attacker. Usually, if the total magnitude of perturbations is within certain budgets (i.e., $\|\boldsymbol{\Delta}_{\boldsymbol{A}}\|_0 \leq \epsilon_{\boldsymbol{A}}$ for $\boldsymbol{A}$ and $\|\boldsymbol{\Delta}_{\boldsymbol{X}}\|_2 \leq \epsilon_{\boldsymbol{X}}$ for $\boldsymbol{X}$), the perturbations are regarded as unnoticeable. The goal of an attacker is to add unnoticeable perturbations to nodes in $\mathcal{V}_{\text{vul}}$, such that the GNN predictions for nodes in $\mathcal{V}_{\text{tst}}$ based on the perturbed graph exhibit as much bias as possible. In addition, we assume that the attacker has access to any information about the victim GNN (i.e., a white-box setting). This is the worst case in practice, which makes it even more challenging to achieve defense.

To defend against the aforementioned attacks, we aim to establish a node classifier on top of an optimized GNN backbone, such that this classifier, theoretically, will not exhibit more bias than a given threshold no matter what unnoticeable perturbations (i.e., perturbations within budgets) are added. We formally formulate the problem of *Certifying GNN Classifiers on Fairness* below.

PROBLEM 5.1.1. ***Certifying GNN Classifiers on Fairness.*** *Given an attributed network $\mathcal{G}$, a test node set $\mathcal{V}_{tst}$, a vulnerable node set $\mathcal{V}_{vul} \in \mathcal{V}_{tst}$, a threshold η for the exhibited bias, and an optimized GNN classifier $f_{\boldsymbol{\theta}^*}$, our goal is to achieve a classifier on top of $f_{\boldsymbol{\theta}^*}$ associated with budgets $\epsilon_{\boldsymbol{A}}$ and $\epsilon_{\boldsymbol{X}}$, such that this classifier will bear comparable utility with $f_{\boldsymbol{\theta}^*}$ but provably not exhibit more bias than η on the nodes in $\mathcal{V}_{tst}$, no matter what unnoticeable node attributes and/or graph structure perturbations (i.e., perturbations within budgets) are made over the nodes in $\mathcal{V}_{vul}$.*

5.1.3 The Proposed Framework – ELEGANT

Here we first introduce the modeling of attack and defense on the fairness of GNNs, then discuss how we achieve certified defense on node attributes. After that, we propose a strategy to achieve both types of certified defense (i.e., defense on node attributes and graph structure) at the same time. Finally, we introduce strategies to achieve the designed certified fairness defense for GNNs in practice.

5.1.3.1 Bias Indicator Function

We first construct an indicator g to mathematically model the attack and defense on the fairness of GNNs. Our rationale is to use g to indicate whether the predictions of $f_{\boldsymbol{\theta}^*}$ exhibit a level of bias exceeding a given threshold. We present the formal definition below.

DEFINITION 5.1.1. *(Bias Indicator Function) Given adjacency matrix $\boldsymbol{A}$ and node attribute matrix $\boldsymbol{X}$, a test node set $\mathcal{V}_{tst}$, a threshold η for the exhibited bias, and an optimized GNN model $f_{\boldsymbol{\theta}^*}$, the bias indicator function is defined as $g(f_{\boldsymbol{\theta}^*}, \boldsymbol{A}, \boldsymbol{X}, \eta, \mathcal{V}_{tst}) = \mathbb{1}\left(\pi(f_{\boldsymbol{\theta}^*}(\boldsymbol{A}, \boldsymbol{X}), \mathcal{V}_{tst}) < \eta\right)$, where $\mathbb{1}(\cdot)$ takes an event as input and outputs 1 if the event happens (otherwise 0); $\pi(\cdot, \cdot)$ denotes the bias metric for predictions (taken as its first parameter) over a set of nodes (taken as its second parameter). Traditional bias metrics include Δ_{SP} [37, 58] and Δ_{EO} [37, 77].*

Correspondingly, the goal of the attacker is to ensure that the indicator g outputs 0 for an η as large as possible, while the goal of certified defense is to ensure for a given threshold η, the indicator g provably yields 1 as long as the attacks are within certain budgets. Note that a reasonable η should ensure that g outputs 1 based on the clean graph data (i.e., graph data without any attacks). Below we first discuss the certified fairness defense over node attributes to maintain the output of g as 1.

5.1.3.2 Certified Fairness Defense over Node Attributes

We now introduce how we achieve certified defense over the node attributes for the fairness of the predictions yielded by $f_{\boldsymbol{\theta}^*}$. Specifically, we propose to construct a smoothed bias indicator function $\tilde{g}_{\boldsymbol{X}}(f_{\boldsymbol{\theta}^*}, \boldsymbol{A}, \boldsymbol{X}, \mathcal{V}_{\text{vul}}, \eta)$ via adding Gaussian noise over the node attributes of vulnerable nodes in $\mathcal{V}_{\text{vul}}$. For simplicity, we use $\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A}, \boldsymbol{X})$ to represent the smoothed bias indicator function over node attributes by omitting $\mathcal{V}_{\text{vul}}$, $f_{\boldsymbol{\theta}^*}$ and η. We define $\tilde{g}_{\boldsymbol{X}}$ below.

DEFINITION 5.1.2. *(Bias Indicator with Node Attribute Smoothing) We define the bias indicator with smoothed node attributes over the nodes in $\mathcal{V}_{vul}$ as $\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A}, \boldsymbol{X}) = \text{argmax}_{c\in\{0,1\}}\Pr(g(f_{\boldsymbol{\theta}^*}, \boldsymbol{A}, \boldsymbol{X} + \gamma_{\boldsymbol{X}}(\boldsymbol{\omega}_{\boldsymbol{X}}, \mathcal{V}_{vul}), \eta, \mathcal{V}_{tst}) = c)$. Here $\boldsymbol{\omega}_{\boldsymbol{X}}$ is a $(d \cdot |\mathcal{V}_{vul}|)$-dimensional vector, where each entry is a random variable following a Gaussian Distribution $\mathcal{N}(0, \sigma^2)$; $\gamma_{\boldsymbol{X}}(\cdot, \cdot)$ maps a vector (its first parameter) to an $(n \times d)$-dimensional matrix, where the vector values are assigned to rows whose indices associate with the indices of a set of nodes (its second parameter) while other matrix entries are zeros.*

We denote $\boldsymbol{\Gamma}_{\boldsymbol{X}} = \gamma_{\boldsymbol{X}}(\boldsymbol{\omega}_{\boldsymbol{X}}, \mathcal{V}_{\text{vul}})$ and $g(\boldsymbol{A}, \boldsymbol{X} + \boldsymbol{\Gamma}_{\boldsymbol{X}}) = g(f_{\boldsymbol{\theta}^*}, \boldsymbol{A}, \boldsymbol{X} + \gamma_{\boldsymbol{X}}(\boldsymbol{\omega}_{\boldsymbol{X}}, \mathcal{V}_{\text{vul}}), \eta, \mathcal{V}_{\text{tst}})$ below for simplicity. We are then able to derive the theoretical certification for the defense on fairness with the defined $\tilde{g}_{\boldsymbol{X}}$ in Definition 5.1.2. We now present the defense certification on fairness below.

THEOREM 5.1.1. *(Certified Fairness Defense for Node Attributes) Denote the probability for $g(\boldsymbol{A}, \boldsymbol{X} + \boldsymbol{\Gamma}_{\boldsymbol{X}})$ to return class c ($c \in \{0, 1\}$) as $P(c)$. Then $\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A}, \boldsymbol{X})$ will provably return $\text{argmax}_{c\in\{0,1\}}P(c)$ for any perturbations (over the attributes of vulnerable nodes) within an l_2 radius $\tilde{\epsilon_{\boldsymbol{X}}} = \frac{\sigma}{2}\left(\Phi^{-1}(\max_{c\in\{0,1\}} P(c)) - \Phi^{-1}(\min_{c\in\{0,1\}} P(c))\right)$, where $\Phi^{-1}(\cdot)$ is the inverse of the standard Gaussian cumulative distribution function.*

Correspondingly, for an η that enables $\max_{c\in\{0,1\}} P(c) = 1$, it is then safe to say that no matter what perturbations $\boldsymbol{\Delta}_{\boldsymbol{X}}$ are made on vulnerable nodes, as long as $\|\boldsymbol{\Delta}_{\boldsymbol{X}}\|_2 \leq \tilde{\epsilon}_{\boldsymbol{X}}$, the constructed $\tilde{g}_{\boldsymbol{X}}$ will provably not yield predictions for $\mathcal{V}_{\text{tst}}$ with a level of bias exceeding η. Nevertheless, it is worth noting that, in GNNs, perturbations may also be made on the structure of the vulnerable nodes, i.e., adding and/or deleting edges between these vulnerable nodes and any nodes in the graph. Hence it is also necessary to achieve certified defense against such structural attacks. Here we propose to also smooth the constructed $\tilde{g}_{\boldsymbol{X}}$ over the graph structure (of the vulnerable nodes) for the purpose of certified fairness defense on the graph structure. However, the adjacency matrix describing the graph structure is naturally binary, and thus should be smoothed in a different way.

5.1.3.3 Certified Fairness Defense over Node Attributes and Graph Structure

We then introduce achieving certified fairness defense against attacks on both node attributes and graph structure. We propose a strategy to leverage noise following Bernoulli distribution to smooth $\tilde{g}_{\boldsymbol{X}}$ over the rows and columns (due to symmetricity) associated with the vulnerable nodes in $\boldsymbol{A}$. In this way, we can smooth both the node attributes and graph structure for g in a randomized manner, and we denote the constructed function as $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$. We present the formal definition below.

DEFINITION 5.1.3. *(Bias Indicator with Attribute-Structure Smoothing) We define the bias indicator function with smoothed node attributes and graph structure over the nodes in $\mathcal{V}_{vul}$ as $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}(\boldsymbol{A},\boldsymbol{X}) = \text{argmax}_{c\in\{0,1\}}\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A}\oplus\gamma_{\boldsymbol{A}}(\boldsymbol{\omega}_{\boldsymbol{A}},\mathcal{V}_{vul}),\boldsymbol{X}) = c)$. Here $\boldsymbol{\omega}_{\boldsymbol{A}}$ is an $(n\cdot|\mathcal{V}_{vul}|)$-dimensional random variable, where each dimension takes 0 and 1 with the probability of β $(0.5 < \beta \leq 1)$ and $1-\beta$, respectively; function $\gamma_{\boldsymbol{A}}(\cdot,\cdot)$ maps a vector (its first parameter) to a symmetric $(n\times n)$-dimensional matrix, where the vector values are assigned to rows whose indices associated with the indices of a set of nodes (its second parameter) and then mirrored to the corresponding columns, while other values are left as zeros.*

We let $\boldsymbol{\Gamma}_{\boldsymbol{A}} = \gamma_{\boldsymbol{A}}(\boldsymbol{\omega}_{\boldsymbol{A}},\mathcal{V}_{\text{vul}})$ below for simplicity. To better illustrate how classifier $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ achieves certified fairness defense over both data modalities of an attributed network, we provide an exemplary case in Fig. 5.1. Here we assume node $v_i \in \mathcal{V}_{\text{vul}}$. Considering the high dimensionality of node attributes and adjacency matrix, we only analyze two entries $\boldsymbol{X}_{i,j}$ and $\boldsymbol{A}_{i,j}$ and omit other entries after noise for simplicity. Here the superscript (i,j) represents the i-th row and j-th column of a matrix. Under binary noise, entry $\boldsymbol{A}_{i,j}$ only has two possible values, i.e., $\boldsymbol{A}_{i,j}\oplus 0$ and $\boldsymbol{A}_{i,j}\oplus 1$. We denote the two cases as Case (1) and Case (2), respectively. We assume that the area where g returns 1 in the span of the two input random entries of g (i.e., $\boldsymbol{X}_{i,j}$ and $\boldsymbol{A}_{i,j}$ under random noise) is an ellipse (marked out with green), where the decision boundary is marked out with deep green. In Case (1), $\boldsymbol{X}_{i,j}$ under random noise follows a Gaussian distribution, whose probability density function is marked out as deep red. We assume that, in this case, the integral of the probability density function within the range of the ellipse (marked out with shallow red) is larger than 0.5. Correspondingly, according to Definition 5.1.2, $\tilde{g}_{\boldsymbol{X}}$ returns 1 in this case. In Case (2), we similarly mark out the probability density function and the area used for integral within the range of the ellipse. We assume that in this case, the integral is smaller than 0.5, and thus $\tilde{g}_{\boldsymbol{X}}$ returns 0. Note that to compute the output of $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$, we need to identify the output of $\tilde{g}_{\boldsymbol{X}}$ with the largest probability. Notice that $\beta > 0.5$, we have that $\boldsymbol{A}_{i,j}\oplus 0$ happens with a larger

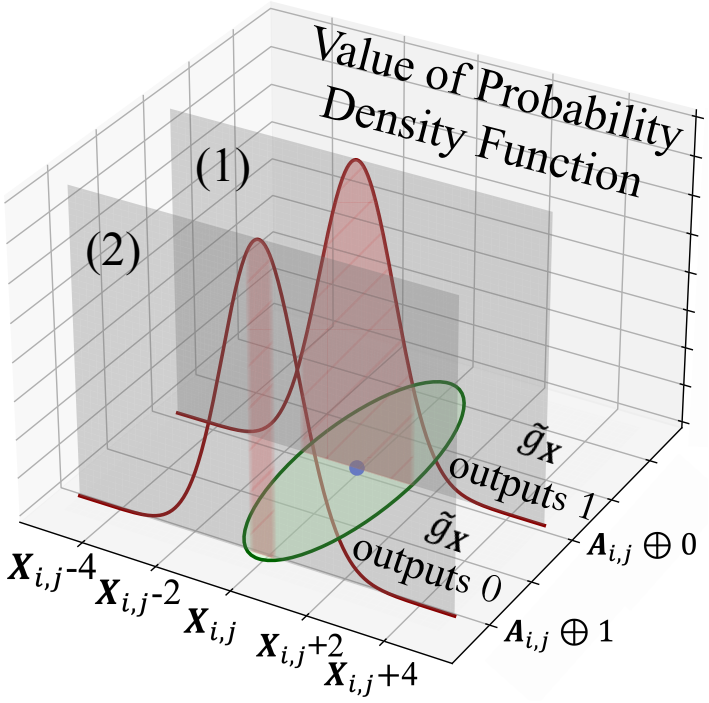


FIGURE 5.1. An example illustrating how ELEGANT works.

probability than $\boldsymbol{A}_{i,j} \oplus 1$. Therefore, $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ outputs 1 in this example. In other words, the bias level of the predictions of $f_{\boldsymbol{\theta}^*}$ is satisfying (i.e., smaller than η) based on $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$.

Below we introduce a desirable property of $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$, i.e., certified fairness defense associated with tractable budgets over both node attributes and graph topology can be achieved.

LEMMA 1. *(Perturbation-Invariant Budgets Existence) There exist tractable budgets $\epsilon_{\boldsymbol{A}}$ and $\epsilon_{\boldsymbol{X}}$, such that for any perturbations made over the node attributes and graph structure of the vulnerable nodes within $\epsilon_{\boldsymbol{A}}$ and $\epsilon_{\boldsymbol{X}}$, $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ provably maintains the same classification results.*

Correspondingly, for an η that enables $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ to return 1, we are then able to achieve certified fairness defense over $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ against perturbations on both node attributes and graph structure. Below we derive the certified fairness defense budgets over the graph structure $\epsilon_{\boldsymbol{A}}$ and node attributes $\epsilon_{\boldsymbol{X}}$ for $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$. We first introduce the derivation of $\epsilon_{\boldsymbol{A}}$. Here, our rationale is: considering that $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ is a binary classifier, we need to ensure that under structure attacks, the probability of $\tilde{g}_{\boldsymbol{X}}$ returning 1 (denoted as $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1)$) is provably greater than 0.5, such that $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ will still return 1. To this end, we propose to derive a lower bound of $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1)$, which we denote as $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$. Finally, we identify the largest perturbation size that keeps such a lower bound larger than 0.5, and the identified perturbation size is then the graph structure perturbation budget. We present the lower bound of $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1)$ below.

LEMMA 2. *(Positive Probability Bound Under Noises) There exists a tractable $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}} \in (0, 1)$, such that $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1) \geq \underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$.*

To derive the perturbation budget $\epsilon_{\boldsymbol{A}}$, we only need to find a $\boldsymbol{\Delta}_{\boldsymbol{A}}$ with the largest l_0-norm that still enables $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$ to be greater than 0.5 (according to Definition 5.1.3). Correspondingly, we derive the theoretical perturbation-invariant budget $\epsilon_{\boldsymbol{A}}$ in Theorem 5.1.2 below.

THEOREM 5.1.2. *(Certified Defense Budget for Structure Perturbations) The certified defense budget over the graph structure $\epsilon_{\boldsymbol{A}}$ for $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ is given as*

$$\epsilon_{\boldsymbol{A}} = \max \tilde{\epsilon_{\boldsymbol{A}}},\ \textit{s.t.}\ \underline{P_{\tilde{g}_{\boldsymbol{X}}=1}} > 0.5,\ \forall\ \|\boldsymbol{\Delta}_{\boldsymbol{A}}\|_0 \leq \tilde{\epsilon_{\boldsymbol{A}}}. \tag{5.1}$$

To solve the optimization problem in Eq. (5.1), we introduce Theorem 5.1.3 to compute $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$.

THEOREM 5.1.3. *(Positive Probability Lower Bound) We have $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}} = \Pr(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} \in \mathcal{H})$. Here $\mathcal{H} = \cup_{i=\mu+1}^{n\cdot|\mathcal{V}_{vul}|}\mathcal{H}_i \cup \mathcal{H}'_\mu$; $\mathcal{H}_i$ is given by*

$$\mathcal{H}_i = \Bigg\{ \bar{\boldsymbol{A}} : \frac{\Pr(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} = \bar{\boldsymbol{A}})}{\Pr(\boldsymbol{A} \oplus \boldsymbol{\Delta}_{\boldsymbol{A}} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} = \bar{\boldsymbol{A}})} = \left(\frac{\beta}{1-\beta}\right)^i, \\ \forall v_i \in \mathcal{V} \setminus \mathcal{V}_{vul}, \|\bar{\boldsymbol{A}}_i - \boldsymbol{A}_i\|_0 = 0 \Bigg\};$$

and μ is defined over the optimization problem of $\operatorname{argmax}_{-n\cdot|\mathcal{V}_{vul}| \leq j \leq n\cdot|\mathcal{V}_{vul}|} j$, s.t. $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1) \leq \Pr\left(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} \in \cup_{k=j}^{n\cdot|\mathcal{V}_{vul}|}\mathcal{H}_j\right)$. Here $\mathcal{H}'_\mu$ is any subregion of $\mathcal{H}_\mu$ that satisfies $\Pr(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} \in \mathcal{H}'_\mu) = \Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = 1) - \Pr\left(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}} \in \cup_{k=j}^{n\cdot|\mathcal{V}_{vul}|}\mathcal{H}_j\right)$.

We provide detailed steps to solve the optimization problem given in Eq. (5.1) in the online version[1]. Now we introduce the theoretical analysis of how to derive $\epsilon_{\boldsymbol{X}}$ in Theorem 5.1.4.

THEOREM 5.1.4. *(Certified Defense Budget over Node Attributes) Denote $\bar{\mathcal{A}}$ as the set of all possible $(n \times n)$-matrices, where entries in rows whose indices associate with those vulnerable nodes may take 1 or 0, while other entries are zeros. The certified defense budget $\epsilon_{\boldsymbol{X}}$ for $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$ is given as $\epsilon_{\boldsymbol{X}} = \min\{\tilde{\epsilon_{\boldsymbol{X}}} : \tilde{\epsilon_{\boldsymbol{X}}}$ is derived with classifier $\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X})$, where $\boldsymbol{\Gamma}_{\boldsymbol{A}} \in \bar{\mathcal{A}}\}$.*

5.1.3.4 Certification in Practice

Estimating the Predicted Label Probabilities. According to Definition 5.1.3, it is necessary to obtain $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}_{\boldsymbol{A}}, \boldsymbol{X}) = c)$ ($c \in \{0, 1\}$) to determine the output of classifier $\tilde{g}_{\boldsymbol{X}}$. We propose to leverage a Monte Carlo method to estimate such a probability. Specifically, we first randomly pick N samples of $\boldsymbol{\Gamma}_{\boldsymbol{A}}$ as $\bar{\mathcal{A}}'$ ($\bar{\mathcal{A}}' \subset \bar{\mathcal{A}}$). Considering the output of $\tilde{g}_{\boldsymbol{X}}$ is binary, we then follow a common strategy [33] to consider this problem as a parameter estimation of a Binomial distribution: we first count the number of returned label 1 and 0 under noise as N_1 and N_0 ($N_1 + N_0 = N$); then we choose a confidence level $1 - \alpha$ and take the α-th quantile of the beta distribution with parameters N_1 and N_0 as the estimated probability lower bound for returning label $c = 1$. We proved that all theoretical analysis still holds true for such an estimation in the online version. We follow a similar strategy to estimate the probability lower bound of yielding 1 for $g(\boldsymbol{A}, \boldsymbol{X} + \boldsymbol{\Gamma}_{\boldsymbol{X}})$.

[1]See online version here https://arxiv.org/abs/2311.02757 for supplementary discussion and experimental results.

Obtaining Fair Classification Results. After achieving certified fairness defense based on $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$, we also need to obtain the corresponding node classification results (given by $f_{\boldsymbol{\theta}^*}$) over $\mathcal{V}_{\text{tst}}$. We propose to collect all classification results associated with the sampled $\boldsymbol{\Gamma}'_{\boldsymbol{A}} \in \bar{\mathcal{A}}'$ that leads to an estimated lower bound of $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}'_{\boldsymbol{A}}, \boldsymbol{X}) = 1)$ to be larger than 0.5 as $\hat{\mathcal{Y}}'$. Here $\hat{\mathcal{Y}}'$ is a set of output matrices of $f_{\boldsymbol{\theta}^*}$, where each matrix consists of the one-hot output classification results (as each row in the matrix) for all nodes. We propose to take $\text{argmin}_{\hat{\boldsymbol{Y}}'} \pi(\hat{\boldsymbol{Y}}', \mathcal{V}_{\text{tst}}), s.t.\ \hat{\boldsymbol{Y}}' \in \hat{\mathcal{Y}}'$ as the final node classification results. Correspondingly, consider $\Pr(\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}'_{\boldsymbol{A}}, \boldsymbol{X}) = 1)$ falls into the confidence interval characterized by $1 - \alpha$, we have a neat probabilistic theoretical guarantee below.

PROPOSITION 5.1.1. *(Probabilistic Guarantee for the Fairness Level of Node Classification). For* $\hat{\boldsymbol{Y}} = \text{argmin}_{\hat{\boldsymbol{Y}}'} \pi(\hat{\boldsymbol{Y}}', \mathcal{V}_{tst}), s.t.\ \hat{\boldsymbol{Y}}' \in \hat{\mathcal{Y}}'$*, we have* $\Pr(\pi(\hat{\boldsymbol{Y}}, \mathcal{V}_{tst}) > \eta) < 0.5^{|\hat{\mathcal{Y}}'|}$.

Note that for a large enough sample size N, the cardinality of $\hat{\mathcal{Y}}'$ also tends to be large in practice. Hence it is safe to argue that $\Pr(\pi(\hat{\boldsymbol{Y}}, \mathcal{V}_{\text{tst}}) > \eta)$ tends to be small enough. In other words, we have a probability that is large enough to obtain results with a bias level lower than threshold η.

Calculation of Perturbation Budgets. We calculate $\epsilon_{\boldsymbol{A}}$ by solving the optimization problem given in Eq. (5.1), and we provide the completed procedure in the online version. For $\epsilon_{\boldsymbol{X}}$, we utilize a Monte Carlo method to estimate its value. More specifically, we leverage $\min\{\tilde{\epsilon_{\boldsymbol{X}}} : \tilde{\epsilon_{\boldsymbol{X}}}$ is derived with classifier $\tilde{g}_{\boldsymbol{X}}(\boldsymbol{A} \oplus \boldsymbol{\Gamma}'_{\boldsymbol{A}}, \boldsymbol{X})$, where $\boldsymbol{\Gamma}'_{\boldsymbol{A}} \in \bar{\mathcal{A}}'\}$ to estimate the value of $\epsilon_{\boldsymbol{X}}$.

5.1.4 Experimental Evaluations

In this section, we aim to answer three research questions: **RQ1**: How well does ELEGANT perform in achieving certified fairness defense? **RQ2**: How does ELEGANT perform under fairness attacks compared to other popular fairness-aware GNNs? **RQ3**: How does ELEGANT perform under different settings of parameters? We present the main experimental settings and representative results in this section due to space limits. Detailed settings and supplementary experiments are in the online version.

5.1.4.1 Experimental Settings

Downstream Task and Datasets. We focus on the widely studied node classification task, which is one of the most representative tasks in the domain of learning on graphs. We adopt three real-world network datasets that are widely used to perform studies on the fairness of GNNs, namely German Credit [3, 6], Recidivism [3, 99], and Credit Defaulter [3, 227]. We provide their basic information, including how these datasets are built and their statistics, in the online version.

Evaluation Metrics. We perform evaluation from three main perspectives, including model utility, fairness, and certified defense. To evaluate utility, we adopt the node classification accuracy. To evaluate fairness, we adopt the widely used metrics Δ_{SP} (measuring bias under *Statistical Parity*) and Δ_{EO} (measuring bias under *Equal Opportunity*). To evaluate certified defense, we extend a traditional metric named *Certified Accuracy* [192, 33] in our experiments, and we name it as *Fairness Certification Rate* (FCR). Specifically, existing GNN certification

works mainly focus on a certain individual node, and utilize certified accuracy to measure the ratio of nodes that are correctly classified and also successfully certified out of all test nodes [192]. In this paper, however, we perform certified (fairness) defense for individuals over an entire test set (instead of for any specific individual). Accordingly, we propose to sample multiple test sets out of nodes that are not involved in the training and validation set. Then we perform certified fairness defense for all sampled test sets, and utilize the ratio of test sets that are successfully certified over all sampled sets as the metric of certified defense. The rationale of FCR is leveraging a Monte Carlo method to estimate the probability of being successfully certified for a randomly sampled test node set.

GNN Backbones and Baselines. Note that ELEGANT serves as a plug-and-play framework for any optimized GNNs ready to be deployed. To evaluate the generality of ELEGANT across GNNs, we adopt three of the most representative GNNs spanning across simple and complex ones, namely Graph Sample and Aggregate Networks [74] (GraphSAGE), Graph Convolutional Networks [111] (GCN), and Jumping Knowledge Networks (JK). Note that to the best of our knowledge, existing works on fairness certification cannot certify the attacks over two data modalities (i.e., continuous node attributes and binary graph topology) at the same time, and thus cannot be naively generalized onto GNNs. Hence we compare the usability of GNNs before and after certification with ELEGANT. Moreover, we also adopt two popular fairness-aware GNNs as baselines to evaluate bias mitigation, including FairGNN [37] and NIFTY [3]. Specifically, FairGNN utilizes adversarial learning to debias node embeddings, while NIFTY designs regularizations to debias node embeddings.

Threat Models. We propose to evaluate the performance of ELEGANT and other fairness-aware GNN models under actual attacks on fairness. We first introduce the threat model over graph structure. To the best of our knowledge, FA-GNN [83] is the only work that performs graph structure attacks targeting the fairness of GNNs. Hence we adopt FA-GNN to attack graph structure. In terms of node attributes, to the best of our knowledge, no existing work has made any explorations. Hence we directly utilize gradient ascend to perform attacks. Specifically, after structure attacks have been performed, we identify the top-ranked node attribute elements (out of the node attribute matrix) that positively influence the exhibited bias the most via gradient ascend. For any given budget (of attacks) on node attributes, we add perturbations to these elements in proportion to their gradients.

5.1.4.2 RQ1: Fairness Certification Effectiveness

To answer RQ1, we investigate the performance of different GNNs after certification across different real-world attributed network datasets over FCR, utility, and fairness. We present the experimental results across three GNN backbones and three real-world attributed network datasets in Table 5.1. Here bias is measured with Δ_{SP}, and we have similar observations on Δ_{EO}. We summarize the main observations as follows: (1) **Fairness Certification Rate (FCR).** We observe that ELEGANT realizes values of FCR around or even higher than 90% for all three GNN backbones and three attributed network datasets, especially for the German Credit dataset, where vanilla GNNs tend to exhibit a high level of bias. The corresponding intuition is that, for nodes in any randomly sampled test set, we have a probability around or higher than 90% to successfully certify the fairness level of the predictions yielded by the GNN model with our proposed framework ELEGANT. Hence ELEGANT achieves a satisfying

TABLE 5.1. Comparison between vanilla GNNs and certified GNNs under ELEGANT over three popular GNNs across three real-world datasets. Here ACC denotes node classification accuracy; E- prefix marks out the GNNs under ELEGANT with certification. ↑ denotes the larger, the better; ↓ denotes the opposite. Numbers are in percentage, and the bests are in bold.

	German Credit			Recidivism			Credit Defaulter		
	ACC (↑)	Bias (↓)	FCR (↑)	ACC (↑)	Bias (↓)	FCR (↑)	ACC (↑)	Bias (↓)	FCR (↑)
SAGE	67.3 ± 2.14	50.6 ± 15.9	N/A	89.8 ± 0.66	9.36 ± 3.15	N/A	**75.9** ± 2.18	13.0 ± 4.01	N/A
E-SAGE	**71.0** ± 1.27	**16.3** ± 10.9	98.7 ± 1.89	**89.9** ± 0.90	**6.39** ± 2.85	94.3 ± 6.65	73.4 ± 0.50	**8.94** ± 0.99	94.3 ± 3.30
GCN	**59.6** ± 3.64	37.4 ± 3.24	N/A	**90.5** ± 0.73	10.1 ± 3.01	N/A	**65.8** ± 0.29	11.1 ± 3.22	N/A
E-GCN	58.2 ± 1.82	**3.52** ± 3.77	96.3 ± 1.89	89.6 ± 0.74	**9.56** ± 3.22	96.0 ± 3.56	65.2 ± 0.99	**7.28** ± 1.46	92.7 ± 5.19
JK	**63.3** ± 4.11	41.2 ± 18.1	N/A	**91.9** ± 0.54	10.1 ± 3.15	N/A	76.6 ± 0.69	9.24 ± 0.60	N/A
E-JK	62.3 ± 4.07	**22.4** ± 1.95	97.0 ± 3.00	89.3 ± 0.33	**6.26** ± 2.78	89.5 ± 10.5	**77.7** ± 0.27	**3.37** ± 2.64	99.3 ± 0.47

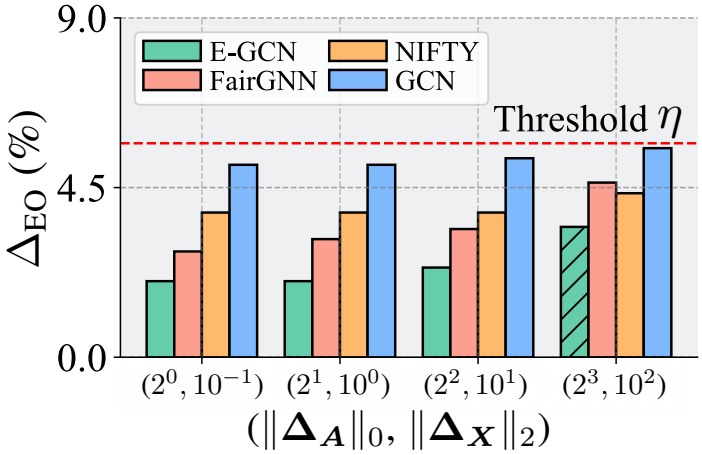
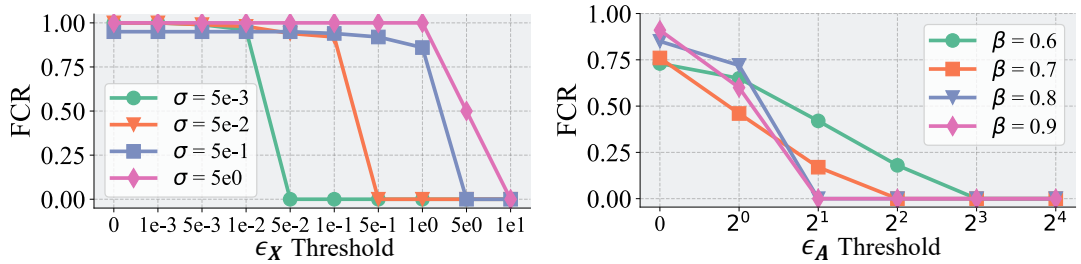


FIGURE 5.2. The bias levels of GCN, E-GCN, FairGNN, and NIFTY under fairness attacks on German Credit. The shaded bar indicates that certified budget $\epsilon_{\boldsymbol{A}} \leq \|\boldsymbol{\Delta}_{\boldsymbol{A}}\|_0$ or $\epsilon_{\boldsymbol{X}} \leq \|\boldsymbol{\Delta}_{\boldsymbol{X}}\|_2$. The y-axis is in logarithmic scale for better visualization purposes.

fairness certification rate across all adopted GNN backbones and datasets. (2) **Utility.** We found that compared with those vanilla GNN backbones, certified GNNs with ELEGANT also exhibit comparable and even higher node classification accuracy values in all cases. Hence we conclude that our proposed framework ELEGANT does not significantly jeopardize the utility of the vanilla GNN models, and those certified GNNs with ELEGANT still bear a high level of usability in terms of node classification accuracy. (3) **Fairness.** Although the goal of ELEGANT is not debiasing GNNs, we observe that certified GNNs with ELEGANT achieve better performances in all cases in terms of algorithmic fairness compared with those vanilla GNNs. This demonstrates that the proposed framework ELEGANT also contributes to bias mitigation. We conjecture that such an advantage of debiasing could be a mixed result of (1) adding random noise on node attributes and graph topology (as in Section 5.1.3.2 and Section 5.1.3.3) and (2) the proposed strategy of obtaining fair classification results (as in Section 5.1.3.4). We provide a more detailed analysis in Section 5.1.4.9.



(A) FCR of certification for σ over node attributes (B) FCR of certification for β over graph topology

FIGURE 5.3. Parameter study of σ over $\epsilon_{\boldsymbol{X}}$ (a) and β over $\epsilon_{\boldsymbol{A}}$ (b). Experimental results are presented based on GCN over German credit and Credit Defaulter for (a) and (b), respectively. Tendencies on other GNNs and datasets are similar.

5.1.4.3 RQ2: Fairness Certification under Attacks

To answer RQ2, we perform attacks on the fairness of GCN, E-GCN, FairGNN (with a GCN backbone), and NIFTY (with a GCN backbone). Considering the large size of the quadratic space spanned by the sizes of perturbations $\boldsymbol{\Delta}_{\boldsymbol{A}}$ and $\boldsymbol{\Delta}_{\boldsymbol{X}}$, we present the evaluation under four representative $(\|\boldsymbol{\Delta}_{\boldsymbol{A}}\|_0, \|\boldsymbol{\Delta}_{\boldsymbol{X}}\|_2)$ pairs. We set the threshold for bias η to be 50% higher than the fairness level of the vanilla GCN model on clean data, since it empirically helps to achieve a high certification success rate under large perturbations.

We present the fairness levels of the four models in terms of Δ_{EO} in Fig. 5.2. Note that we utilize a vanilla GCN to predict the labels for test nodes to to obtain fair classification results (as in Section 5.1.3.4), and we also have similar observations on other GNNs/datasets. (1) **Fairness.** We found that the GCN model with the proposed framework ELEGANT achieves the lowest level of bias in all cases of fairness attacks. This observation is consistent with the superiority in fairness found in Table 5.1, which demonstrates that the fairness superiority of ELEGANT maintains even under attacks within a wide range of attacking perturbation sizes. (2) **Certification on Fairness.** We now compare the performance of E-GCN across different attacking perturbation sizes. We observed that under relatively small attacking perturbation sizes, i.e., $(2^0, 10^{-1})$, $(2^1, 10^0)$, and $(2^2, 10^1)$, ELEGANT successfully achieves certification over fairness, and the bias level increases slowly as the size of attacks increases. Under relatively large attacking perturbation size, i.e., $(2^3, 10^2)$, although the attacking budgets go beyond the certified budgets, GCN under ELEGANT still exhibits a fairness level far lower than the given bias threshold η, and the fairness superiority maintains. Hence the adopted estimation strategies are safe in achieving fairness certification.

5.1.4.4 RQ3: Parameter Study

To answer RQ3, we propose to perform parameter study focusing on two most critical parameters, σ and β. To examine how σ and β influence the effectiveness of ELEGANT in terms of both FCR and certified defense budgets, we set numerical ranges for $\epsilon_{\boldsymbol{X}}$ (from 0 to 1e1) and $\epsilon_{\boldsymbol{A}}$ (from 0 to 2^4) and divide the two ranges into grids. In both ranges, we consider the dividing values of the grids as thresholds for certification budgets. In other words, under each threshold, we only consider the test sets with the corresponding certified defense

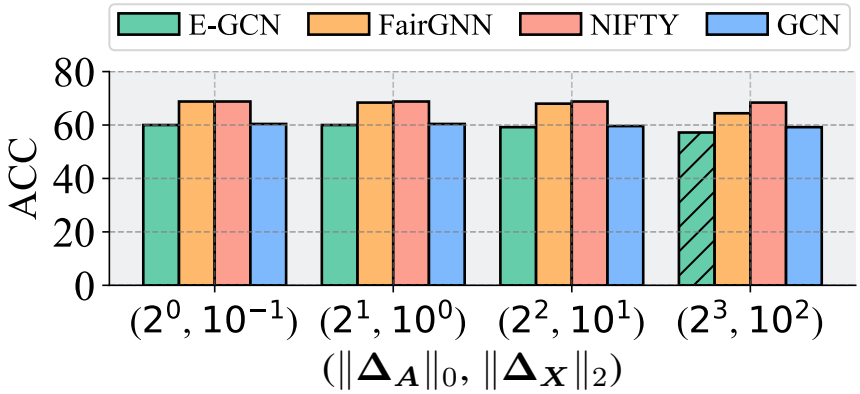


FIGURE 5.4. The utility of GCN, E-GCN, FairGNN, and NIFTY under fairness attacks on German Credit. The shaded bar indicates that certified budget $\epsilon_{\boldsymbol{A}} \leq \|\boldsymbol{\Delta}_{\boldsymbol{A}}\|_0$ or $\epsilon_{\boldsymbol{X}} \leq \|\boldsymbol{\Delta}_{\boldsymbol{X}}\|_2$.

budget being larger than this threshold as successfully certified ones, and the values of FCR are re-computed accordingly. Our rationale here is that with the thresholds (for $\epsilon_{\boldsymbol{X}}$ and $\epsilon_{\boldsymbol{A}}$) increasing, if FCR reduces slowly, this demonstrates that most successfully certified test sets are associated with large certified defense budgets. However, if FCR reduces fast, then most successfully certified test sets only bear small certified defense budgets.

Here we present the experimental results of σ and β with the most widely used GCN model based on German Credit in Fig. 5.3a and Credit Defaulter in Fig. 5.3b, respectively. We also have similar observations on other GNNs and datasets. We summarize the main observations as follows: (1) **Analysis on σ.** We observe that most cases with larger σ are associated with a larger FCR compared with the cases where σ is relatively small. In other words, larger values of σ typically make FCR reduce slower w.r.t. the increasing of $\epsilon_{\boldsymbol{X}}$ threshold. This indicates that increasing the value of σ helps realize larger certified defense budgets on node attributes, i.e., the increase of σ dominates the tendency of $\epsilon_{\boldsymbol{X}}$ given in Theorem 5.1.4. Nevertheless, it is worth mentioning that if σ is too large, the information encoded in the node attributes could be swamped by the Gaussian noise and finally corrupt the classification accuracy. Hence moderately large values for σ, e.g., 5e-1 and 5e0, are recommended. (2) **Analysis on β.** We found that (1) for cases with relatively large β (e.g., 0.8 and 0.9), the FCR also tends to be larger (compared with cases where β is smaller) at $\epsilon_{\boldsymbol{A}}$ threshold being 0. Such a tendency is reasonable, since in these cases, the expected magnitude of the added Bernoulli noise is small. Correspondingly, GNNs under ELEGANT perform similarly to vanilla GNNs, and thus an η larger than the bias level of vanilla GNNs is easier to be satisfied (compared with cases under smaller values of β); (2) for cases with relatively large β, the value of FCR reduces faster (w.r.t. $\epsilon_{\boldsymbol{A}}$ threshold) than cases where β is smaller. Therefore, we recommend that for any test set of nodes: (1) if the primary goal is to achieve certification with a high probability, then larger values for β (e.g., 0.8 and 0.9) would be preferred; (2) if the goal is to achieve certification with larger certified defense budgets on the graph topology, then smaller values for β (e.g., 0.6 and 0.7) should be selected.

TABLE 5.2. Comparison between vanilla GNNs and certified GNNs under ELEGANT over three popular GNNs across three real-world datasets. Here ACC is node classification accuracy, and E- prefix marks out the GNNs under ELEGANT with certification. ↑ denotes the larger, the better; ↓ denotes the opposite. Different from the table in Section 5.1.4.2 (where the bias is measured with Δ_{SP}), the bias is measured with Δ_{EO} here. Numerical values are in percentage, and the best ones are in bold.

	German Credit			Recidivism			Credit Defaulter		
	ACC (↑)	**Bias (↓)**	**FCR (↑)**	**ACC (↑)**	**Bias (↓)**	**FCR (↑)**	**ACC (↑)**	**Bias (↓)**	**FCR (↑)**
SAGE	67.3 ± 2.14	41.8 ± 11.0	N/A	89.8 ± 0.66	6.09 ± 3.10	N/A	**75.9** ± 2.18	10.4 ± 1.59	N/A
E-SAGE	**72.2** ± 1.26	**8.63** ± 6.15	100 ± 0.00	**90.8** ± 0.97	**3.12** ± 3.64	81.0 ± 13.0	73.4 ± 0.61	**7.18** ± 1.06	88.7 ± 6.02
GCN	**59.6** ± 3.64	35.0 ± 4.77	N/A	**90.5** ± 0.73	6.35 ± 1.65	N/A	**65.8** ± 0.29	13.5 ± 4.23	N/A
E-GCN	58.8 ± 3.74	**29.8** ± 6.82	93.3 ± 8.73	89.3 ± 0.92	**3.93** ± 3.12	96.0 ± 4.97	63.5 ± 0.37	**9.12** ± 0.95	80.5 ± 14.5
JK	63.3 ± 4.11	37.7 ± 15.9	N/A	**91.9** ± 0.54	5.26 ± 3.25	N/A	76.6 ± 0.69	8.04 ± 0.57	N/A
E-JK	**63.4** ± 3.68	**31.2** ± 15.5	93.7 ± 8.96	90.1 ± 0.55	**2.54** ± 1.62	83.7 ± 8.96	**76.9** ± 0.86	**2.90** ± 2.04	95.7 ± 4.80

5.1.4.5 Evaluation of Model Utility

In Section 5.1.4.3, we present the comparison between ELEGANT and baseline models over the fairness level under attacks. We now present the comparison over the utility under attacks. Specifically, we utilize node classification accuracy as the indicator of model utility, and we present the results in Fig. 5.4. The fairness-aware GNNs are found to exhibit better utility compared with the vanilla GNNs, which is a common observation consistent with a series of existing works [3, 47]. More importantly, we observe that the ELEGANT does not jeopardize the performance of GNN compared with the utility of the vanilla GNN. This demonstrate a high level of usability for ELEGANT in real-world applications.

5.1.4.6 Certification under Different Fairness Metrics

In Section 5.1.4.2, we present the experimental results based on the fairness metric of Δ_{SP}, which measures the exhibited bias under the fairness notion of *Statistical Parity*. We also perform the experiments based on Δ_{EO}, which measures the exhibited bias under the fairness notion of *Equal Opportunity*. We present the experimental results in Table 5.2. We summarize the observations below. (1) **Fairness Certification Rate (FCR).** We observe that ELEGANT realizes large values of FCR (larger than 80%) for all three GNN backbones and three attributed network datasets. Similar to our discussion in Section 5.1.4.2, this demonstrate that for nodes in any randomly sampled test set, we have a probability around or larger than 80% to successfully certify the fairness level of the predictions yielded by the GNN model with our proposed framework ELEGANT. As a consequence, we argue that ELEGANT also achieves a satisfying fairness certification rate across all adopted GNN backbones and datasets on the basis of Δ_{EO}. In addition, we also observe that the German Credit dataset bears relatively larger values of FCR, while the values of FCR are relatively smaller with relatively larger standard deviation values on Recidivism and Credit Defaulter datasets. A possible reason is that we set the threshold (i.e., η) as a value 25% higher than the bias exhibited by the vanilla

GNNs. Consequently, if the vanilla GNNs already exhibit a low level of bias, the threshold determined with such a strategy could be hard to satisfy under the added noise. This evidence indicates that the proposed framework ELEGANT tends to deliver better performance under scenarios where vanilla GNNs exhibit a high level of bias with the proposed strategy. (2) **Utility.** Compared with vanilla GNNs, certified GNNs with ELEGANT exhibit comparable and even higher node classification accuracy values in all cases. Therefore, we argue that the proposed framework ELEGANT does not significantly jeopardize the utility of the vanilla GNN models in certifying the fairness level of node classification. (3) **Fairness.** We observe that certified GNNs with ELEGANT are able to achieve better performances in terms of algorithmic fairness compared with those vanilla GNNs. This evidence indicates that the proposed framework ELEGANT also helps to mitigate the exhibited bias (by the backbone GNN models). We conjecture that such bias mitigation should be attributed to the same reason discussed in Section 5.1.4.2.

Algorithm 3 Certified Defense on the Fairness of GNNs

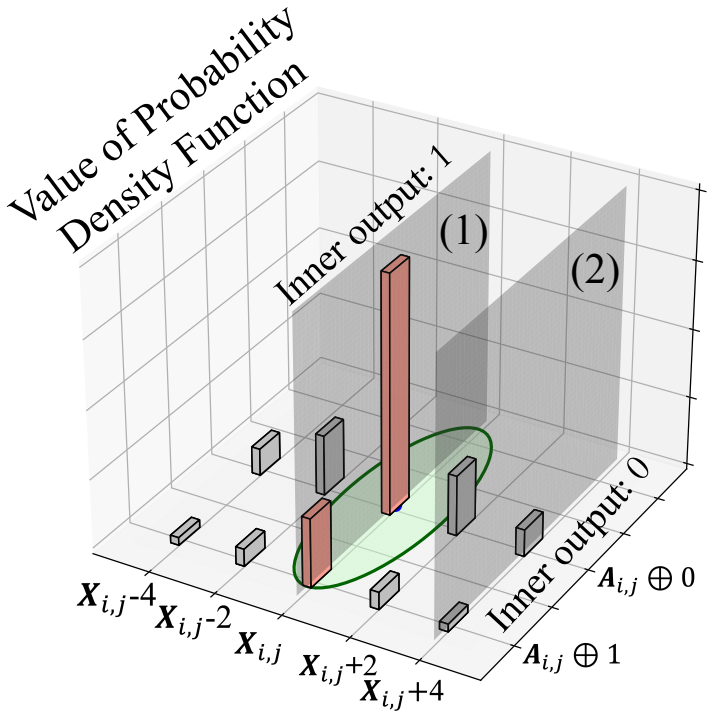
Input:

$\mathcal{G}$: graph data with potential malicious attacks; $f_{\boldsymbol{\theta}^*}$: an optimized GNN node classifier; $\mathcal{V}_{\text{train}}$, $\mathcal{V}_{\text{validation}}$, $\mathcal{V}_{\text{test}} \in \mathcal{V}$: the node set for training, validation, and test, respectively; $\mathcal{V}_{\text{vul}} \in \mathcal{V}_{\text{test}}$: the set of vulnerable nodes that may bear attacks (on node attributes and/or graph topology); N_1, N_2: sample size for the set of Bernoulli and Gaussian noise, respectively; η: a given threshold for the exhibited bias; α: the parameter to indicate the confidence level $(1-\alpha)$ of the estimation; σ: the std of the added Gaussian noise; β: the probability of returning zero of the added Bernoulli noise;

Output:

$\epsilon_{\boldsymbol{A}}$: the certified defense budget over the adjacency matrix $\boldsymbol{A}$; $\epsilon_{\boldsymbol{X}}$: the certified defense budget over the node attribute matrix $\boldsymbol{X}$; $\hat{\boldsymbol{Y}}'$: the output node classification results from the certified classifier;

Sample a set of Bernoulli noise $\mathcal{Q}_{\text{B}}$ containing N_1 samples;
Sample a set of Gaussian noise $\mathcal{Q}_{\text{G}}$ containing N_2 samples;
for $\omega_{\boldsymbol{A}} \in \mathcal{Q}_{\text{B}}$ **do**
  for $\omega_{\boldsymbol{X}} \in \mathcal{Q}_{\text{G}}$ **do**
    Calculate and collect the output of $f_{\boldsymbol{\theta}^*}$ under the noise of $\omega_{\boldsymbol{A}}$ and $\omega_{\boldsymbol{X}}$;
    Calculate and collect the output of g based on the output of $f_{\boldsymbol{\theta}^*}$;
  end for
  Under $\mathcal{Q}_{\text{G}}$, collect the number of g returning 1 and 0 as n_1 and n_0, respectively;
  Estimate the lower bound of returning c as $\underline{P_{g=c}}$ determined by the larger one between n_1 and n_0;
  if $n_1 > n_0$ and $\underline{P_{g=1}}$ is larger than 0.5 with a confidence level larger than $1-\alpha$ **or** $n_1 < n_0$ and $\underline{P_{g=0}}$ is larger than 0.5 with a confidence level larger than $1-\alpha$ **then**
    Calculate and collect the value of $\tilde{\epsilon}_{\boldsymbol{X}}$;
  else
    return ABSTAIN
  end if
end for
Collect the number of cases where $n_1 > n_0$ and estimate the lower bound of returning 1 as $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$;
if $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$ is larger than 0.5 with a confidence level larger than $1-\alpha$ **then**
  Calculate $\epsilon_{\boldsymbol{X}}$ (out of the collected $\tilde{\epsilon}_{\boldsymbol{X}}$) and $\epsilon_{\boldsymbol{A}}$ (based on the estimated $\underline{P_{\tilde{g}_{\boldsymbol{X}}=1}}$);
  Find $\boldsymbol{Y}'$ out of the collected output of $f_{\boldsymbol{\theta}^*}$;
  return $\boldsymbol{Y}'$, $\epsilon_{\boldsymbol{X}}$, and $\epsilon_{\boldsymbol{A}}$;
else
  return ABSTAIN
end if



(A) Δ_{SP}

FIGURE 5.5. An example illustrating how ELEGANT works with a different order to achieve certified defense.

5.1.4.7 Ordering the Inner and Outer Defense

We first review the general pipeline to achieve certified fairness defense. Specifically, we first model the fairness attack and defense by formulating the bias indicator function g. Then, we achieve certified defense over the node attributes for g, which leads to classifier $\tilde{g}_{\boldsymbol{X}}$. Finally, we realize certified defense for $\tilde{g}_{\boldsymbol{X}}$ over the graph topology, which leads to classifier $\tilde{g}_{\boldsymbol{A},\boldsymbol{X}}$. In general, we may consider the certified defense over node attributes and graph topology as the inner certified classifier and outer certified classifier, respectively. Now, a natural question is: *is it possible to achieve certified defense in a different order, i.e., first achieve certified defense over the graph topology (as the inner classifier), and then realize certified defense over the node attributes (as the outer classifier)?* Note that this is not the research focus of this paper, but we will provide insights about this question. In fact, it is also feasible to achieve certified defense in the reversed order compared with the approach presented in our paper. We provide an illustration in Fig. 5.5. We follow a similar setting to plot this figure as in Section 5.1.3.3. Specifically, in case (1), both $\boldsymbol{A}_{i,j} \oplus 0$ and $\boldsymbol{A}_{i,j} \oplus 1$ lead to a positive outcome for g; in case (2), both $\boldsymbol{A}_{i,j} \oplus 0$ and $\boldsymbol{A}_{i,j} \oplus 1$ lead to a negative outcome. However, considering the Gaussian distribution around $\boldsymbol{X}_{i,j}$, samples will fall around case (1) with a much higher number compared with case (2). Hence, in this example, it would be reasonable to assume that the classifier with Bernoulli noise over graph topology (the inner certified classifier) will return 1 with a higher probability. This example thus illustrates how certification following a different order returns 1.

However, such a formulation bears higher computational costs in calculating the certified budgets. The reason is that we are able to utilize a closed-form solution to calculate $\epsilon_{\boldsymbol{X}}$ based

on a set of Gaussian noise and the corresponding output from the bias indicator function. However, based on a set of Bernoulli noise and the corresponding output from the bias indicator function, we will need to solve the optimization problem given in Theorem 5.1.2 to calculate $\epsilon_{\boldsymbol{A}}$, which bears a higher time complexity than calculating $\epsilon_{\boldsymbol{X}}$. If we follow the strategy provided in Section 5.1.3.4 to calculate the inner and outer certification budgets, the certified budget of the inner certification will always be calculated multiple times, while the certified budget of the outer certification will only be calculated once. Considering the high computational cost of calculating $\epsilon_{\boldsymbol{A}}$, we thus argue that it is more efficient to realize the certification over graph topology as the outer certified classifier.

5.1.4.8 Time Complexity Analysis

We now present a comprehensive analysis on the time complexity of ELEGANT. We present the analysis from both theoretical and experimental perspectives.

Theoretical. The time complexity is linear w.r.t. the total number of the random perturbations N, i.e., $\mathcal{O}(N)$. We perform 30,000 random perturbations over the span of node attributes and graph structure. We note that the actual running time is acceptable since the certification does not require re-training (which is the most costly process). In addition, all runnings do not rely on the prediction results from each other. Hence they can be paralleled altogether theoretically to further reduce the running time.

Experimental. We perform a study of running time, and we present the results in Table 5.3. Specifically, we compare the running time of a successful certification under 30,000 random noise samples and a regular training-inference cycle with vanilla GCN. We observe that (1) although ELEGANT improves the computational cost compared with the vanilla GNN backbones, the running time remains acceptable; and (2) ELEGANT has less running time growth rate on larger datasets. For example, E-SAGE has around 10x running time on German Credit (a smaller dataset) while only around 4x on Credit Default (a larger dataset) compared to vanilla SAGE. Hence we argue that ELEGANT bears a high level of usability in terms of complexity and running time.

TABLE 5.3. Comparison of running time (in seconds) on different datasets using different methods.

	German	**Recidivism**	**Credit**
SAGE	5.27 ± 0.38	34.14 ± 1.08	40.11 ± 0.36
E-SAGE	$\mathbf{53.23 \pm 1.31}$	$\mathbf{137.12 \pm 58.66}$	$\mathbf{157.51 \pm 37.21}$
GCN	5.59 ± 0.37	34.94 ± 1.16	40.59 ± 0.32
E-GCN	$\mathbf{53.79 \pm 30.19}$	$\mathbf{212.94 \pm 10.38}$	$\mathbf{214.11 \pm 10.31}$
JK	5.78 ± 0.43	34.68 ± 0.88	39.44 ± 1.56
E-JK	$\mathbf{59.99 \pm 25.01}$	$\mathbf{238.37 \pm 1.81}$	$\mathbf{252.99 \pm 17.03}$

5.1.4.9 Additional Results on Different GNN Backbones & Baselines

We perform additional experiments over two popular GNNs, including APPNP [112] and GCNII [27], to evaluate the generalization ability of ELEGANT onto different backbones.

TABLE 5.4. Performance comparison of classification accuracy. Numbers are in percentage.

	German	Recidivism	Credit
SAGE	67.3 ± 2.14	89.8 ± 0.66	75.9 ± 2.18
E-SAGE	$\mathbf{71.0 \pm 1.27}$	$\mathbf{89.9 \pm 0.90}$	$\mathbf{73.4 \pm 0.50}$
GCN	59.6 ± 3.64	90.5 ± 0.73	65.8 ± 0.29
E-GCN	$\mathbf{58.2 \pm 1.82}$	$\mathbf{89.6 \pm 0.74}$	$\mathbf{65.2 \pm 0.99}$
JK	63.3 ± 4.11	91.9 ± 0.54	76.6 ± 0.69
E-JK	$\mathbf{62.3 \pm 4.07}$	$\mathbf{89.3 \pm 0.33}$	$\mathbf{77.7 \pm 0.27}$
APPNP	69.9 ± 2.17	95.3 ± 0.78	74.4 ± 3.05
E-APPNP	$\mathbf{69.4 \pm 0.83}$	$\mathbf{95.9 \pm 0.02}$	$\mathbf{74.6 \pm 0.32}$
GCNII	60.9 ± 1.00	90.4 ± 0.95	77.7 ± 0.22
E-GCNII	$\mathbf{60.4 \pm 4.45}$	$\mathbf{88.8 \pm 0.24}$	$\mathbf{77.6 \pm 0.02}$

TABLE 5.5. Comparison of fairness (measured with Δ_{SP}). Numbers are in percentage.

	German	Recidivism	Credit
SAGE	50.6 ± 15.9	9.36 ± 3.15	13.0 ± 4.01
E-SAGE	$\mathbf{16.3 \pm 10.9}$	$\mathbf{6.39 \pm 2.85}$	$\mathbf{8.94 \pm 0.99}$
GCN	37.4 ± 3.24	10.1 ± 3.01	11.1 ± 3.22
E-GCN	$\mathbf{3.52 \pm 3.77}$	$\mathbf{9.56 \pm 3.22}$	$\mathbf{7.28 \pm 1.46}$
JK	41.2 ± 18.1	10.1 ± 3.15	9.24 ± 0.60
E-JK	$\mathbf{22.4 \pm 1.95}$	$\mathbf{6.26 \pm 2.78}$	$\mathbf{3.37 \pm 2.64}$
APPNP	27.4 ± 4.81	9.71 ± 3.57	12.3 ± 3.14
E-APPNP	$\mathbf{13.1 \pm 5.97}$	$\mathbf{2.23 \pm 0.04}$	$\mathbf{10.8 \pm 0.07}$
GCNII	51.4 ± 0.36	9.70 ± 3.37	7.62 ± 0.29
E-GCNII	$\mathbf{24.9 \pm 0.47}$	$\mathbf{3.78 \pm 0.93}$	$\mathbf{1.72 \pm 0.81}$

TABLE 5.6. Performance in FCR on different datasets and backbone GNNs. Numbers are in percentage.

	German	Recidivism	Credit
E-SAGE	98.7 ± 1.89	94.3 ± 6.65	94.3 ± 3.3
E-GCN	96.3 ± 1.89	96.0 ± 3.56	92.7 ± 5.19
E-JK	97.0 ± 3.00	89.5 ± 10.5	99.3 ± 0.47
E-APPNP	97.8 ± 3.14	87.1 ± 3.79	95.5 ± 6.43
E-GCNII	94.7 ± 5.27	92.9 ± 9.93	99.0 ± 1.41

We present all numerical results in Table 5.4 (in terms of accuracy), 5.5 (in terms of fairness), and 5.6 (in terms of FCR). We observe that ELEGANT achieves comparable utility, a superior level of fairness, and a large percentage of FCR. This verifies the satisfying usability of ELEGANT, which remains consistent with the paper.

TABLE 5.7. Comparison of fairness (measured with Δ_{SP}). Numbers are in percentage.

	German	Recidivism	Credit
SAGE	50.6 ± 15.9	9.36 ± 3.15	13.0 ± 4.01
E-SAGE	$\mathbf{16.3 \pm 10.9}$	$\mathbf{6.39 \pm 2.85}$	$\mathbf{8.94 \pm 0.99}$
GCN	37.4 ± 3.24	10.1 ± 3.01	11.1 ± 3.22
E-GCN	$\mathbf{3.52 \pm 3.77}$	$\mathbf{9.56 \pm 3.22}$	$\mathbf{7.28 \pm 1.46}$
JK	41.2 ± 18.1	10.1 ± 3.15	9.24 ± 0.60
E-JK	$\mathbf{22.4 \pm 1.95}$	$\mathbf{6.26 \pm 2.78}$	$\mathbf{3.37 \pm 2.64}$
[98]	14.8 ± 18.3	9.59 ± 0.65	3.84 ± 0.17
[207]	3.66 ± 0.52	8.04 ± 2.97	7.10 ± 5.10

In addition, we provide a detailed fairness comparison between ELEGANT and robust GNNs from [98] and [207] in Table 5.7. We observe that the best performances still come from the GNNs equipped with ELEGANT on all datasets. Hence we argue that ELEGANT exhibits satisfying performance in usability, which remains consistent with the discussion in the paper.

Why ELEGANT Improves Fairness? We note that improving fairness is a byproduct of ELEGANT, and our focus is to achieve certification over the fairness level of the prediction results. We now provide a detailed discussion about why fairness is improved here. First, existing works found that the distribution difference in the node attribute values and edge existence across different subgroups is a significant source of bias [47, 37, 63]. However, adding noise on both node attributes and graph topology may reduce such distributional divergence and mitigate bias. Second, As mentioned in Section 5.1.3.4, the proposed strategy to obtain the output predictions in ELEGANT is to select the fairest result among the output set $\hat{\mathcal{Y}}'$, where each output is derived based on a sample $\mathbf{\Gamma}'_{\boldsymbol{A}} \in \bar{\mathcal{A}}'$ (i.e., $\operatorname{argmin}_{\hat{\boldsymbol{Y}}'} \pi(\hat{\boldsymbol{Y}}', \mathcal{V}_{\text{tst}}) \;\; s.t.\; \hat{\boldsymbol{Y}}' \in \hat{\mathcal{Y}}'$). Such a strategy provides a large enough probability to achieve certification in light of Proposition 1. Meanwhile, we point out that such a strategy also helps to significantly improve fairness since highly biased outputs are excluded.

5.1.5 Related Work

Algorithmic Fairness in GNNs. Existing GNN works on fairness mainly focus on group fairness and individual fairness [50]. Specifically, group fairness requires that each demographic subgroup (divided by sensitive attributes such as gender and race) in the graph should have their fair share of interest based on predictions [134]. Adversarial training is among the most popular strategies [37, 50]. In addition, regularization [3, 63, 240], topology modification [47, 179], and orthogonal projection [148] are also commonly used strategies. On the other hand, individual fairness it requires that similar individuals should be treated similarly [58], where such similarity may be determined in different ways [104, 46]. Designing regularization terms to promote individual fairness for GNNs is a common strategy [63, 46, 177].

GNN Defense Against Attacks. Existing works on GNN defense are categorized into five mainstreams, namely adversarial training [220, 41, 198], graph data purification [61, 96, 206, 110], perturbation detection [223, 84, 97], and certified defense [171, 192, 10, 248, 89]. Among them, certified defense is the only approach that secures GNNs theoretically, such that

attackers cannot find any adversary to fool the GNNs [171, 192, 10, 248, 89]. Note that most certified defense approaches only secure the prediction for a specific data point (e.g., a node in node classification). Different from them, ELEGANT enables us to secure the fairness level for GNNs, which are affected by all predictions in the test set.

5.1.6 Conclusion

In this paper, we take initial steps to tackle a novel problem of certifying GNN node classifiers on fairness. To address this problem, we propose a principled framework, ELEGANT, which achieves certification on top of any optimized GNN node classifier associated with certain perturbation budgets, such that it is impossible for attackers to corrupt the fairness level of predictions within such budgets. Notably, ELEGANT is designed to serve as a plug-and-play framework for any optimized GNNs ready to be deployed and does not rely on any assumption over GNN structure or parameters. Extensive experiments verify the satisfying effectiveness of ELEGANT. In addition, we also found ELEGANT beneficial to GNN debiasing, and explored how its parameters influence the certification performance. We leave certifying the fairness level of GNNs over other learning tasks on graphs as future works.

5.2 A Flexible Framework of Certified Unlearning for Graph Neural Networks

5.2.1 Introduction

Graph-structured data is ubiquitous among various real-world applications, such as online social platform [74], finance system [193], and chemical discovery [85]. In recent years, Graph Neural Networks (GNNs) have exhibited promising performance in various graph-based downstream tasks [74, 245, 214, 222]. The success of GNNs is mainly attributed to its message-passing mechanism, which enables each node to take advantage of the information from its multi-hop neighbors [74, 111]. As a consequence, GNNs have been widely adopted in a plethora of realms [246, 45, 243, 64, 211].

Despite the success of GNNs, their widespread usage has also raised social concerns about the issue of privacy protection [209, 31, 244]. It is worth noting that, in practice, the graph data used for training may contain sensitive personal information of the involved individuals [209, 146, 204]. Once trained, these GNNs typically encode such personal information in the learnable parameters. As a consequence, privacy leakage may happen when the trained GNNs are deployed and exposed to potential attackers [146, 204]. For example, the similarity of the health records between patients could provide key information for disease diagnosis [239]. Therefore, GNNs can be trained on patient networks for disease prediction, where the connections between patients indicate high similarity scores of their health records. However, malicious attackers can easily reveal the patients' health records that are used for training via membership inference attack [146], which severely threatens privacy. Facing such a threat of privacy leakage, legislation such as the General Data Protection Regulation (GDPR) (GDPR 2016) [163], the California Consumer Privacy Act (CCPA) (CCPA 2018) [151], and the Personal Information Protection and Electronic Documents Act (PIPEDA 2000) [1] have emphasized the importance of *the right to be forgotten* [116]. Specifically, users should have the right to request the deletion of their personal information from those learning models that encode it. Such an urgent need poses challenges towards removing certain personal information from the trained GNNs.

The need for information removal from these trained models has led to the development of *machine unlearning* [14, 219]. Specifically, the ultimate goal of machine unlearning is to remove information regarding certain training data from a previously trained model. A straightforward approach is to perform model re-training. However, on the one hand, the model owner may not have full access to the training data; on the other hand, re-training can be prohibitively expensive even if training data is fully accessible [57]. To achieve more efficient information removal, a series of existing works [14, 20, 86] proposed to directly modify the parameters of the trained models. Nevertheless, most of these works only achieve unlearning empirically and fail to provide any theoretical guarantee. This problem has led to the emerging of certified unlearning [172, 70], which aims to develop unlearning approaches with theoretical guarantee on their effectiveness. In the domain of graph learning, a few recent works, such as [209, 31], have explored to achieve certified unlearning for GNNs. However, a major limitation of these approaches is their low flexibility. First, most approaches are designed to completely unlearn a given set of nodes or edges, while this may not comply

with certain unlearning needs in real-world applications. For example, on a social network platform, a user may decide to stop disclosing certain personal information to the GNN-based friend recommendation model but continue using the platform. In such a case, the attribute information of this user should then be partially removed from the GNN model, which protects the user's privacy and maintains algorithmic personalization as well. Therefore, it is desired to develop flexible certified unlearning approaches for GNNs to handle unlearning requests centered on node attributes. Second, existing certified unlearning approaches are mostly designed for a specific type of GNNs [31] or the GNNs trained following a specially designed objective function [209, 31]. However, various GNNs and objectives have been adopted for diverse real-world applications, and thus it is also desired to develop more flexible certified unlearning approaches for different GNNs trained with different objectives. Nevertheless, existing exploration in developing flexible and certified unlearning approaches for GNNs remains nascent.

In this paper, we study a novel and critical problem of developing a certifiable unlearning framework that can flexibly unlearn personal information in graphs and generalize across GNNs. We note that this is a non-trivial task. In essence, we mainly face three challenges. *(i) Characterizing node dependencies.* Different from tabular data, the nodes in graph data usually have dependencies with each other. Properly characterizing node dependencies thus becomes the first challenge to achieve unlearning for GNNs. *(ii) Achieving flexible unlearning.* Unlearning requests may be initiated towards nodes, node attributes (partial or full), and edges. Meanwhile, various GNNs have been adopted for different applications, and most of these GNNs have different model structures and optimization objectives. Therefore, achieving flexible unlearning for different types of unlearning requests, GNN structures, and objectives becomes the second challenge. *(iii) Obtaining certification for unlearning.* To reduce the risk of privacy leakage, it is critical for the model owner to ensure that the information needed to be removed has been completely wiped out before model deployment. However, GNNs may have complex structures, and it is difficult to examine whether certain sensitive personal information remains being encoded or not. Meanwhile, certified unlearning for GNNs usually requires strict conditions (e.g., assuming that GNNs are trained under a specially designed objective [209, 31]) and thus sacrifices flexibility. Properly certifying the effectiveness of unlearning is our third challenge.

Our Contributions. We propose IDEA (flexIble anD cErtified unleArning), which is a flexible framework of certified unlearning for GNNs. Specifically, to tackle the first two challenges, we propose to model the intermediate state between the optimization objectives with and without the instances (e.g., nodes, edges, and attributes) to be unlearned. Meanwhile, four different types of common unlearning requests are instantiated, and GNN parameters after unlearning can be efficiently approximated with flexible unlearning request specifications. To tackle the third challenge, we propose a novel theoretical certification on the unlearning effectiveness of IDEA. We show that our certification method brings an empirically tighter bound on the distance between the approximated and actual GNN parameters compared to other existing alternatives. We summarize our contributions as: **(1) Problem Formulation.** We formulate and make an initial investigation on a novel research problem of flexible and certified unlearning for GNNs. **(2) Algorithm Design.** We propose IDEA, a flexible framework of certified unlearning for GNNs without relying on any specific GNN structures or any specially designed objective functions, which shows significant value for practical use. **(3)**

Experimental Evaluation. We conduct comprehensive experiments on real-world datasets to verify the superiority of IDEA over existing alternatives in multiple key perspectives, including bound tightness, unlearning efficiency, model utility, and unlearning effectiveness.

5.2.2 Preliminaries

5.2.2.1 Notations

We use bold uppercase letters (e.g., $\boldsymbol{A}$), bold lowercase letters (e.g., $\boldsymbol{x}$), and normal lowercase letters (e.g., n) to denote matrices, vectors, and scalars, respectively. We represent an attributed graph as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}\}$. Here $\mathcal{V} = \{v_1, ..., v_n\}$ denotes the set of nodes, where n is the total number of nodes. $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ represents the set of edges. $\mathcal{X} = \{x_{1,1}, ..., x_{n,c}\}$ is the set of node attribute values, where c is the total number of node attribute dimensions, and $x_{i,j}$ represents the attribute value of node v_i at the j-th attribute dimension ($1 \leq i \leq n, 1 \leq j \leq c$). We utilize $f_{\boldsymbol{\theta}}$ to represent a GNN model parameterized by the learnable parameters in $\boldsymbol{\theta}$.

In this paper, we focus on the commonly studied node classification task, which widely exists in real-world applications. Specifically, we are given the labels of a set of training nodes $\mathcal{V}_{\text{trn}}$ ($\mathcal{V}_{\text{trn}} \subset \mathcal{V}$) as $\mathcal{Y}_{\text{trn}}$. Here $\mathcal{Y}_{\text{trn}} = \{Y_1, ..., Y_m\}$, where $Y_i \in \{1, ..., c\}$ ($1 \leq i \leq m$) is the node label of v_i; c is the total number of possible classes; and m represents the number of training nodes, i.e., $m = |\mathcal{V}_{\text{trn}}|$. Our goal here is to optimize the parameter $\boldsymbol{\theta}$ of the GNN model f with k message-passing layers as $\boldsymbol{\theta}^*$ w.r.t. certain objective function over $\mathcal{V}_{\text{trn}}$, such that $f_{\boldsymbol{\theta}^*}$ is able to achieve accurate predictions for the nodes in the test set $\mathcal{V}_{\text{tst}}$ ($\mathcal{V}_{\text{tst}} \cap \mathcal{V}_{\text{trn}} = \varnothing$).

5.2.2.2 Problem Statement

In this subsection, we formally present the problem formulation of *Flexible and Certified Unlearning for GNNs*. We first elaborate on the mathematical formulation of certified unlearning for GNNs. Specifically, certified unlearning requires that the unlearning strategy have a theoretical guarantee of unlearning effectiveness. We adopt a commonly used criterion for the effectiveness of unlearning, i.e., $(\varepsilon - \delta)$ *Certified Unlearning*. Here ε and δ are two parameters controlling the relaxation of such a criterion. We present the definition of $(\varepsilon - \delta)$ certified unlearning for GNNs below.

DEFINITION 5.2.1. $(\varepsilon - \delta)$ ***Certified Unlearning for GNNs.*** *Let $\mathcal{H}$ be the hypothesis space of a GNN model parameters and $\mathcal{A}$ be the associated optimization process. Given a graph $\mathcal{G}$ for GNN optimization and a $\Delta\mathcal{G}$ that characterizes the information to be unlearned, $\mathcal{U}$ is an $(\varepsilon - \delta)$ certified unlearning process iff $\forall\ \mathcal{T} \subseteq \mathcal{H}$, we have*

$$\Pr\left(\mathcal{U}\left(\mathcal{G}, \Delta\mathcal{G}, \mathcal{A}\left(\mathcal{G}\right)\right) \in \mathcal{T}\right) \leq e^{\varepsilon} \Pr\left(\mathcal{A}\left(\mathcal{G} \ominus \Delta\mathcal{G}\right) \in \mathcal{T}\right) + \delta, \text{ and}$$
$$\Pr\left(\mathcal{A}\left(\mathcal{G} \ominus \Delta\mathcal{G}\right) \in \mathcal{T}\right) \leq e^{\varepsilon} \Pr\left(\mathcal{U}\left(\mathcal{G}, \Delta\mathcal{G}, \mathcal{A}\left(\mathcal{G}\right)\right) \in \mathcal{T}\right) + \delta,$$

where $\mathcal{G} \ominus \Delta\mathcal{G}$ represents the graph data with $\Delta\mathcal{G}$ being removed.

The intuition of Definition 5.2.1 is that, once the two inequalities above are satisfied, the difference between the distribution of the unlearned GNN parameters and that of the re-trained GNN parameters over $\mathcal{G} \ominus \Delta\mathcal{G}$ is bounded by a small threshold ε and relaxed by a probability δ. We note that, different from most existing literature on GNN unlearning, the information to be unlearned does not necessarily come from a node or an edge in Definition 5.2.1. Such an

extension paves the way towards more flexible certified unlearning for GNNs. We formally present the problem formulation of *Flexible and Certified Unlearning for GNNs* below.

PROBLEM 5.2.1. ***Flexible and Certified Unlearning for GNNs.*** *Given a GNN model $f_{\boldsymbol{\theta}^*}$ optimized over $\mathcal{G}$ and any request to unlearn information characterized by $\Delta\mathcal{G}$, our goal is to achieve $(\varepsilon - \delta)$ certified unlearning over $f_{\boldsymbol{\theta}^*}$.*

5.2.3 Unlearning Request Instantiations

We instantiate the unlearning requests characterized by $\Delta\mathcal{G}$, namely *Node Unlearning Request*, *Edge Unlearning Request*, and *Attribute Unlearning Request*. We present an illustration in Fig. 5.6.

Node Unlearning Request. The most common unlearning request in GNN applications is to unlearn a given set of nodes. For example, in a social network platform, a GNN model can be trained on the friendship network formed by the platform users to perform friendship recommendation. When a user has decided to quit such a platform and withdrawn the consent of using her private data, this user may request to unlearn the node associated with her from the social network. In such a case, the information to be unlearned is characterized by $\Delta\mathcal{G} = \{\Delta\mathcal{V}, \kappa_e(\Delta\mathcal{V}), \kappa_x(\Delta\mathcal{V})\}$. Here κ_e and κ_x return the set of the direct edges and node attributes associated with nodes in $\Delta\mathcal{V}$, respectively.

Edge Unlearning Request. In addition to the information encoded by the nodes, edges can also encode critical private information and may need to be unlearned as well. In fact, it has been empirically proved that malicious attackers can easily infer the edges used for training, which directly threatens privacy [79]. In such a case, the information to be unlearned is characterized by $\Delta\mathcal{G} = \{\varnothing, \Delta\mathcal{E}, \varnothing\}$.

Attribute Unlearning Request. Both requests above fail to represent cases where only node attributes are requested to be unlearned. Here we show two common node attribute unlearning requests. *(1) Full Attribute Unlearning.* In this case, all information regarding the attributes of a set of nodes is requested to be unlearned. For example, a social network platform user may withdraw the consent for the GNN-based friend recommendation algorithm to encode any of its attributes during training. In such a case, the information to be unlearned is characterized by $\Delta\mathcal{G} = \{\varnothing, \varnothing, \Delta\mathcal{X}\}$, where for node v_i, if $x_{i,j} \in \Delta\mathcal{X}$, then $\forall j \in \{1, ..., c\}, x_{i,j} \in \Delta\mathcal{X}$. *(2) Partial Attribute Unlearning.* The attributes of a node may also be requested to be partially unlearned. For example, in a social network, a user may withdraw the consent of using the information regarding certain attribute(s) due to various reasons, e.g., feeling being unfairly treated. However, this user may still continue using such a platform, and thus other attributes should not be unlearned to ensure satisfying personalized service quality. In such a case, the information to be unlearned is characterized by $\Delta\mathcal{G} = \{\varnothing, \varnothing, \Delta\mathcal{X}\}$, where for node v_i, if $x_{i,j} \in \Delta\mathcal{X}$, then $\exists j \in \{1, ..., c\}, x_{i,j} \notin \Delta\mathcal{X}$. Note that the two types of attribute unlearning can be requested together. Hence, we utilize $\Delta\mathcal{X}$ to characterize a mixture of both types of attributes to be unlearned.

Based on the instantiations above, we denote $\Delta\mathcal{G} = \{\Delta\mathcal{V}, \Delta\mathcal{E} \cup \kappa_e(\Delta\mathcal{V}), \Delta\mathcal{X} \cup \kappa_x(\Delta\mathcal{V})\}$ as a potential combination of all types of unlearning requests. Accordingly, we formally define $\mathcal{G} \ominus \Delta\mathcal{G} = \{\mathcal{V}\backslash\Delta\mathcal{V}, \mathcal{E}\backslash\Delta\mathcal{E}\backslash\kappa_e(\Delta\mathcal{V}), \mathcal{X}\backslash\Delta\mathcal{X}\backslash\kappa_x(\Delta\mathcal{V})\}$.

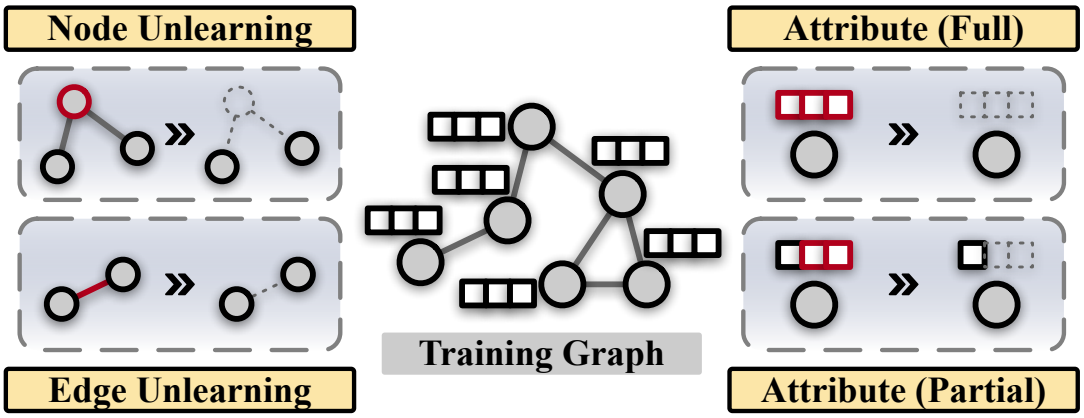


FIGURE 5.6. An illustration of common unlearning requests.

5.2.4 Methodology

In this section, we present our proposed framework IDEA, which aims to achieve flexible and certified unlearning for GNNs. We first present the general formulation of flexible unlearning for GNNs. Then, we introduce a unified modeling integrating different instantiations of unlearning requests. We finally propose a novel theoretical guarantee on the effectiveness of IDEA as the certification.

5.2.4.1 Flexible Unlearning for GNNs

We first present a unified formulation of flexible unlearning for GNNs. In general, our rationale here is to design a framework to directly approximate the change in the (optimal) learnable parameter $\boldsymbol{\theta}^*$ during unlearning. Specifically, we first review the training process of a given GNN model f over graph data $\mathcal{G}$. Then, we consider the training objective with information of $\Delta\mathcal{G}$ being removed as a perturbed training objective over $\mathcal{G}$. We are now able to analyze how the optimal learnable parameter $\boldsymbol{\theta}^*$ would change when the objective function is modified. Note that we adopt a generalized formulation of such modification over the objective function, such that our analysis can be adapted to different unlearning requests.

In a typical training process of a given GNN model f over graph data $\mathcal{G}$, the optimal learnable parameter $\boldsymbol{\theta}^*$ is obtained via solving the optimization problem of

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{m} \sum_{v_i \in \mathcal{V}_{\text{trn}}} \mathscr{L}\left(\boldsymbol{\theta}, v_i, \mathcal{G}\right), \tag{5.2}$$

where a typical choice of $\mathscr{L}$ is cross-entropy loss in node classification tasks. Here we consider that the computation of $\mathscr{L}$ also relies on other necessary information such as $\hat{Y}_i$ by default and omit them for simplicity. As a comparison, the optimal learnable parameter trained over $\mathcal{G} \ominus \Delta\mathcal{G}$, which we denoted as $\tilde{\boldsymbol{\theta}}^*$, is obtained via solving the problem of

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{m - |\Delta\mathcal{V}|} \sum_{v_i \in \mathcal{V}_{\text{trn}} \setminus \Delta\mathcal{V}} \mathscr{L}\left(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G}\right). \tag{5.3}$$

To study how the optimal parameters change when transforming from Eq. (5.2) to Eq. (5.3), it is necessary to analyze how the objective function and optimal solution change between the two cases. To systematically compare Eq. (5.2) and Eq. (5.3), here we define $\phi_k(\cdot)$

as a function that takes a node and a graph as its input and outputs the set of nodes in the computation graph of the input node (excluding the input node itself). Here a computation graph is a subgraph centered on a given node with neighbors up to k hops away, where k is the layer number of the studied GNN. We have the following proposition.

PROPOSITION 5.2.1. ***Localized Equivalence of Training Nodes.*** *Given $\Delta\mathcal{G} = \{\Delta\mathcal{V}, \Delta\mathcal{E}, \Delta\mathcal{X}\}$ to be unlearned and an objective $\mathscr{L}$ computed over $f_{\boldsymbol{\theta}}$, $\mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}) = \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G})$ holds $\forall v_i \notin \phi_k(v_j) \cup \{v_j\}, v_j \in \Delta\mathcal{V} \cup \gamma_e(\Delta\mathcal{E}) \cup \gamma_x(\Delta\mathcal{X})$. Here γ_e and γ_x return the set of nodes that directly connect to the edges in $\mathcal{E}$ and that have attribute(s) in $\mathcal{X}$, respectively.*

The intuition of Proposition 5.2.1 is that, under a given $f_{\boldsymbol{\theta}}$, the value of $\mathscr{L}$ maintains the same between Eq. (5.2) and Eq. (5.3) for those training nodes that are not topologically close to the instances (i.e., nodes, attributes, and edges) in $\Delta\mathcal{G}$. To bridge Eq. (5.2) and Eq. (5.3), we then propose a principled formulation to characterize their intermediate state. Specifically, we add an additional term over Eq. (5.3) by defining $\boldsymbol{\theta}^*_{\Delta\mathcal{G},\xi}$ with

$$\boldsymbol{\theta}^*_{\Delta\mathcal{G},\xi} \coloneqq \arg\min_{\boldsymbol{\theta}} \frac{1}{m} \sum_{v_i \in \mathcal{V}_{\text{trn}}} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}) + \xi \left(\mathscr{L}_{\text{add}} - \mathscr{L}_{\text{sub}}\right). \tag{5.4}$$

We then introduce the modeling of $\mathscr{L}_{\text{add}}$ and $\mathscr{L}_{\text{sub}}$. Specifically, we formulate $\mathscr{L}_{\text{add}}$ with

$$\begin{aligned}\mathscr{L}_{\text{add}} = \alpha_1 \sum_{v_i \in \mathcal{V}_1} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G}) + \alpha_2 \sum_{v_i \in \mathcal{V}_2} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G}) \\ + \alpha_3 \sum_{v_i \in \mathcal{V}_3} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G}) + \alpha_4 \sum_{v_i \in \mathcal{V}_4} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G} \ominus \Delta\mathcal{G}).\end{aligned} \tag{5.5}$$

Here $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in \{0, 1\}$ are used to flag whether the requests of node unlearning, full attribute unlearning, partial node attribute unlearning, and edge unlearning exist or not, respectively. We now introduce $\mathcal{V}_1$, $\mathcal{V}_2$, $\mathcal{V}_3$, and $\mathcal{V}_4$. Specifically, $\mathcal{V}_1$ represents the set of training nodes whose computation graph includes those nodes to be unlearned. We denote the sets of nodes associated with $\Delta\mathcal{X}$ when their unlearned attributes are replaced with any non-informative numbers (e.g., 0) as $\mathcal{V}_x^{(\text{Full})}$ and $\mathcal{V}_x^{(\text{Partial})}$ for full and partial attribute unlearning, respectively. $\mathcal{V}_2$ and $\mathcal{V}_3$ include training nodes whose computation graph includes attributes to be unlearned fully and partially plus the nodes in $\mathcal{V}_x^{(\text{Full})}$ and $\mathcal{V}_x^{(\text{Partial})}$, respectively; $\mathcal{V}_4$ is the set of nodes whose computation graph includes those edges to be unlearned. Mathematically, we formulate $\mathcal{V}_1$, $\mathcal{V}_2$, $\mathcal{V}_3$, and $\mathcal{V}_4$ as

$$\mathcal{V}_1 = \cup_{v_i \in \Delta\mathcal{V}} \left(\phi_k(v_i) \cap \mathcal{V}_{\text{trn}}\right), \tag{5.6}$$

$$\mathcal{V}_2 = \mathcal{V}_x^{(\text{Full})} \cup \{v_i : v_i \in \phi_k(v_j) \cap \mathcal{V}_{\text{trn}},\ v_j \in \mathcal{V}_x^{(\text{Full})}\}, \tag{5.7}$$

$$\mathcal{V}_3 = \mathcal{V}_x^{(\text{Partial})} \cup \{v_i : v_i \in \phi_k(v_j) \cap \mathcal{V}_{\text{trn}},\ v_j \in \mathcal{V}_x^{(\text{Partial})}\}, \tag{5.8}$$

$$\mathcal{V}_4 = \cup_{v_i \in \gamma_e(\Delta\mathcal{E})} \left(\phi_k(v_i) \cap \mathcal{V}_{\text{trn}}\right) \tag{5.9}$$

We then formulate $\mathscr{L}_{\text{sub}}$ as

$$\begin{aligned}\mathscr{L}_{\text{sub}} = \alpha_1 \sum_{v_i \in \tilde{\mathcal{V}}_1} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}) + \alpha_2 \sum_{v_i \in \tilde{\mathcal{V}}_2} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}) \\ + \alpha_3 \sum_{v_i \in \tilde{\mathcal{V}}_3} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}) + \alpha_4 \sum_{v_i \in \tilde{\mathcal{V}}_4} \mathscr{L}(\boldsymbol{\theta}, v_i, \mathcal{G}),\end{aligned} \tag{5.10}$$

where $\tilde{\mathcal{V}}_1$ includes all nodes in $\Delta\mathcal{V}$ and the training nodes within k hops away from the nodes in $\Delta\mathcal{V}$; We denote the sets of nodes associated with $\Delta\mathcal{X}$ with their vanilla attributes as $\tilde{\mathcal{V}}_x^{(\text{Full})}$ and $\tilde{\mathcal{V}}_x^{(\text{Partial})}$ for full and partial attribute unlearning, respectively. $\tilde{\mathcal{V}}_2$ and $\tilde{\mathcal{V}}_3$ include training nodes whose computation graph includes attributes to be unlearned fully and partially plus the nodes in $\tilde{\mathcal{V}}_x^{(\text{Full})}$ and $\tilde{\mathcal{V}}_x^{(\text{Partial})}$, respectively; $\tilde{\mathcal{V}}_4$ is the set of nodes whose computation graph includes those edges to be unlearned, i.e., $\tilde{\mathcal{V}}_4 = \mathcal{V}_4$. Mathematically, we have

$$\tilde{\mathcal{V}}_1 = \cup_{v_i \in \Delta\mathcal{V}} \left(\phi_k(v_i) \cap \mathcal{V}_{\text{trn}}\right) \cup \Delta\mathcal{V}, \tag{5.11}$$

$$\tilde{\mathcal{V}}_2 = \tilde{\mathcal{V}}_x^{(\text{Full})} \cup \{v_i : v_i \in \phi_k(v_j) \cap \mathcal{V}_{\text{trn}},\ v_j \in \tilde{\mathcal{V}}_x^{(\text{Full})}\}, \tag{5.12}$$

$$\tilde{\mathcal{V}}_3 = \tilde{\mathcal{V}}_x^{(\text{Partial})} \cup \{v_i : v_i \in \phi_k(v_j) \cap \mathcal{V}_{\text{trn}},\ v_j \in \tilde{\mathcal{V}}_x^{(\text{Partial})}\}, \tag{5.13}$$

$$\tilde{\mathcal{V}}_4 = \mathcal{V}_4. \tag{5.14}$$

We then have the complete formulation of Eq. (5.4) given Eq. (5.5) to (5.14). Based on the modeling above, we have the optimal equivalence between Eq. (5.4) and Eq. (5.3) below.

LEMMA 3. ***Optimal Equivalence.*** *The optimal solution to Eq. (5.4) (denoted as $\boldsymbol{\theta}^*_{\Delta\mathcal{G},\xi}$) equals to the optimal solution to Eq. (5.3) (denoted as $\tilde{\boldsymbol{\theta}}^*$) when $\xi = \frac{1}{m}$.*

Now we have successfully bridged the gap between Eq. (5.2) and Eq. (5.3) by modeling their intermediate states with Eq. (5.4). More importantly, Lemma 3 paves the way towards directly approximating $\tilde{\boldsymbol{\theta}}^*$ based on $\boldsymbol{\theta}^*$ by giving Theorem 5.2.1 below.

THEOREM 5.2.1. ***Approximation with Infinitesimal Residual.*** *Given a graph data $\mathcal{G}$, $\Delta\mathcal{G} = \{\Delta\mathcal{V}, \Delta\mathcal{E}, \Delta\mathcal{X}\}$ to be unlearned, and an objective $\mathscr{L}$ computed over an $f_{\boldsymbol{\theta}^*}$, using $\boldsymbol{\theta}^* + \frac{1}{m}\Delta\bar{\boldsymbol{\theta}}^*$ as an approximation of $\tilde{\boldsymbol{\theta}}^*$ only brings a first-order infinitesimal residual w.r.t. $\|\boldsymbol{\theta}^* - \tilde{\boldsymbol{\theta}}^*\|_2$, where $\Delta\bar{\boldsymbol{\theta}}^* = -\boldsymbol{H}_{\boldsymbol{\theta}^*}^{-1}\left(\nabla_{\boldsymbol{\theta}}\mathscr{L}_{add} - \nabla_{\boldsymbol{\theta}}\mathscr{L}_{sub}\right)$, and $\boldsymbol{H}_{\boldsymbol{\theta}^*} := \nabla^2_{\boldsymbol{\theta}}\frac{1}{m}\sum_{v_i \in \mathcal{V}_{trn}} \mathscr{L}\left(\boldsymbol{\theta}, v_i, \mathcal{G}\right)$.*

We note that the approximation strategy above relies on the assumption that $\forall \mathcal{V}_i \cap \mathcal{V}_j = \varnothing$ and $\forall \tilde{\mathcal{V}}_i \cap \tilde{\mathcal{V}}_j = \varnothing$ for $i, j \in \{1, 2, 3, 4\}$ when $i \neq j$. However, it can also handle cases where such an assumption does not hold. We show this in Proposition 5.2.2.

PROPOSITION 5.2.2. ***Serializability of Approximation.*** *Any mixture of unlearning request instantiations can be split into multiple sets of unlearning requests, where each set of unlearning requests satisfies $\forall \mathcal{V}_i \cap \mathcal{V}_j = \varnothing$ and $\forall \tilde{\mathcal{V}}_i \cap \tilde{\mathcal{V}}_j = \varnothing$ for $i, j \in \{1, 2, 3, 4\}$ when $i \neq j$. Serially performing approximation following these request sets achieves upper-bounded error.*

Unlearning in Practice. The approximation approach given by Theorem 5.2.1 requires computing the inverse matrix of the Hessian matrix, which usually leads to high computational costs. Here we propose to utilize the stochastic estimation method [34] to perform estimation based on an iterative approach, which reduces the time complexity to $O(tp)$. Here t is the total number of iterations adopted by the stochastic estimation method, and p represents the total number of learnable parameters in $\boldsymbol{\theta}$.

5.2.4.2 Unlearning Certification

In this subsection, we introduce a novel certification based on Theorem 5.2.1. According to the unlearning process given by Definition 5.2.1, our goal is to achieve guaranteed closeness between $\tilde{\boldsymbol{\theta}}^*$ (i.e., the ideal unlearned parameter derived from Eq. (5.3)) and the approximation

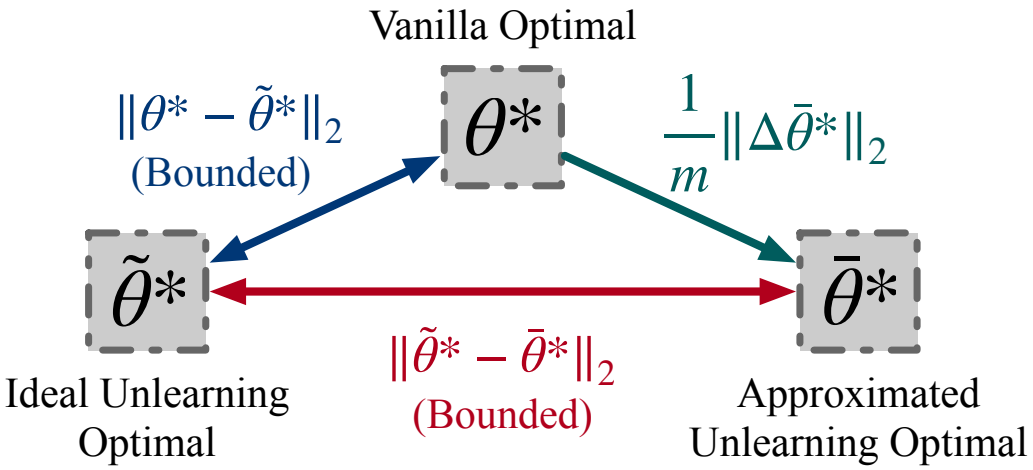


FIGURE 5.7. Distances between $\boldsymbol{\theta}^*$, $\tilde{\boldsymbol{\theta}}^*$, and $\bar{\boldsymbol{\theta}}^*$. Here, $\boldsymbol{\theta}^*$ denotes the optimal parameter before unlearning; $\tilde{\boldsymbol{\theta}}^*$ is the ideal optimal parameter after unlearning, which is obtained via re-training; $\bar{\boldsymbol{\theta}}^*$ is an approximation of $\tilde{\boldsymbol{\theta}}^*$ give by Theorem 5.2.1.

of such a parameter (denoted as $\bar{\boldsymbol{\theta}}^*$). Then we are able to achieve certifiable unlearning effectiveness.

Although certified unlearning for GNNs is studied by some recent explorations [209, 31], these approaches can only be applied when the studied GNN model is trained following a specially modified objective. In particular, such a modification requires adding an additional regularization term of $\boldsymbol{\theta}$ scaled by a random vector onto the objective, which is specially designed for certification purposes. However, most GNNs are optimized following common objectives (e.g., cross-entropy loss) instead of such a modified objective. Therefore, these certified unlearning approaches cannot be flexibly used across different GNNs in real-world applications. Here we aim to develop a certified unlearning approach based on Theorem 5.2.1, such that it is not tailored for any optimization objective and thus can be easily generalized across various GNNs. Towards this goal, we first review the ℓ_2 distances between $\boldsymbol{\theta}^*$, $\tilde{\boldsymbol{\theta}}^*$, and $\bar{\boldsymbol{\theta}}^*$. We present an illustration in Fig. 5.7. It is difficult to directly analyze the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. We thus start by analyzing the ℓ_2 distance between $\boldsymbol{\theta}^*$ and $\tilde{\boldsymbol{\theta}}^*$. We found that the ℓ_2 distance between $\boldsymbol{\theta}^*$ and $\tilde{\boldsymbol{\theta}}^*$ is upper bounded under common assumptions, which are widely adopted in other existing works tackling unlearning problems [209, 31, 70]. We first present these assumptions below.

ASSUMPTION 1. *For the training objective of a given GNN model, we have: (1) The loss values of optimal points are bounded: $|\mathscr{L}(\boldsymbol{\theta}^*)| \leq C$ and $|\mathscr{L}(\tilde{\boldsymbol{\theta}}^*)| \leq C$; (2) The loss function $\mathscr{L}$ is L-Lipschitz continuous; (3) The loss function $\mathscr{L}$ is λ-strongly convex.*

Based on Assumption 1, we now present the bound between $\boldsymbol{\theta}^*$ and $\tilde{\boldsymbol{\theta}}^*$ in Theorem 5.2.2.

THEOREM 5.2.2. ***Distance Bound in Optimals.*** *The ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\boldsymbol{\theta}^*$ is given by*

$$\|\tilde{\boldsymbol{\theta}}^* - \boldsymbol{\theta}^*\|_2 \leq \frac{L|\Delta\mathcal{V}| + \sqrt{4m\lambda C|\tilde{\mathcal{V}}| + L^2|\Delta\mathcal{V}|^2}}{m\lambda}. \tag{5.15}$$

Denote $\mathcal{V}_x^{(F+P)} = \mathcal{V}_x^{(Full)} \cup \mathcal{V}_x^{(Partial)}$*, and* $\tilde{\mathcal{V}}$ *is given by*

$$\tilde{\mathcal{V}} = \mathcal{V}_1 \cup \mathcal{V}_4 \cup \{v_i : v_i \in \phi_k(v_j) \cap \mathcal{V}_{trn},\ v_j \in \mathcal{V}_x^{(F+P)}\}. \tag{5.16}$$

Here the rationale of $\tilde{\mathcal{V}}$ is to describe the set of nodes whose computation graphs involve any instance (i.e., nodes, attributes, and edges) to be unlearned. Noticing the relationship between $\boldsymbol{\theta}^*$, $\tilde{\boldsymbol{\theta}}^*$, and $\bar{\boldsymbol{\theta}}^*$ give by Fig. 5.7, we further show the bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$ in Proposition 5.2.3.

PROPOSITION 5.2.3. ***Distance Bound in Approximation.*** *The* ℓ_2 *distance bound between* $\tilde{\boldsymbol{\theta}}^*$ *and* $\bar{\boldsymbol{\theta}}^*$ *is given by*

$$\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2 \leq \frac{\lambda\|\Delta\bar{\boldsymbol{\theta}}^*\|_2 + L|\Delta\mathcal{V}| + \sqrt{4m\lambda C|\tilde{\mathcal{V}}| + L^2|\Delta\mathcal{V}|^2}}{m\lambda}. \tag{5.17}$$

The rationale of Proposition 5.2.3 is to characterize the maximum ℓ_2 distance between the ideal unlearning optimal and the approximation of unlearning optimal given by Theorem 5.2.1. Finally, based on Proposition 5.2.3, we are able to present the certification in Theorem 5.2.3.

THEOREM 5.2.3. *Let* $\boldsymbol{\theta}^* = \mathcal{A}(\mathcal{G})$ *be the empirical minimizer over* $\mathcal{G}$*,* $\tilde{\boldsymbol{\theta}}^* = \mathcal{A}(\mathcal{G} \ominus \Delta\mathcal{G})$ *be the empirical minimizer over* $\mathcal{G} \ominus \Delta\mathcal{G}$ *and* $\bar{\boldsymbol{\theta}}^*$ *be an approximation of* $\tilde{\boldsymbol{\theta}}^*$*. Define* ζ *as an upper bound of* $\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2$*. We have* $\mathcal{U}(\mathcal{G}, \Delta\mathcal{G}, \mathcal{A}(\mathcal{G})) = \bar{\boldsymbol{\theta}}^* + \boldsymbol{b}$ *is an* $(\varepsilon - \delta)$ *certified unlearning process, where* $\boldsymbol{b} \sim \mathcal{N}(0, \sigma^2\boldsymbol{I})$ *and* $\sigma \geq \frac{\zeta}{\varepsilon}\sqrt{2\ln(1.25/\delta)}$*.*

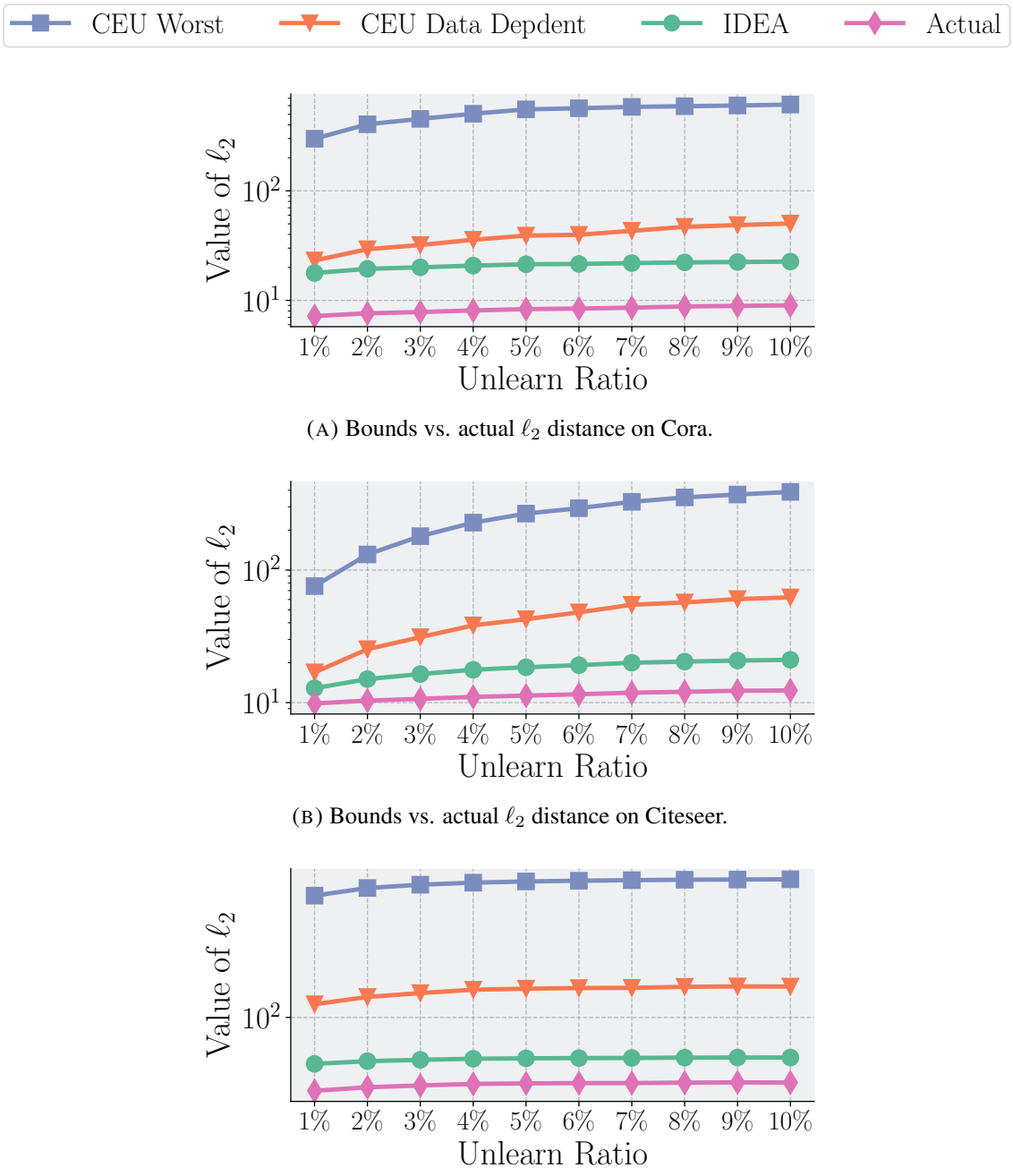
Therefore, according to Theorem 5.2.3, we are able to achieve certified unlearning by adding zero-mean Gaussian noise over the approximation derived from Theorem 5.2.1.

5.2.5 Experimental Evaluations

We empirically evaluate the performance of IDEA in this section. In particular, we aim to answer the following research questions. **RQ1**: How tight can IDEA bound the ℓ_2 distance between the ideal optimal $\tilde{\boldsymbol{\theta}}^*$ and the approximation $\bar{\boldsymbol{\theta}}^*$? **RQ2**: How well can IDEA improve the efficiency of unlearning compared with re-training and other alternatives? **RQ3**: How well can IDEA maintain the utility of the original GNN model? **RQ4**: How well can IDEA unlearn the information requested to be removed from the GNN?

5.2.5.1 Experimental Setup

Downstream Task and Datasets. We adopt the widely studied node classification task as the downstream task, which accounts for a wide range of real-world applications based on GNNs. We perform experiments over five real-world datasets, including Cora [111], Citeseer [111], PubMed [111], Coauthor-CS [174, 26], and Coauthor-Physics [174, 26]. These datasets usually serve as commonly used benchmark datasets for GNN performance over node classification tasks. Specifically, Cora, Citeseer, and PubMed are citation networks, where nodes denote research publications and edges represent the citation relationship between any pair of publications. The node attributes are bag-of-words representations of the publication keywords. Coauthor-CS, and Coauthor-Physics are two coauthor networks, where nodes



(A) Bounds vs. actual ℓ_2 distance on Cora.

(B) Bounds vs. actual ℓ_2 distance on Citeseer.

(C) Bounds vs. actual ℓ_2 distance on PubMed.

FIGURE 5.8. Bounds and actual value of the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$, i.e., $\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2$, over Cora, CiteSeer and PubMed datasets. *CEU Worst*, *CEU Data Dependent*, *IDEA*, and *Actual* represent the worst bound based on CEU, the data-dependent bound based on CEU, the bound based on IDEA, and the actual value of $\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2$ derived from re-training, respectively.

represent authors and edges denote the collaboration relationship between any pair of authors. We leave more dataset details, e.g., their statistics, in the online version[2].

[2]See online version here: https://openreview.net/pdf?id=C7nFUdAdXR

Backbone GNNs. To evaluate the generalization ability of IDEA across different GNNs, we propose to utilize two types of GNNs, including linear and non-linear GNNs. In terms of linear GNNs, we adopt the popular SGC [205]; in terms of non-linear GNNs, we adopt three popular ones, including GCN [111], GAT [188], and GIN [222].

Unlearning Requests. We consider all unlearning requests presented in Section 5.2.3. For each type of request, we perform experiments over a wide range of scales in terms of the number of unlearned instances (e.g., nodes and edges). For experiments with fixed ratios, we adopt a ratio of 5% to perform unlearning for nodes or edges unless otherwise specified.

Threat Models. To evaluate the effectiveness of the unlearning strategy, we propose to adopt different types of threat models. Although IDEA is able to flexibly perform four different types of unlearning requests, there are only limited threat models can be chosen from. In our experiments, we adopt two state-of-the-art threat models, namely MIA-Graph [146] and StealLink [79], for node membership inference attack and link stealing attack, respectively.

Baselines. We adopt five types of baselines for performance comparison. *(1) Re-Training.* We adopt the re-training approach to obtain an ideal model based on the optimization problem given by Eq. (5.3). *(2) Exact Unlearning.* We adopt the popular GraphEraser [26] as a representative method for exact unlearning. Specifically, exact unlearning methods aim to achieve the exact same probability distribution in the model space (after unlearning) compared with the re-trained model. As a comparison, IDEA aims to approximate the distribution of the re-trained model through unlearning. *(3) Certified Unlearning.* Finally, we adopt two representative approaches for certified unlearning, namely Certified Graph Unlearning (CGU) [30] and Certified Edge Unlearning (CEU) [209]. CGU is able to unlearn nodes, attributes, and edges. However, it is only applicable for the SGC model. As a comparison, CEU can be adapted to different GNNs. Nevertheless, it is specially designed for edge unlearning.

Evaluation Metrics. We evaluate IDEA with different metrics to answer the four research questions. *(1) Bound Tightness.* We propose to compare the numerical values of the bounds given by IDEA, the bounds given by other baselines, and the actual ℓ_2 distance of model parameters yielded by re-training. A smaller bound on the ℓ_2 distance indicates better tightness. *(2) Model Utility.* We utilize the F1 score to measure the model utility after unlearning. A higher F1 score indicates better performance. *(3) Unlearning Efficiency.* We utilize the running time (in seconds) that the unlearning methods take to measure efficiency, and a shorter running time indicates better efficiency. *(4) Unlearning Effectiveness.* We use the attack successful rate after unlearning to measure unlearning effectiveness. Lower attack successful rates indicate better effectiveness.

5.2.5.2 Evaluation of Bound Tightness

To answer **RQ1**, we first evaluate how tight the derived bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$ can be across different GNNs, graph datasets, and unlearning ratios. We also compare the bound derived based on IDEA and other bounds in existing works. To the best of our knowledge, CEU [209] is the only existing certified unlearning approach that provides generalizable bounds across different GNNs. In particular, CEU provides bounds over the objective function after unlearning, and we adapt such bounds over the objective function to ℓ_2 distance

TABLE 5.8. F1 score on five real-world graph datasets under node classification task. All numerical values are reported in percentage, and the F1 scores given by the proposed framework IDEA are marked in bold.

		Cora	CiteSeer	PubMed	CS	Physics
GCN	**Re-Training**	76.88 ± 0.3	67.27 ± 0.6	76.20 ± 0.0	86.79 ± 0.3	92.30 ± 0.0
	Random	47.97 ± 0.5	46.25 ± 5.6	70.98 ± 0.1	80.64 ± 0.3	75.23 ± 0.1
	BEKM	50.68 ± 2.0	46.85 ± 4.9	69.64 ± 0.1	80.30 ± 0.2	74.85 ± 0.1
	BLPA	43.79 ± 2.2	40.24 ± 8.3	63.42 ± 5.7	85.10 ± 0.3	78.93 ± 0.5
	IDEA	$\mathbf{72.08 \pm 1.2}$	$\mathbf{61.56 \pm 1.2}$	$\mathbf{73.11 \pm 0.0}$	$\mathbf{86.13 \pm 0.4}$	$\mathbf{91.93 \pm 0.1}$
SGC	**Re-Training**	76.14 ± 0.6	65.77 ± 0.0	75.90 ± 0.0	87.10 ± 0.1	92.01 ± 0.0
	Random	46.00 ± 0.7	45.25 ± 3.0	69.03 ± 0.1	81.30 ± 0.3	80.81 ± 0.1
	BEKM	48.83 ± 1.8	46.45 ± 0.4	69.76 ± 0.1	80.08 ± 0.2	74.87 ± 0.2
	BLPA	63.59 ± 1.4	39.44 ± 2.8	62.98 ± 4.6	86.95 ± 0.1	87.38 ± 0.1
	IDEA	$\mathbf{72.94 \pm 1.9}$	$\mathbf{63.16 \pm 1.0}$	$\mathbf{73.63 \pm 0.8}$	$\mathbf{84.68 \pm 0.3}$	$\mathbf{91.21 \pm 0.1}$
GIN	**Re-Training**	82.90 ± 0.6	74.27 ± 0.5	85.31 ± 0.6	90.28 ± 0.2	95.57 ± 0.2
	Random	69.25 ± 6.3	51.85 ± 2.7	83.64 ± 1.2	89.17 ± 0.1	91.74 ± 0.5
	BEKM	74.05 ± 3.5	65.17 ± 2.8	84.35 ± 0.3	89.39 ± 0.5	92.30 ± 0.3
	BLPA	62.48 ± 2.9	55.06 ± 7.2	82.25 ± 1.6	62.29 ± 0.7	71.66 ± 1.4
	IDEA	$\mathbf{72.57 \pm 2.8}$	$\mathbf{66.37 \pm 4.6}$	$\mathbf{82.33 \pm 0.2}$	$\mathbf{88.48 \pm 0.6}$	$\mathbf{94.63 \pm 0.1}$
GAT	**Re-Training**	83.76 ± 0.3	75.88 ± 0.1	85.02 ± 0.1	92.24 ± 0.1	95.28 ± 0.1
	Random	58.18 ± 2.0	55.43 ± 4.3	68.20 ± 6.9	80.75 ± 0.1	78.26 ± 0.1
	BEKM	64.20 ± 1.5	57.35 ± 2.8	71.67 ± 0.2	80.37 ± 0.3	77.47 ± 0.2
	BLPA	60.88 ± 1.0	58.26 ± 2.6	67.34 ± 3.4	85.22 ± 0.2	86.12 ± 0.2
	IDEA	$\mathbf{84.38 \pm 0.6}$	$\mathbf{75.78 \pm 0.9}$	$\mathbf{84.92 \pm 0.2}$	$\mathbf{92.20 \pm 0.2}$	$\mathbf{95.41 \pm 0.0}$

bounds between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$ based on the common assumption of the objective function being Lipschitz continuous [209]. We compare the bounds and the ℓ_2 distances below. *(1) CEU Worst Bound.* We compute the theoretical worst bound derived based on CEU as a baseline of the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. *(2) CEU Data-Dependent Bound.* We compute the data-dependent bound derived based on CEU as a baseline of the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. A data-dependent bound is tighter than the Worst Bound. *(3) IDEA Bound.* We compute the bound given by Proposition 5.2.3 as the bound for the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. *(4) Actual Values.* We compare the bounds above with the actual ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. Note that we focus on edge unlearning tasks to analyze the tightness of the derived bounds, since this is the only unlearning task CEU supports. We use *Unlearn Ratio* to refer to the ratio of edges to be unlearned from the GNN.

We present the bounds and the actual value of the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$ over a wide range of unlearn ratios (from 1% to 10%), which covers common values, in Fig. 5.8. We also have similar observations in other cases[3]. We summarize the observations below. (1) From the perspective of the general tendency, we observe that larger unlearn ratios usually lead to larger values in both the derived bounds and the actual ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. This reveals that a larger unlearn ratio tends to make the approximation of $\tilde{\boldsymbol{\theta}}^*$ (with the calculated $\bar{\boldsymbol{\theta}}^*$) more difficult, which is in alignment with existing works [209]. (2) From the perspective of the bound tightness, we found that IDEA is able to give tighter bounds in all cases compared with the bounds given by CEU, especially in cases with larger unlearn ratios. This reveals that the approximation of $\tilde{\boldsymbol{\theta}}^*$ given by IDEA can better characterize the difference between $\tilde{\boldsymbol{\theta}}^*$ and $\boldsymbol{\theta}^*$ compared with CEU.

[3]See online version for supplementary discussion and experimental results.

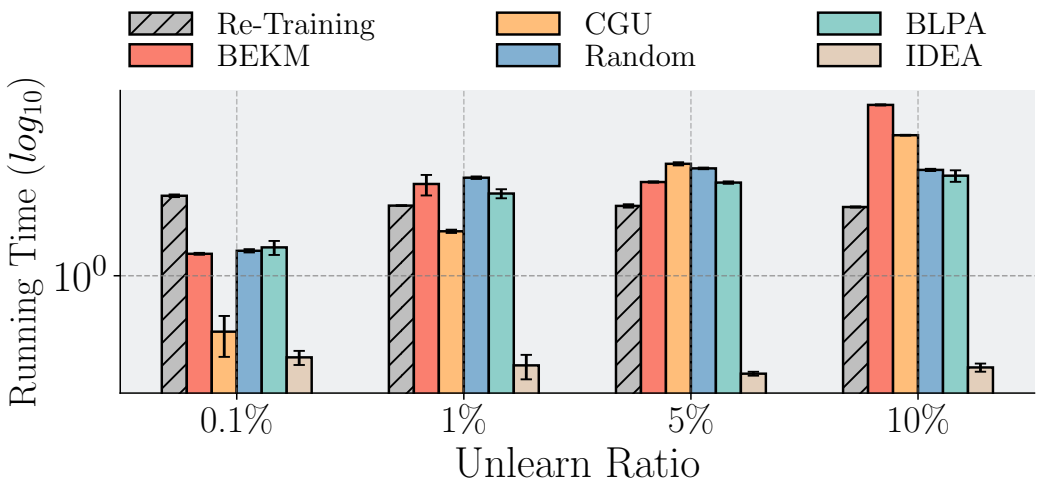


FIGURE 5.9. Efficiency comparison between IDEA and other baselines including retraining. Running time is measured with seconds in log scale.

5.2.5.3 Evaluation of Unlearning Efficiency

To answer **RQ2**, we then evaluate the efficiency of IDEA in performing unlearning. Specifically, we adopt the common node unlearning task as an example, and we measure the running time of unlearning in seconds. We note that CGU only supports performing unlearning on SGC, and thus we adopt SGC as the backbone GNN for IDEA and all other baselines for a fair comparison. We use *Unlearn Ratio* to refer to the ratio of training nodes to be unlearned from the GNN. Here we present a comparison between IDEA and baselines on Cora dataset in Fig. 5.9. We also have similar observations on other GNNs and datasets[4]. We summarize the observations below. (1) From the perspective of the general tendency, we observe that the running time of re-training does not change across different unlearning ratios. This is because the number of optimization epochs dominates the running time of re-training, while the total epoch number does not change no matter how many training nodes are removed. However, the efficiency of all other baselines is sensitive to the unlearning ratio, and this is because their running time is closely dependent on the total number of nodes to be unlearned. Finally, we found that the running time of IDEA is not sensitive to the unlearn ratio. This is because the number of nodes to be unlearned will only marginally influence the computational costs associated with Theorem 5.2.1. The stable running time across different numbers of nodes to be unlearned serves as a key superiority of IDEA over other baselines. (2) From the perspective of time comparison, we found that IDEA achieves significant superiority over all other baselines across the wide range of unlearning ratios, especially on relatively large ones (e.g., 10%). Such an observation indicates that IDEA is able to perform unlearning with satisfying efficiency, which further reveals its practical significance in real-world applications.

[4]See online version for supplementary discussion and experimental results.

5.2.5.4 Evaluation of Model Utility

To answer **RQ3**, we now compare model utility after performing unlearning with IDEA and other baselines. We note that GraphEraser is the only baseline that supports flexible generalization across different GNN backbones. Therefore, we adopt the three variants of GraphEraser, i.e., Random, BEKM, and BLPA, as the corresponding baselines for comparison. We adopt the most common task of node unlearning, and we adopt the F1 score (of node classification) to measure the model utility after re-training/unlearning. We present comprehensive empirical results (including four different GNN backbones and all five real-world datasets) in Table 5.8. In addition to the baselines, we also report the performance of re-training, i.e., the F1 score given by a re-trained model with the unlearned nodes being removed from the training graph, for comparison. We summarize the observations below, and similar observations are also found in different settings[5]. (1) From the perspective of the general tendency, we observe that unlearning approaches are usually associated with worse utility performance compared with re-training. Such a sacrifice is usually considered acceptable, since these unlearning approaches can bring significant improvement in efficiency compared with re-training. (2) From the perspective of model utility, we found that IDEA achieves competitive utility compared with other baselines. Specifically, compared with re-training, IDEA only sacrifices limited utility performance in most cases, and shows better performance in certain cases. Furthermore, IDEA shows consistent superiority compared with alternatives in most cases.

5.2.5.5 Evaluation of Unlearning Effectiveness

To answer **RQ4**, we compare the unlearning effectiveness of IDEA and other baselines. Specifically, we utilize the state-of-the-art attack methods MIA-Graph and StealLink to evaluate the unlearning effectiveness of node and edge unlearning tasks, respectively. To also have CGU as a baseline, we adopt SGC as the backbone GNN to ensure a fair comparison. We present the attack successful rates after node and edge unlearning in Table 5.9. All attacks are performed over those unlearned nodes/edges, and thus a lower AUC score represents better unlearning performance. In terms of node attributes unlearning, we note that to the best of our knowledge, no existing membership inference attack method supports the associated attack. Here we use the average loss value as an unlearning performance indicator. Specifically, we perform partial attribute unlearning under different ratios (20%, 50%, 80%) of unlearn attribute dimensions to the total attribute dimensions. Note that partial attribute unlearning aims to twist the GNN model such that the GNN model behaves as if it were trained on those nodes with the unlearn attribute values being set to non-informative numbers (as in Section 5.2.3). Here we follow a common choice [30] to set such a number as zero. Accordingly, we evaluate the performance with the average loss values regarding the nodes with the unlearn attributes being set to zeros, and a lower loss value indicates better unlearning effectiveness. We present the results in Table 5.10. Note that CGU only supports full attribute unlearning, while the three variants of GraphEraser only support node/edge unlearning. Therefore, we perform attribute unlearning and node unlearning for CGU and GraphEraser, respectively. Based on the settings above, we have the observations below, and consistent observations are also found under different settings[6]. (1) From the perspective of node and edge unlearning, we

[5]See online version for supplementary discussion and experiments.

[6]See online version for supplementary discussion and experiments.

TABLE 5.9. Attack AUC scores after node and edge unlearning on Cora. The results given by IDEA are marked in bold.

	Node Unlearning (↓)	**Edge Unlearning (↓)**
Random	50.38 ± 0.5	55.64 ± 2.8
BEKM	50.35 ± 1.2	51.81 ± 0.3
BLPA	50.30 ± 0.4	50.84 ± 3.4
CGU	54.67 ± 2.9	66.52 ± 0.6
IDEA	$\mathbf{50.86 \pm 1.8}$	$\mathbf{50.11 \pm 0.9}$

TABLE 5.10. Average loss values on Cora regarding the nodes with the unlearn attributes being set to zeros. Ratio of unlearn node attribute dimensions to all attribute dimensions varies across 20%, 50%, and 80%. Lower values represent better performance, and results from IDEA are marked in bold.

	20% (↓)	**50% (↓)**	**80% (↓)**
Random	1.32 ± 0.06	1.38 ± 0.06	1.35 ± 0.09
BEKM	1.41 ± 0.16	1.47 ± 0.14	1.39 ± 0.12
BLPA	1.47 ± 0.11	1.69 ± 0.37	1.50 ± 0.05
CGU	1.62 ± 0.02	1.73 ± 0.04	1.78 ± 0.06
IDEA	$\mathbf{1.29 \pm 0.01}$	$\mathbf{1.31 \pm 0.01}$	$\mathbf{1.33 \pm 0.01}$

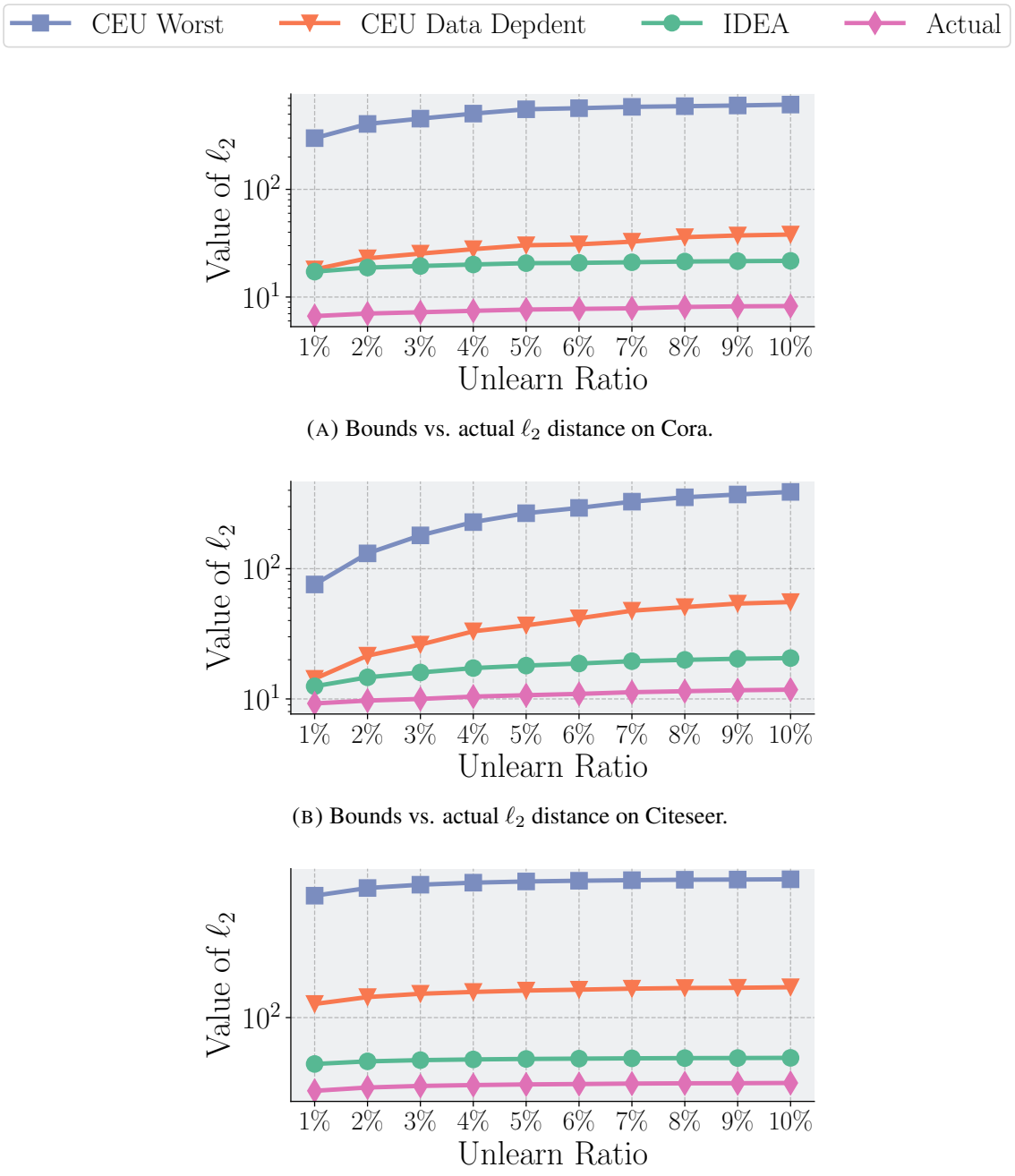
observe that the attack AUC scores over IDEA are among the lowest in both unlearning tasks. Noticing that the AUC scores given by IDEA are only marginally above 50%, the unlearned node/edge information has been almost completely removed from the trained GNNs. (2) From the perspective of attribute unlearning, IDEA exhibits the lowest average loss values in all (attribute) unlearn ratios. This indicates the superior attribute unlearning performance.

5.2.5.6 Evaluation of Bound Tightness

In this subsection, we present additional experimental results regarding the bound tightness of the proposed model IDEA. Specifically, here we adopt SGC as our backbone GNN model, and we present the comparison between three bounds and actual value of the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$ in Figure 5.10.

We present an review of the introduction for the bounds and the ℓ_2 distances below (as in Section 5.2.5.2). *(1) CEU Worst Bound.* We compute the theoretical worst bound derived based on CEU as a baseline of the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. *(2) CEU Data-Dependent Bound.* We compute the data-dependent bound derived based on CEU as a baseline of the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. Usually, data-dependent bound is tighter than the CEU Worst Bound. *(3) IDEA Bound.* We compute the bound given by Proposition 5.2.3 as the IDEA bound for the ℓ_2 distance bound between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. *(4) Actual Values.* We compare the bounds above with the actual values of the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. In addition, we also follow the wide range of unlearning ratios as presented in Section 5.2.5.2.

Below we summarize the observations, which remains consistent with the observations presented in Section 5.2.5.2 and are also found on other GNNs and datasets. (1) From the perspective of the general tendency, we observe that when the value of unlearn ratio is increased (i.e., more edges are unlearned), it generally results in higher values for both the



(A) Bounds vs. actual ℓ_2 distance on Cora.

(B) Bounds vs. actual ℓ_2 distance on Citeseer.

(C) Bounds vs. actual ℓ_2 distance on PubMed.

FIGURE 5.10. Bounds and actual value of the ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$, i.e., $\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2$, over Cora, CiteSeer and PubMed datasets. *CEU Worst*, *CEU Data Dependent*, *IDEA*, and *Actual* represent the worst bound based on CEU, the data-dependent bound based on CEU, the bound based on IDEA, and the actual value of $\|\tilde{\boldsymbol{\theta}}^* - \bar{\boldsymbol{\theta}}^*\|_2$ derived from re-training, respectively.

obtained bounds and the actual ℓ_2 distance between $\tilde{\boldsymbol{\theta}}^*$ and $\bar{\boldsymbol{\theta}}^*$. This reveals that a higher unlearn ratio tends to make approximating $\tilde{\boldsymbol{\theta}}^*$ (using the calculated $\bar{\boldsymbol{\theta}}^*$) more difficult, which is also consistent with prior research [209]. (2) From the perspective of the bound tightness, the obtained results indicate that IDEA consistently provides tighter bounds than those produced

TABLE 5.11. The running time (in seconds) of edge unlearning on Coauthor-CS dataset based on GCN. CGU is excluded from comparison since its backbone only support SGC. The running time of IDEA is marked in bold.

	0.1%	**1%**	**5%**	**10%**
Re-Training	170.0 ± 4.5	170.7 ± 5.6	169.8 ± 7.1	201.2 ± 14
Random	11.54 ± 0.8	11.09 ± 0.2	11.20 ± 0.2	11.07 ± 0.5
BEKM	9.66 ± 0.05	11.23 ± 0.3	10.88 ± 0.2	11.55 ± 0.3
BLPA	10.75 ± 0.1	10.71 ± 0.3	10.77 ± 0.2	10.94 ± 0.2
IDEA	$\mathbf{2.823 \pm 0.0}$	$\mathbf{2.874 \pm 0.1}$	$\mathbf{2.851 \pm 0.1}$	$\mathbf{3.328 \pm 0.1}$

by CEU across all scenarios, especially in cases with higher unlearn ratios. This suggests that the approximation of $\tilde{\boldsymbol{\theta}}^*$ offered by IDEA can capture the distance between $\tilde{\boldsymbol{\theta}}^*$ and $\boldsymbol{\theta}^*$ more accurately compared to CEU, especially for larger unlearn ratios.

5.2.5.7 Evaluation of Unlearning Efficiency

In this subsection, we present additional experimental results regarding the unlearning efficiency on different tasks. Specifically, we have shown efficiency comparison between IDEA and other baselines on node unlearning task in Section 5.2.5.3, and here we present additional results under edge unlearning task based on GCN, with all other settings being consistent with the experiments presented in Section 5.2.5.3. We show edge unlearning performance under a wide range of unlearning ratios (i.e., the ratio of edges to be unlearned to the total number of edges in the graph) from 0.1% to 10%. We present the experimental results in Table 5.11. We observe that the running time of the three variants of GraphEraser does not change as much as in Section 5.2.5.3 across the wide range of ratios. This is because GraphEraser perform partition over the input graph, which results in different shards. Each shard will contribute to the overall running time when the unlearn node/edge appear in such a shard. However, the number of edges is much more than the number of nodes. As a consequence, unlearn edges appears in most shards even if the unlearn ratio is as small as 0.1%. Hence most shards contribute to the running time in most cases, leading to a more stable running time across all ratios. Meanwhile, we also observe that IDEA not only achieves significantly less running time compared with the re-training approach, but also costs less running time compared with all other baselines. Such observation is consistent with the observation in Section 5.2.5.3, and can also be found on other datasets and GNNs. This indicates the superiority of IDEA in terms of the unlearning efficiency.

TABLE 5.12. AUC scores of attacks on Co-author CS based on GCN after node and edge unlearning, respectively. The results given by IDEA is marked in bold.

	Node Unlearning (↓)	**Edge Unlearning (↓)**
Random	51.51 ± 0.7	50.26 ± 0.4
BEKM	50.24 ± 0.8	50.26 ± 0.1
BLPA	51.42 ± 0.8	50.04 ± 0.2
IDEA	$\mathbf{50.15 \pm 0.8}$	$\mathbf{50.02 \pm 0.7}$

TABLE 5.14. Node classification accuracy after edge unlearning. The results marked with (U) and (R) are accuracy values derived from unlearning with IDEA and from re-training, respectively. The accuracy values of IDEA are marked in bold for the convenience of comparison.

	0.1%	**1%**	**5%**	**10%**
Cora (U)	**81.18 ± 0.9**	**81.18 ± 1.4**	**79.34 ± 0.8**	**78.35 ± 1.0**
Cora (R)	84.50 ± 0.3	84.38 ± 0.3	82.41 ± 0.5	82.29 ± 0.3
CiteSeer (U)	**69.87 ± 1.3**	**69.87 ± 0.8**	**68.47 ± 0.8**	**67.27 ± 0.6**
CiteSeer (R)	74.97 ± 0.3	75.38 ± 0.5	72.97 ± 0.7	73.57 ± 0.7
PubMed (U)	**81.02 ± 1.0**	**80.90 ± 0.9**	**79.63 ± 0.9**	**77.96 ± 0.8**
PubMed (R)	84.99 ± 0.1	84.77 ± 0.0	83.82 ± 0.1	81.91 ± 0.1
CS (U)	**88.62 ± 5.4**	**92.42 ± 0.3**	**91.58 ± 0.3**	**90.91 ± 0.7**
CS (R)	91.89 ± 1.6	92.73 ± 0.3	92.82 ± 0.1	91.84 ± 0.2
Physics (U)	**95.68 ± 0.4**	**95.64 ± 0.1**	**95.50 ± 0.3**	**95.19 ± 0.3**
Physics (R)	96.11 ± 0.2	96.03 ± 0.2	96.00 ± 0.2	95.86 ± 0.2

TABLE 5.13. Average loss values regarding the nodes with the unlearn attribute values being set to zero on CiteSeer. Lower values represents better unlearning performance, and the results given by IDEA are marked in bold.

	20% (↓)	**50% (↓)**	**80% (↓)**
Random	1.44 ± 0.09	1.43 ± 0.08	1.48 ± 0.12
BEKM	1.44 ± 0.06	1.50 ± 0.04	1.51 ± 0.09
BLPA	1.40 ± 0.19	1.45 ± 0.27	1.50 ± 0.09
IDEA	**1.25 ± 0.01**	**1.28 ± 0.01**	**1.33 ± 0.02**

5.2.5.8 Evaluation of Model Utility

We now present additional results for the evaluation of model utility. We have shown the node classification accuracy comparison based on the SGC model in Section 5.2.5.4, and here we show the node classification accuracy comparison between IDEA and re-training based on the GCN model in Table 5.14. We maintain all other settings to be consistent with those in Section 5.2.5.4. The model utility given by IDEA and re-training is compared across a wide range of unlearn ratio values (from 0.1% to 10%). We observe that the unlearning given by IDEA only sacrifices limited utility compared with re-training, which remains consistent with the observation in Section 5.2.5.4, and such an observation can also be found on other datasets, unlearning tasks and GNNs. This indicates the satisfying usability of IDEA.

5.2.5.9 Evaluation of Unlearning Effectiveness

Finally, we present additional results for the evaluation of unlearning effectiveness. To evaluate the generalization capability of IDEA, here we utilize a different GNN model (compared with the results in Section 5.2.5.5), which is GCN, as the backbone. We adopt a consistent evaluation protocol as shown in Section 5.2.5.5, and we present the experimental results of node/edge unlearning and attribute unlearning in Table 5.12 and Table 5.13, respectively. We found that, first, in terms of node and edge unlearning tasks, we observe that the attacks on IDEA show the lowest attack successful AUC scores, which are marginally above 50%

(almost equivalent to random guess). This indicates the effectiveness of IDEA in performing node and edge unlearning. Second, we observe that the average loss values given by IDEA are the lowest among all baselines. Since the loss values are collected from the nodes whose unlearn attribute values have already been set to zero, a lower loss value indicates better unlearning effectiveness. Therefore, the satisfying effectiveness of IDEA is further validated. Additionally, we note that these observations can also be found on other unlearning tasks, datasets, and GNNs, which indicates the satisfying unlearning effectiveness of IDEA.

5.2.6 Related Work

Certified Machine Unlearning. The general desiderata of machine unlearning is to remove the influence of certain training data on the model parameters, such that the model can behave as if it never saw such data [219, 145, 14]. Re-training the model without making the unlearning data visible is an ideal way to achieve such a goal, while it is usually infeasible in practice due to various reasons such as prohibitively high computational costs. A popular way to approach the goal of unlearning is to directly approximate the re-trained model parameters, a.k.a., *approximate unlearning* [219, 186]. Certified machine unlearning is under the umbrella of approximate unlearning, and it has stood out due to the capability of providing theoretical guarantee on the unlearning effectiveness. A commonly used criterion of certified unlearning is $(\varepsilon - \delta)$ *certified unlearning* [70, 172], which utilizes two parameters ε and δ to describe the proximity between the re-trained model parameter distribution and approximated model parameter distribution in the model space. In recent years, various techniques have been proposed to achieve certified unlearning [241, 202, 139]. However, they overwhelmingly focus on independent, identically distributed (i.i.d.) data and fail to consider the dependency between data points. Therefore, they cannot be directly adopted to perform unlearning over GNNs. Different from the works above, our paper proposes a certified unlearning approach for GNNs, which necessitates the modeling of dependencies between instances in graphs (e.g., nodes and edges).

Machine Unlearning for Graph Neural Networks. Over the years, GNNs have been increasingly deployed in a plethora of applications [245, 210]. Similar to other machine learning models, they also face the risk of privacy leakage, where the private information is considered to be encoded in the training data [167]. Such a threat has prompted the emerging of unlearning approaches for GNNs [26, 149, 29, 208]. However, these works only achieve unlearning for GNNs empirically, failing to provide theoretical guarantee on the effectiveness. To further strengthen the power of unlearning for GNNs and enhance the confidence of model owners before model deployment, a few recent works have initiated explorations on certified unlearning for GNNs. Wu et al. [209] propose CEU to unlearn edges that are visible to GNNs during training, while edge unlearning is the only type of request it is able to handle. Chien et al. [30] proposed a different certified unlearning approach for GNNs to also handle node and attribute unlearning requests, while such an approach is only applicable to a specially simplified GNN model. Meanwhile, these approaches can only handle limited types of unlearning requests, which further jeopardizes their flexibility in real-world applications. Different from these works, our paper proposes a flexible unlearning framework that can handle different types of unlearning requests. On top of this framework, an effectiveness certification is proposed without relying on specific GNN structures or objective functions.

5.2.7 Conclusion

In this paper, we propose IDEA, a flexible framework of certified unlearning for GNNs. Specifically, we first formulate and study a novel problem of flexible and certified unlearning for GNNs, which aims to flexibly handle different unlearning requests with theoretical guarantee. To tackle this problem, we develop IDEA by analyzing the objective difference before and after certain information is removed from the graph. We further present theoretical guarantee as the certification for unlearning effectiveness. Extensive experiments on real-world datasets demonstrate the superiority of IDEA in multiple key perspectives. Meanwhile, two future directions are worth further investigation. First, we focus on the common node classification task in this paper, and we will extend the proposed framework to other tasks, such as graph classification. Second, considering that GNNs may be trained in a decentralized manner, it is critical to study GNN unlearning under a distributed setting.

CHAPTER 6

Conclusion and Future Directions

In this chapter, we first conclude the key research contributions of this dissertation, and then discuss the future research directions in the area of fair graph machine learning.

6.1 Key Research Contributions

In this section, we summarize the research contributions in explanation, optimization, and certification from the perspective of fairness for graph machine learning.

Fairness Explanation in Graph Machine Learning. In this research theme, we present two research works that achieve explanation for Graph Neural Networks (GNNs) from the data and model perspectives, respectively. We summarize their main contributions below.

In the first work, we propose an innovative framework to interpret, measure and mitigate unfairness in GNNs at the level of individual training nodes. This contributes to the potential for developing more transparent and fair graph machine learning methods. We summarize the main contributions below. First, this work formulates a novel problem of GNNs by attributing model bias to the influence of specific training nodes. This provides a new perspective on understanding how bias arises in GNNs. Second, this work proposes a new fairness metric called Probabilistic Distribution Disparity (PDD) which measures the bias exhibited in the probabilistic outputs of GNNs. PDD provides finer granularity compared to traditional fairness metrics computed on predicted labels, allowing it to better capture the influence of individual training nodes on the bias exhibited by the model. Third, this work develops an efficient algorithm named BIND to estimate the influence of each training node on the exhibited bias (by the model) without requiring expensive retraining of GNNs. The algorithm analyzes the training loss to characterize dependencies between nodes, which enables us to characterize the non-IID nature of graph data. Finally, based on the experiments on real-world datasets, it demonstrates: (1) PDD is consistent with traditional fairness metrics; (2) BIND can efficiently and effectively estimate node influence on model bias; (3) Debiasing GNNs by deleting harmful training nodes identified by BIND can reduce bias while preserving utility.

In the second work, we study a novel problem of providing structural explanations of bias exhibited in GNNs. This research direction aims to improve the transparency of the fairness levels of GNNs in terms of the graph structure. First, this work formulates the problem of identifying two edge sets in a node's computation graph that can maximally account for the bias and maximally contribute to the fairness of the GNN's prediction for that node. Explaining both bias and fairness provides a more comprehensive understanding compared to only explaining one aspect. Second, this work proposes a novel metric to quantify the

bias in a node's GNN prediction by measuring its contribution to the overall distribution difference in the output space between different sensitive groups. This novel bias metric enables quantitative explanations of bias. Third, this work introduces REFEREE, a novel framework with two explainers that learn to identify the aforementioned edge sets while remaining faithful to the original predictions. The two explainers work in a contrastive manner to better distinguish bias-inducing and fairness-promoting edges. Finally, experiments on real-world datasets demonstrate: (1) REFEREE outperforms baselines in identifying edges that explain bias and fairness; (2) REFEREE's explanations maintain high fidelity to the original GNN's predictions; (3) Removing edges identified by REFEREE as promoting bias can effectively mitigate the overall bias of GNNs.

Fairness Optimization in Graph Machine Learning. In this research theme, we present three research works that achieve GNN bias mitigation. Here, two works focus on the commonly used group fairness notion, and one work aims to achieve bias mitigation from the more granular individual fairness perspective. We summarize their contributions below.

While most prior work on fairness in graph mining has focused on group fairness with respect to protected attributes, the first work explores the less-studied notion of individual fairness to ensure GNNs treat similar nodes similarly. First, this work refines the notion of individual fairness from a ranking perspective. Specifically, this paper defines individual fairness based on the consistency between the similarity ranking of nodes to any given node in the input space and the output space. Second, in this work, we proposes a novel plug-and-play framework named REDRESS to optimize individual fairness in GNNs while preserving the model utility. REDRESS optimizes GNN utility and individual fairness jointly in an end-to-end manner, where the individual fairness promotion is achieved through a ranking-based optimization. Third, extensive experiments in node classification and link prediction tasks on real-world graphs demonstrate the superiority of REDRESS compared with other alternatives on both fairness and utility metrics.

In the second work, we propose a novel framework named EDITS to mitigate the bias encoded in the input graph data for GNNs in a model-agnostic manner. This enables training fairer GNNs without needing to modify the GNN architecture or perform retraining. Specifically, this work first formulates a novel problem of debiasing attributed networks used as input for GNNs. We analyze how bias can arise from both node attributes and graph structure, and how these propagate bias in GNNs. This provides insights into the sources of bias in graph data. Second, we propose novel metrics to quantify attribute bias and structural bias in attributed networks. These enable measuring bias directly in the input graph data. Third, we develop a debiasing framework named EDITS to mitigate bias in both node attributes and graph structure while preserving the GNN performance on downstream tasks. We use separate modules for attribute debiasing and structural debiasing. EDITS works in a model-agnostic way, i.e., it does not rely on the adopted GNN structure. Finally, based on experiments on both synthetic and real-world datasets, we demonstrate: (1) the proposed bias metrics can effectively quantify bias in graph data; (2) EDITS can successfully mitigate bias in the input attributed networks; (3) using the debiased graph data from EDITS can reduce bias in the predictions of various GNN models while maintaining utility.

In the third work, we take initial steps towards developing a fair knowledge distillation framework for compressing GNNs while mitigating bias at the same time. Specifically, first,

we formulate a novel problem of fair knowledge distillation for GNN-based teacher-student frameworks. This draws attention to an understudied but critical fairness issue in GNN model compression. Second, we propose a principled framework named RELIANT to learn a less biased student GNN. A key innovation is learning a proxy of the model bias exhibited on the training nodes, which enables debiasing the student model's predictions by excluding the bias proxy during inference. Importantly, RELIANT is designed to be agnostic to the specific teacher and student model architectures, whcih allows it to be easily adapted to various GNN knowledge distillation approaches. Third, we reformulate the debiasing objective using differentiable polynomials to approximate bias under traditional fairness notions. This enables RELIANT to be optimized in an end-to-end manner using efficient gradient-based techniques. Finally, we conduct extensive experiments on multiple real-world datasets which demonstrate that RELIANT can effectively debias the student model while maintaining comparable utility as the teacher GNN across different knowledge distillation frameworks and GNN backbones.

Fairness Certification in Graph Machine Learning. This research theme serves as an early investigation into the problem of achieving certification for GNNs from the perspective of algorithmic fairness. We summarize the contributions of the two introduced works below.

In the first work, we formulates a novel research problem of certifying GNN classifiers on their fairness levels. The goal is to obtain a classifier on top of an optimized GNN that will be guaranteed not to exhibit bias above a given threshold under certain perturbation budgets, no matter what perturbations are made to the node attributes and/or graph structure. This work proposes a principled framework called ELEGANT to address this problem. ELEGANT serves as a plug-and-play framework that can achieve certified fairness defense for any optimized GNN ready to be deployed, without relying on assumptions about the GNN structure or parameters. In particular, the proposed framework ELEGANT leverages randomized smoothing techniques to defend against malicious attacks on both node attributes and graph topology in a concurrent manner. It provides theoretical certification on the fairness of GNNs under the perturbation budgets. Third, theoretical analysis is provided to derive the certified perturbation budgets on node attributes and graph topology that ELEGANT can take while provably maintaining the fairness of GNN predictions under a specified threshold. Finally, extensive experiments on real-world datasets demonstrate the effectiveness of the proposed framework in achieving high fairness certification rates across different GNN backbones while maintaining comparable utility to the vanilla GNNs. Additionally, ELEGANT is also shown to be beneficial for debiasing GNNs as a byproduct.

In the second work, we note that certain sensitive information such as gender or race can be encoded in the GNNs in the training stage, and such information may bring more bias to the model. Meanwhile, the consent (from the involved individuals) of using such information can also be withdrawn. Therefore, it becomes a critical need to remove (i.e., unlearn) such information from an optimized GNN model. Accordingly, we formulate a novel problem of achieving flexible and certified unlearning for GNNs. We instantiate four types of practical unlearning requests, including node, edge, full attribute, and partial attribute unlearning. This expands the scope of GNN unlearning beyond the commonly studied node and edge unlearning scenarios. We further design the IDEA framework to handle diverse unlearning requests and provide theoretical guarantees on the unlearning effectiveness. Importantly, IDEA is not tailored to specific GNN architectures or objective functions, enhancing its

flexibility and generalizability. Then, we present the theoretical certification of the unlearning effectiveness in IDEA. Such a certification provides tighter bounds on the distance between the approximated and actual parameters after information removal than existing alternatives. This strengthens the reliability of the unlearning process and boosts confidence in deploying the unlearned models. Finally, we conduct comprehensive experiments on real-world datasets to demonstrate the superiority of IDEA over state-of-the-art baselines in terms of bound tightness, unlearning efficiency, model utility preservation, and unlearning effectiveness against privacy attacks. These results highlight the practical significance of IDEA.

6.2 Future Directions

6.2.1 Short-Term Plan

Benchmarking Large-Scale Responsible Graph Machine Learning. Despite the significant progress of responsible graph machine learning, there are still challenges deserving further research, and the problem of benchmarking the performance of responsible graph machine learning is among the most significant ones. Reasons include (1) existing works are under inconsistent settings and (2) most existing works only perform experiments on graph datasets with limited scales. Correspondingly, it becomes difficult to compare the performance across different works, and their usability also remains unclear. Through large-scale benchmarking, I aim to reveal their performances in not only fairness but also other related perspectives (e.g., model transparency and privacy), presenting a clear landscape of advances in responsible graph machine learning.

Exploring Pareto Optimal for Responsible Graph Machine Learning. Existing efforts towards responsible graph machine learning usually sacrifice the performance from other perspectives such as utility (e.g., accuracy in node classification tasks). Nevertheless, it remains unclear (1) whether it is possible to achieve responsible models at no or little costs; and (2) where the theoretical Pareto optimal boundary of improving accountability versus the model performance in utility is. Properly answering these questions is crucial, since it is necessary to evaluate whether the cost is affordable or not before the deployment of responsible graph machine learning models. Through this, I aim to reveal a clearer boundary of costs in achieving accountability, and ultimately facilitate the benefits we gain from responsible graph machine learning.

Building Privacy-Preserving Graph Machine Learning. If a graph machine learning model potentially makes biased predictions, then there is a risk of privacy leakage. A preliminary reason is that most biased predictions are dependent on the sensitive attributes regarding the individuals involved, such as their gender, race, and occupation. For example, consider a social network where nodes represent individuals and edges represent their social connections. A graph-based recommendation system may inadvertently expose the likelihood of someone belonging to a particular minority group based on the homogeneous nature of their immediate network connections. This not only can lead to biased recommendations, but also risk revealing the privacy of social connections for the involved individuals without their explicit consent. Therefore, to facilitate responsible graph machine learning, it is an urgent need to ensure that these models are privacy-preserving before deployment.

6.2.2 Long-Term Plan

Facilitating Responsible AI in Human-Computer Interaction (HCI). A bigger picture behind my current study is responsible AI, which directly benefits the interaction between human and technology. Correspondingly, a huge potential of my research lies in facilitating the development of intuitive, explainable, and privacy-preserving AI technologies tailored for the next-generation HCI systems. The core of this research direction is developing interactive frameworks that can seamlessly interpret and represent the nature of human behavior, preferences, and cognitive processes when interacting with AI-driven interfaces. The challenges in this domain arise from the complex human dynamics, emotions, and the vast range of interaction modalities (e.g., voice and gestures). To address these, I will explore the following pertinent questions: (1) How can we balance the automation of AI with the need of users to control, ensuring that AI does not override human decisions in interactive scenarios? (2) How can we amplify the generalization capacity of responsible AI models to help users from different demographic subgroups interact and engage well with systems? (3) How can we ensure the robustness and trustworthiness of AI models used for HCI systems, especially when only ambiguous or incomplete user inputs and feedback are available? (4) How can we design AI-powered HCI systems to ethically recognize and adapt to diverse cultural, social, and personal user contexts to ensure a globally inclusive interaction experience? I am firmly convinced that pioneering efforts in this domain will pave the way for more humane, transparent, and adaptable AI systems that truly resonate with human needs and aspirations. As part of my commitment to pave the advancement in this area, I will actively seek funding opportunities.

Supporting Sustainable Decision-Making with AI. I have worked on bridging the gap between responsible graph machine learning algorithms with inclusive decision-making. However, in a higher level, how to achieve more sustainable decision-making with the help of machine learning algorithms remains unanswered, e.g., promoting long-term societal benefits, environmental health, and equitable growth. Therefore, there remain various interesting research topics in bridging the gap between responsible AI algorithms and sustainable decision-making. I believe it would continuously provide guidance to the next-generation AI, covering a wide range of applications (e.g., bio-medicine, recommender system, epidemiological study, economic analysis, and human-involved AI). In the future, I will mainly explore the following questions in this area: (1) How can we measure the societal impact of AI-driven decisions, especially on marginalized populations and social inequalities? (2) How can we foster collaborations between AI and fields like environmental science and sociology to align AI algorithms with sustainability goals? (3) How can we better detect and mitigate biases in AI that influence crucial sustainability domains like conservation and public health? (4) How can we enhance the transparency and explainability of AI models, ensuring stakeholders in sustainable sectors fully understand and trust AI-driven recommendations and actions?

Investigating AI for Metropolitan Development. Responsible AI also plays a critical role in facilitating metropolitan development, such as providing accountable suggestions to help decision-making for urban planners. For example, with ethical AI technology, urban planners are able to create a more inclusive environment for residents from different communities, and thus the social good can be largely facilitated. Such a goal directly delivers a positive social impact to residents, which is in accordance with the priorities outlined in the United Nations'

17 Sustainable Development Goals (SDGs). In my Ph.D. study, I have rich research and proposal submission opportunities in the domain of responsible AI, including different aspects such as developing AI techniques to mitigate discrimination and facilitate inclusive decision-making in ML model predictions [46, 47, 52] and improving ML model explainability to provide feedback for human [51, 48, 137]. In the future, I will investigate the following research questions: (1) How can we develop responsible AI frameworks to mitigate societal biases and disparities between different demographic subgroups in urban planning? (2) How can we utilize AI-powered frameworks to ensure that the metropolitan feedback and monitoring system serves various demographic groups, especially marginalized populations? and (3) How can we take advantage of the developed AI systems to foster community engagement and implement iterative feedback mechanisms? (4) How can AI be utilized to optimize resource allocation in urban planning, ensuring equitable access to essential services and opportunities across different demographic groups?

References

[1] Privacy Act. 'Personal Information Protection and Electronic Documents Act'. In: *Department of Justice, Canada. Full text available at http://laws. justice. gc. ca/en/P-8.6/text. html* (2000).

[2] Alekh Agarwal et al. 'A reductions approach to fair classification'. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 60–69.

[3] Chirag Agarwal, Himabindu Lakkaraju and Marinka Zitnik. 'Towards a Unified Framework for Fair and Stable Graph Representation Learning'. In: *UAI*. PMLR. 2021, pp. 2114–2124.

[4] M. Al Hasan et al. 'Link prediction using supervised learning'. In: *Workshop on Link Analysis, Counter-Terrorism and Security*. Vol. 30. 2006, pp. 798–805.

[5] Martin Arjovsky, Soumith Chintala and Léon Bottou. 'Wasserstein GAN'. In: *ICML*. 2017.

[6] Arthur Asuncion and David Newman. *UCI machine learning repository*. 2007.

[7] Federico Baldassarre and Hossein Azizpour. 'Explainability techniques for graph convolutional networks'. In: *arXiv preprint arXiv:1905.13686* (2019).

[8] Solon Barocas, Moritz Hardt and Arvind Narayanan. 'Fairness in machine learning'. In: *NeurIPS Tutorial* 1 (2017).

[9] Alex Beutel et al. 'Data decisions and theoretical implications when adversarially learning fair representations'. In: *arXiv preprint arXiv:1707.00075* (2017).

[10] Aleksandar Bojchevski and Stephan Günnemann. 'Certifiable robustness to graph perturbations'. In: *NeurIPS* 32 (2019).

[11] Aleksandar Bojchevski, Johannes Klicpera and Stephan Günnemann. 'Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more'. In: *ICML*. 2020.

[12] Giorgian Borca-Tasciuc et al. 'Provable Fairness for Neural Network Models using Formal Verification'. In: *arXiv preprint arXiv:2212.08578* (2022).

[13] Avishek Bose and William Hamilton. 'Compositional Fairness Constraints for Graph Embeddings'. In: *ICML*. 2019.

[14] Lucas Bourtoule et al. 'Machine unlearning'. In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2021, pp. 141–159.

[15] Joan Bruna et al. 'Spectral networks and locally connected networks on graphs'. In: *arXiv preprint arXiv:1312.6203* (2013).

[16] Vanessa Buhrmester, David Münch and Michael Arens. 'Analysis of explainers of black box deep neural networks for computer vision: A survey'. In: *MLKE* (2021).

[17] Robin Burke et al. 'Balanced neighborhoods for fairness-aware collaborative recommendation'. In: *RecSys*. 2017.

[18] Maarten Buyl and Tijl De Bie. 'DeBayes: a Bayesian Method for Debiasing Network Embeddings'. In: *Proceedings of the 37th ICML (ICML '20)*. 2020, pp. 1220–1229.

[19] Toon Calders, Faisal Kamiran and Mykola Pechenizkiy. 'Building classifiers with independency constraints'. In: *2009 IEEE ICDM Workshops*. IEEE. 2009, pp. 13–18.
[20] Yinzhi Cao and Junfeng Yang. 'Towards making systems forget with machine unlearning'. In: *2015 IEEE symposium on security and privacy*. IEEE. 2015, pp. 463–480.
[21] Nicholas Carlini and David Wagner. 'Adversarial examples are not easily detected: Bypassing ten detection methods'. In: *AISec*. 2017, pp. 3–14.
[22] Simon Caton and Christian Haas. 'Fairness in machine learning: A survey'. In: *CSUR* (2020).
[23] Olivier Chapelle et al. 'Expected reciprocal rank for graded relevance'. In: *CIKM*. 2009, pp. 621–630.
[24] Hongge Chen et al. 'Multi-stage influence function'. In: *NeurIPS* (2020).
[25] Jianbo Chen et al. 'Learning to explain: An information-theoretic perspective on model interpretation'. In: *ICML*. 2018.
[26] Min Chen et al. 'Graph unlearning'. In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2022, pp. 499–513.
[27] Ming Chen et al. 'Simple and deep graph convolutional networks'. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 1725–1735.
[28] Dawei Cheng et al. 'Graph neural network for fraud detection via spatial-temporal attention'. In: *TKDE* (2020).
[29] Jiali Cheng et al. 'GNNDelete: A General Strategy for Unlearning in Graph Neural Networks'. In: *arXiv preprint arXiv:2302.13406* (2023).
[30] Eli Chien, Chao Pan and Olgica Milenkovic. 'Certified graph unlearning'. In: *arXiv preprint arXiv:2206.09140* (2022).
[31] Eli Chien, Chao Pan and Olgica Milenkovic. 'Efficient model updates for approximate unlearning of graph-structured data'. In: *The Eleventh International Conference on Learning Representations*. 2022.
[32] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*. 92. American Mathematical Soc., 1997.
[33] Jeremy Cohen, Elan Rosenfeld and Zico Kolter. 'Certified adversarial robustness via randomized smoothing'. In: *ICML*. 2019.
[34] R Dennis Cook and Sanford Weisberg. 'Characterizations of an empirical influence function for detecting influential cases in regression'. In: *Technometrics* 22.4 (1980), pp. 495–508.
[35] Sam Corbett-Davies and Sharad Goel. 'The measure and mismeasure of fairness: A critical review of fair machine learning'. In: *NeurIPS*. 2019.
[36] Marco Cuturi and Arnaud Doucet. 'Fast computation of Wasserstein barycenters'. In: *ICML*. 2014.
[37] Enyan Dai and Suhang Wang. 'Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information'. In: *WSDM*. 2021.
[38] Enyan Dai and Suhang Wang. 'Towards Self-Explainable Graph Neural Network'. In: *CIKM*. 2021, pp. 302–311.
[39] Enyan Dai and Suhang Wang. 'Learning fair graph neural networks with limited and private sensitive attribute information'. In: *TKDE* (2022).
[40] Enyan Dai et al. 'A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability'. In: *ACM Computing* (2022).

[41] Quanyu Dai et al. ‘Adversarial training methods for network embedding’. In: *WWW*. 2019, pp. 329–339.
[42] Giuseppe Dattoli, Paolo E Ricci and Clemente Cesarano. ‘A note on Legendre polynomials’. In: *International Journal of Nonlinear Sciences and Numerical Simulation* 2.4 (2001), pp. 365–370.
[43] Michaël Defferrard, Xavier Bresson and Pierre Vandergheynst. ‘Convolutional neural networks on graphs with fast localized spectral filtering’. In: *NeurIPS* (2016).
[44] Yanzhuo Ding et al. ‘Visualizing and understanding neural machine translation’. In: *ACL*. 2017.
[45] Kien Do, Truyen Tran and Svetha Venkatesh. ‘Graph transformation policy network for chemical reaction prediction’. In: *SIGKDD*. 2019, pp. 750–760.
[46] Yushun Dong et al. ‘Individual Fairness for Graph Neural Networks: A Ranking based Approach’. In: *SIGKDD*. 2021, pp. 300–310.
[47] Yushun Dong et al. ‘EDITS: Modeling and Mitigating Data Bias for Graph Neural Networks’. In: *The Web Conf*. 2022, pp. 1259–1269.
[48] Yushun Dong et al. ‘On Structural Explanation of Bias in Graph Neural Networks’. In: *KDD*. 2022.
[49] Yushun Dong et al. ‘ELEGANT: Certified Defense on the Fairness of Graph Neural Networks’. In: *arXiv preprint arXiv:2311.02757* (2023).
[50] Yushun Dong et al. ‘Fairness in graph mining: A survey’. In: *IEEE Transactions on Knowledge and Data Engineering* (2023).
[51] Yushun Dong et al. ‘Interpreting Unfairness in Graph Neural Networks via Training Node Attribution’. In: *AAAI* (2023).
[52] Yushun Dong et al. ‘RELIANT: Fair Knowledge Distillation for Graph Neural Networks’. In: *SDM*. 2023.
[53] Yushun Dong et al. ‘IDEA: A Flexible Framework of Certified Unlearning for Graph Neural Networks’. In: *SIGKDD*. 2024.
[54] Yingtong Dou et al. ‘Enhancing Graph Neural Network-based Fraud Detectors Against Camouflaged Fraudsters’. In: *Proceedings of the 29th CIKM (CIKM ’20)*. 2020, pp. 315–324.
[55] Mengnan Du et al. ‘Fairness in deep learning: A computational perspective’. In: *IEEE Intelligent Systems* 36.4 (2020), pp. 25–34.
[56] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml. Accessed: 2023-02-27. 2017. URL: http://archive.ics.uci.edu/ml.
[57] Keyu Duan et al. ‘A comprehensive study on large-scale graph training: Benchmarking and rethinking’. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 5376–5389.
[58] Cynthia Dwork et al. ‘Fairness through awareness’. In: *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*. 2012, pp. 214–226.
[59] Michael D Ekstrand and Daniel Kluver. ‘Exploring author gender in book rating and recommendation’. In: *User Modeling and User-Adapted Interaction* (2021).
[60] Yanai Elazar and Yoav Goldberg. ‘Adversarial removal of demographic attributes from text data’. In: *arXiv preprint arXiv:1808.06640* (2018).

[61] Negin Entezari et al. ‘All you need is low (rank) defending against adversarial attacks on graphs’. In: *WSDM*. 2020.
[62] Golnoosh F. et al. ‘A Fairness-aware Hybrid Recommender System’. In: *CoRR* abs/1809.09030 (2018).
[63] Wei Fan et al. ‘Fair Graph Auto-Encoder for Unbiased Graph Representations with Wasserstein Distance’. In: *ICDM*. IEEE. 2021, pp. 1054–1059.
[64] Wenqi Fan et al. ‘Graph neural networks for social recommendation’. In: *The Web Conf*. 2019, pp. 417–426.
[65] Michael Feldman et al. ‘Certifying and removing disparate impact’. In: *SIGKDD*. 2015.
[66] Shangbin Feng et al. ‘TwiBot-22: Towards Graph-Based Twitter Bot Detection’. In: *arXiv preprint arXiv:2206.04564* (2022).
[67] Ruth C Fong and Andrea Vedaldi. ‘Interpretable explanations of black boxes by meaningful perturbation’. In: *ICCV*. 2017.
[68] Santo Fortunato. ‘Community detection in graphs’. In: *Physics Reports* 486.3-5 (2010), pp. 75–174.
[69] Jianping Gou et al. ‘Knowledge distillation: A survey’. In: *International Journal of Computer Vision* 129.6 (2021), pp. 1789–1819.
[70] Chuan Guo et al. ‘Certified data removal from machine learning models’. In: *arXiv preprint arXiv:1911.03030* (2019).
[71] Ruocheng Guo, Jundong Li and Huan Liu. ‘Learning individual causal effects from networked observational data’. In: *WSDM*. 2020, pp. 232–240.
[72] Zhiwei Guo and Heng Wang. ‘A deep graph neural network-based mechanism for social recommendations’. In: *TKDE* (2020).
[73] Md Haidar, Mehdi Rezagholizadeh et al. ‘Textkd-gan: Text generation using knowledge distillation and generative adversarial networks’. In: *Canadian conference on artificial intelligence*. Springer. 2019, pp. 107–118.
[74] Will Hamilton, Zhitao Ying and Jure Leskovec. ‘Inductive representation learning on large graphs’. In: *NeurIPS*. Vol. 30. 2017, pp. 1024–1034.
[75] Xiao Han et al. ‘CreditPrint: Credit Investigation via Geographic Footprints by Deep Learning’. In: *arXiv:1910.08734* (2019).
[76] Xudong Han, Timothy Baldwin and Trevor Cohn. ‘Balancing out Bias: Achieving Fairness Through Balanced Training’. In: *arXiv preprint arXiv:2109.08253* (2021).
[77] Moritz Hardt, Eric Price and Nati Srebro. ‘Equality of Opportunity in Supervised Learning’. In: *NeurIPS*. Vol. 29. 2016.
[78] Huarui He et al. ‘Compressing Deep Graph Neural Networks via Adversarial Knowledge Distillation’. In: *arXiv preprint arXiv:2205.11678* (2022).
[79] Xinlei He et al. ‘Stealing links from graph neural networks’. In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021, pp. 2669–2686.
[80] Kexin Huang and Marinka Zitnik. ‘Graph meta learning via local subgraphs’. In: *NeurIPS*. 2020.
[81] Qiang Huang et al. ‘Graphlime: Local interpretable model explanations for graph neural networks’. In: *arXiv preprint arXiv:2001.06216* (2020).
[82] Xiao Huang, Jundong Li and Xia Hu. ‘Label informed attributed network embedding’. In: *WSDM*. 2017.

[83] Hussain Hussain et al. ‘Adversarial Inter-Group Link Injection Degrades the Fairness of Graph Neural Networks’. In: *arXiv preprint arXiv:2209.05957* (2022), pp. 975–980.

[84] Vassilis N Ioannidis, Dimitris Berberidis and Georgios B Giannakis. ‘Graphsac: Detecting anomalies in large-scale graphs’. In: *arXiv preprint arXiv:1910.09589* (2019).

[85] John J Irwin et al. ‘ZINC: a free tool to discover chemistry for biology’. In: *Journal of chemical information and modeling* 52.7 (2012), pp. 1757–1768.

[86] Zachary Izzo et al. ‘Approximate data deletion from machine learning models’. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 2008–2016.

[87] Zeinab S Jalali et al. ‘On the Information Unfairness of Social Networks’. In: *SIAM International Conference on Data Mining*. SIAM. 2020, pp. 613–521.

[88] Kalervo Järvelin and Jaana Kekäläinen. ‘Cumulated gain-based evaluation of IR techniques’. In: *TOIS* 20.4 (2002), pp. 422–446.

[89] Jinyuan Jia et al. ‘Certified robustness of community detection against adversarial structural perturbation via randomized smoothing’. In: *TheWebConf*. 2020, pp. 2718–2724.

[90] Heinrich Jiang and Ofir Nachum. ‘Identifying and correcting label bias in machine learning’. In: *AISTATS*. 2020, pp. 702–712.

[91] Xiaoqi Jiao et al. ‘TinyBERT: Distilling BERT for Natural Language Understanding’. In: *EMNLP*. 2020.

[92] Guangyin Jin et al. ‘Addressing crime situation forecasting task with temporal graph convolutional neural network approach’. In: *ICMTMA*. 2020, pp. 474–478.

[93] Hongwei Jin and Xinhua Zhang. ‘Latent adversarial training of graph convolution networks’. In: *ICML workshop on learning and reasoning with graph-structured representations*. 2019.

[94] Hongwei Jin et al. ‘Certified robustness of graph convolution networks for graph classification under topological attacks’. In: *NeurIPS* (2020).

[95] Jiayin Jin et al. ‘Input-agnostic certified group fairness via gaussian parameter smoothing’. In: *ICML*. PMLR. 2022, pp. 10340–10361.

[96] W. Jin et al. ‘Graph structure learning for robust graph neural networks’. In: *Proceedings of the 26th SIGKDD (KDD ’20)*. 2020, pp. 66–74.

[97] Wei Jin et al. ‘Adversarial Attacks and Defenses on Graphs: A Review, A Tool and Empirical Studies’. In: *arXiv preprint arXiv:2003.00653* (2020).

[98] Wei Jin et al. ‘Node similarity preserving graph convolutional networks’. In: *Proceedings of the 14th ACM international conference on web search and data mining*. 2021, pp. 148–156.

[99] Kareem L Jordan and Tina L Freiburger. ‘The effect of race/ethnicity on sentencing: Examining sentence type, jail length, and prison length’. In: *J Crim Justice* 13.3 (2015), pp. 179–196.

[100] Chaitanya K Joshi et al. ‘On Representation Knowledge Distillation for Graph Neural Networks’. In: *arXiv preprint arXiv:2111.04964* (2021).

[101] Christopher Jung et al. ‘Eliciting and enforcing subjective individual fairness’. In: *arXiv preprint arXiv:1905.10660* (2019).

[102] Faisal Kamiran and Toon Calders. 'Data preprocessing techniques for classification without discrimination'. In: *Knowledge and Information Systems* 33.1 (2012), pp. 1–33.
[103] Jian Kang and Hanghang Tong. 'Fair Graph Mining'. In: *CIKM*. 2021, pp. 4849–4852.
[104] Jian Kang et al. 'InFoRM: Individual Fairness on Graph Mining'. In: *SIGKDD*. 2020, pp. 379–389.
[105] Jian Kang et al. 'RawlsGCN: Towards Rawlsian Difference Principle on Graph Convolutional Network'. In: *WWW*. 2022, pp. 1214–1225.
[106] Mintong Kang et al. 'Certifying some distributional fairness with subpopulation decomposition'. In: *arXiv preprint arXiv:2205.15494* (2022).
[107] Leonid V Kantorovich. 'Mathematical methods of organizing and planning production'. In: *Management science* (1960).
[108] Haitham Khedr and Yasser Shoukry. *CertiFair: A Framework for Certified Global Fairness of Neural Networks*. 2022. arXiv: 2205.09927 [cs.LG].
[109] Diederik P. Kingma and Jimmy Ba. 'Adam: A Method for Stochastic Optimization'. In: *ICLR*. 2015.
[110] Thomas N. Kipf and Max Welling. 'Variational Graph Auto-Encoders'. In: *CoRR* abs/1611.07308 (2016).
[111] Thomas N. Kipf and Max Welling. 'Semi-Supervised Classification with Graph Convolutional Networks'. In: *ICLR*. 2017.
[112] Johannes Klicpera, Aleksandar Bojchevski and Stephan Günnemann. 'Predict then Propagate: Graph Neural Networks meet Personalized PageRank'. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. 2019.
[113] Pang Wei Koh and Percy Liang. 'Understanding black-box predictions via influence functions'. In: *ICML*. PMLR. 2017, pp. 1885–1894.
[114] Emmanouil Krasanakis, Symeon Papadopoulos and Ioannis Kompatsiaris. 'Applying Fairness Constraints on Graph Node Ranks Under Personalization Bias'. In: *International Conference on Complex Networks and Their Applications*. Springer. 2020, pp. 610–622.
[115] Matt J Kusner et al. 'Counterfactual fairness'. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4066–4076.
[116] Chanhee Kwak et al. 'Let machines unlearn–machine unlearning and the right to be forgotten'. In: (2017).
[117] Youngchun Kwon et al. 'Uncertainty-aware prediction of chemical reaction yields with graph neural networks'. In: *Journal of Cheminformatics* (2022).
[118] Preethi L., Krishna P. G. and Gerhard W. 'Operationalizing Individual Fairness with Pairwise Fair Representations'. In: *Proc. VLDB Endow.* 13.4 (2019), pp. 506–518.
[119] Charlotte Laclau et al. 'All of the Fairness for Edge Prediction with Optimal Transport'. In: *Proceedings of the 24th AISTATS (AISTATS '21)*. Vol. 130. PMLR. 2021, pp. 1774–1782.
[120] Preethi Lahoti, Krishna P. Gummadi and Gerhard Weikum. 'iFair: Learning Individually Fair Data Representations for Algorithmic Decision Making'. In: *ICDE*. 2019.
[121] Jure Leskovec and Julian J Mcauley. 'Learning to discover social circles in ego networks'. In: *NeurIPS*. Vol. 25. 2012.

[122] Ron Levie et al. ‘Cayleynets: Graph convolutional neural networks with complex rational spectral filters’. In: *IEEE Trans. Signal Process.* (2018).
[123] Kaiyang Li et al. ‘Adversarial privacy-preserving graph embedding against inference attack’. In: *IEEE Internet Things J.* (2020).
[124] Michelle M Li, Kexin Huang and Marinka Zitnik. ‘Graph representation learning in biomedicine and healthcare’. In: *Nat. Biomed. Eng.* (2022), pp. 1–17.
[125] Peizhao Li et al. ‘On Dyadic Fairness: Exploring and Mitigating Bias in Graph Connections’. In: *ICLR*. OpenReview.net, 2021.
[126] Frederick Liu and Besim Avci. ‘Incorporating priors with feature attribution on text classification’. In: *arXiv preprint arXiv:1906.08286* (2019).
[127] Guiliang Liu et al. ‘Learning Tree Interpretation from Object Representation for Deep Reinforcement Learning’. In: *NeurIPS* (2021).
[128] Ninghao Liu, Qizhang Feng and Xia Hu. ‘Interpretability in Graph Neural Networks’. In: *Graph Neural Networks: Foundations, Frontiers, and Applications* (2022).
[129] Yang Liu et al. ‘Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection’. In: *Proceedings of the 31th WWW (WWW ’21)*. 2021, pp. 3168–3177.
[130] Christos Louizos et al. ‘The variational fair autoencoder’. In: *arXiv preprint arXiv:1511.00830* (2015).
[131] Donald Loveland et al. ‘FairEdit: Preserving Fairness in Graph Neural Networks through Greedy Graph Editing’. In: *arXiv preprint arXiv:2201.03681* (2022).
[132] Dongsheng Luo et al. ‘Parameterized explainer for graph neural network’. In: *NeurIPS*. 2020, pp. 19620–19631.
[133] Ninareh M. et al. ‘A survey on bias and fairness in machine learning’. In: *arXiv preprint arXiv:1908.09635* (2019).
[134] Ninareh M. et al. ‘A survey on bias and fairness in machine learning’. In: *CSUR* 54.6 (2021), pp. 1–35.
[135] Jiaqi Ma, Junwei Deng and Qiaozhu Mei. ‘Subgroup generalization and fairness of graph neural networks’. In: *NeurIPS*. 2021, pp. 1048–1061.
[136] Jiaqi Ma, Shuangrui Ding and Qiaozhu Mei. ‘Towards more practical adversarial attacks on graph neural networks’. In: *NeurIPS* 33 (2020), pp. 4756–4766.
[137] Jing Ma et al. ‘Assessing the Causal Impact of COVID-19 Related Policies on Outbreak Dynamics: A Case Study in the US’. In: *WWW*. 2022.
[138] Jing Ma et al. ‘Learning Fair Node Representations with Graph Counterfactual Fairness’. In: *WSDM*. 2022, pp. 695–703.
[139] Ananth Mahadevan and Michael Mathioudakis. ‘Certifiable machine unlearning for linear models’. In: *arXiv preprint arXiv:2106.15093* (2021).
[140] Paul Mangold et al. ‘Differential Privacy has Bounded Impact on Fairness in Classification’. In: *arXiv preprint arXiv:2210.16242* (2022).
[141] Farzan Masrour et al. ‘Bursting the Filter Bubble: Fairness-Aware Network Link Prediction’. In: *AAAI Conference on Artificial Intelligence*. Vol. 34. 01. 2020, pp. 841–848.
[142] Shira Mitchell et al. ‘Algorithmic fairness: Choices, assumptions, and definitions’. In: *Annu. Rev. Stat. Appl.* 8 (2021), pp. 141–163.
[143] Jiaming Mu et al. ‘A hard label black-box adversarial attack against graph neural networks’. In: *SIGSAC*. 2021, pp. 108–125.

[144] Nicolo Navarin, Luca Oneto and Michele Donini. 'Learning deep fair graph neural networks'. In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. 2020, pp. 31–36.
[145] Thanh Tam Nguyen et al. 'A survey of machine unlearning'. In: *arXiv preprint arXiv:2209.02299* (2022).
[146] Iyiola E Olatunji, Wolfgang Nejdl and Megha Khosla. 'Membership inference attack on graph neural networks'. In: *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE. 2021, pp. 11–20.
[147] John Palowitch and Bryan Perozzi. 'MONET: Debiasing Graph Embeddings via the Metadata-Orthogonal Training Unit'. In: *CoRR* abs/1909.11793 (2019). URL: http://arxiv.org/abs/1909.11793.
[148] John Palowitch and Bryan Perozzi. 'Debiasing Graph Representations via Metadata-Orthogonal Training'. In: *ASONAM* (2020).
[149] Chao Pan, Eli Chien and Olgica Milenkovic. 'Unlearning graph classifiers with limited data resources'. In: *Proceedings of the ACM Web Conference 2023*. 2023, pp. 716–726.
[150] Shirui Pan et al. 'Adversarially regularized graph autoencoder for graph embedding'. In: *arXiv preprint arXiv:1802.04407* (2018).
[151] Stuart L Pardau. 'The california consumer privacy act: Towards a european-style privacy regime in the united states'. In: *J. Tech. L. & Pol'y* 23 (2018), p. 68.
[152] Eli Pariser. *The filter bubble: How the new personalized web is changing what we read and how we think*. Penguin, 2011.
[153] Hogun Park and Jennifer Neville. 'Role Equivalence Attention for Label Propagation in Graph Neural Networks'. In: *PAKDD*. 2020.
[154] Adam Paszke et al. 'Automatic differentiation in pytorch'. In: (2017).
[155] Dana Pessach and Erez Shmueli. 'Algorithmic fairness'. In: *arXiv preprint arXiv:2001.09784* (2020).
[156] Andrija Petrović et al. 'FAIR: Fair adversarial instance re-weighting'. In: *Neurocomputing* (2022).
[157] Geoff Pleiss et al. 'On fairness and calibration'. In: *arXiv preprint arXiv:1709.02012* (2017).
[158] Gregory Plumb et al. 'Regularizing black-box models for improved interpretability'. In: *arXiv preprint arXiv:1902.06787* (2019).
[159] Tahereh Pourhabibi et al. 'Fraud detection: A systematic literature review of graph-based anomaly detection approaches'. In: *Decision Support Systems* (2020).
[160] Yongye Qian et al. 'Interaction graph neural network for news recommendation'. In: *International Conference on Web Information Systems Engineering*. Springer. 2020, pp. 599–614.
[161] C Quoc and Viet Le. 'Learning to rank with nonsmooth cost functions'. In: *NeurIPS* (2007).
[162] Tahleen Rahman et al. 'Fairwalk: Towards Fair Graph Embedding'. In: *IJCAI*. 2019, pp. 3289–3295.
[163] General Data Protection Regulation. 'General data protection regulation (GDPR)'. In: *Intersoft Consulting, Accessed in October* 24.1 (2018).

[164] Andrew Slavin Ross, Michael C Hughes and Finale Doshi-Velez. 'Right for the right reasons: Training differentiable models by constraining their explanations'. In: *arXiv preprint arXiv:1703.03717* (2017).
[165] Anian Ruoss et al. 'Learning certified individually fair representations'. In: *NeurIPS* 33 (2020), pp. 7584–7596.
[166] Pritam Saha et al. 'GraphCovidNet: A graph neural network based model for detecting COVID-19 from CT scans and X-rays of chest'. In: *Scientific Reports* 11.1 (2021), pp. 1–16.
[167] Anwar Said et al. 'A Survey of Graph Unlearning'. In: *preprint arXiv:2310.02164* (2023).
[168] Wojciech Samek, Thomas Wiegand and Klaus-Robert Müller. 'Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models'. In: *arXiv preprint arXiv:1708.08296* (2017).
[169] Alvaro Sanchez-Gonzalez et al. 'Learning to simulate complex physics with graph networks'. In: *ICML*. 2020.
[170] Michael Sejr Schlichtkrull, Nicola De Cao and Ivan Titov. 'Interpreting graph neural networks for nlp with differentiable edge masking'. In: *arXiv preprint arXiv:2010.00577* (2020).
[171] Jan Schuchardt et al. 'Collective robustness certificates: Exploiting interdependence in graph neural networks'. In: *ICLR*. 2020.
[172] Ayush Sekhari et al. 'Remember what you want to forget: Algorithms for machine unlearning'. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 18075–18086.
[173] Prithviraj Sen et al. 'Collective Classification in Network Data'. In: *AI Mag.* 29.3 (2008), pp. 93–106.
[174] O. Shchur et al. 'Pitfalls of graph neural network evaluation'. In: *arXiv preprint arXiv:1811.05868* (2018).
[175] Chence Shi et al. 'A graph to graphs framework for retrosynthesis prediction'. In: *ICML*. 2020.
[176] Valentina Shumovskaia et al. 'Linking bank clients using graph neural networks powered by rich transactional data'. In: *DSAA*. Springer, 2020, pp. 1–11.
[177] Weihao Song et al. 'GUIDE: Group Equality Informed Individual Fairness in Graph Neural Networks'. In: *KDD*. 2022, pp. 1625–1634.
[178] Weiping Song et al. 'Session-based social recommendation via dynamic graph attention networks'. In: *WSDM*. 2019, pp. 555–563.
[179] Indro Spinelli et al. 'Biased Edge Dropout for Enhancing Fairness in Graph Representation Learning'. In: *arXiv preprint arXiv:2104.14210* 3.3 (2021), pp. 344–354.
[180] Yiwei Sun et al. 'Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach'. In: *TheWebConf*. 2020.
[181] Susheel Suresh et al. 'Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns'. In: *arXiv preprint arXiv:2106.06586* (2021).
[182] Lubos Takac and Michal Zabovsky. 'Data analysis in public social networks'. In: *Internation. scient. workshop*. Vol. 1. 6. 2012.
[183] Jie Tang et al. 'Arnetminer: extraction and mining of academic social networks'. In: *SIGKDD*. 2008, pp. 990–998.

[184] Lei Tang and Huan Liu. 'Relational learning via latent social dimensions'. In: *SIGKDD*. 2009.
[185] Xianfeng Tang et al. 'Investigating and Mitigating Degree-Related Biases in Graph Convoltuional Networks'. In: *Proceedings of the 29th CIKM (CIKM '20)*. 2020, pp. 1435–1444.
[186] Anvith Thudi et al. 'Unrolling sgd: Understanding factors influencing machine unlearning'. In: *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2022, pp. 303–319.
[187] Amanda L Traud, Peter J Mucha and Mason A Porter. 'Social structure of Facebook networks'. In: *Physica A: Statistical Mechanics and its Applications* 391.16 (2012), pp. 4165–4180.
[188] Petar Veličković et al. 'Graph attention networks'. In: *ICLR*. 2018.
[189] Cédric Villani. *Optimal transport: old and new*. Vol. 338. Springer Science & Business Media, 2008.
[190] Cedric Villani. *Topics in optimal transportation*. Vol. 58. American Mathematical Soc., 2021.
[191] Minh N Vu and My T Thai. 'Pgm-explainer: Probabilistic graphical model explanations for graph neural networks'. In: *arXiv preprint arXiv:2010.05788* (2020).
[192] Binghui Wang et al. 'Certified robustness of graph neural networks against adversarial structural perturbation'. In: *SIGKDD*. 2021.
[193] Daixin Wang et al. 'A semi-supervised graph attentive network for financial fraud detection'. In: *ICDM*. IEEE. 2019, pp. 598–607.
[194] Daixin Wang et al. 'Temporal-Aware Graph Neural Network for Credit Risk Prediction'. In: *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. SIAM. 2021, pp. 702–710.
[195] Hao Wang, Berk Ustun and Flavio Calmon. 'Repairing without retraining: Avoiding disparate impact with counterfactual distributions'. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6618–6627.
[196] Haonan Wang, Ziwei Wu and Jingrui He. 'Training Fair Deep Neural Networks by Balancing Influence'. In: *arXiv preprint arXiv:2201.05759* (2022).
[197] Xiang Wang et al. 'Causal Screening to Interpret Graph Neural Networks'. In: (2020).
[198] Xiaoyun Wang, Xuanqing Liu and Cho-Jui Hsieh. 'Graphdefense: Towards robust graph convolutional networks'. In: *arXiv preprint arXiv:1911.04429* (2019).
[199] Yaojing Wang et al. 'Bringing Order to Network Embedding: A Relative Ranking based Approach'. In: *Proceedings of the 29th CIKM (CIKM '20)*. 2020.
[200] Yu Wang and Tyler Derr. 'Tree decomposed graph neural network'. In: *CIKM*. 2021.
[201] Yu Wang et al. 'Improving Fairness in Graph Neural Networks via Mitigating Sensitive Attribute Leakage'. In: *SIGKDD*. 2022.
[202] Alexander Warnecke et al. 'Machine unlearning of features and labels'. In: *preprint arXiv:2108.11577* (2021).
[203] Yanhao Wei et al. 'Credit scoring with social network data'. In: *Marketing Science* 35.2 (2016), pp. 234–258.
[204] Bang Wu et al. 'Adapting membership inference attacks to GNN for graph classification: Approaches and implications'. In: *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2021, pp. 1421–1426.

[205] Felix Wu et al. ‘Simplifying graph convolutional networks’. In: *ICML*. PMLR. 2019, pp. 6861–6871.
[206] Huijun Wu et al. ‘Adversarial Examples for Graph Data: Deep Insights into Attack and Defense’. In: *IJCAI*. 2019, pp. 4816–4823.
[207] Huijun Wu et al. ‘Adversarial examples on graph data: Deep insights into attack and defense’. In: *arXiv preprint arXiv:1903.01610* (2019).
[208] Jiancan Wu et al. ‘GIF: A General Graph Unlearning Strategy via Influence Function’. In: *Proceedings of the ACM Web Conference 2023*. 2023, pp. 651–661.
[209] Kun Wu et al. ‘Certified edge unlearning for graph neural networks’. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 2606–2617.
[210] Lingfei Wu et al. ‘Graph neural networks: foundation, frontiers and applications’. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 4840–4841.
[211] Shu Wu et al. ‘Session-based recommendation with graph neural networks’. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 346–353.
[212] Xiang Wu et al. ‘Learning an evolutionary embedding via massive knowledge distillation’. In: *International Journal of Computer Vision* (2020).
[213] Yongkai Wu, Lu Zhang and Xintao Wu. ‘Counterfactual Fairness: Unidentification, Bound and Algorithm’. In: *International Joint Conference on Artificial Intelligence*. 2019, pp. 1438–1444.
[214] Zonghan Wu et al. ‘A comprehensive survey on graph neural networks’. In: *IEEE Trans Neural Netw Learn Syst* 32.1 (2020), pp. 4–24.
[215] Han Xie et al. ‘Federated graph classification over non-iid graphs’. In: *NeurIPS* (2021).
[216] Bingbing Xu et al. ‘Towards Consumer Loan Fraud Detection: Graph Neural Networks with Role-Constrained Conditional Random Field’. In: *AAAI*. 2021, pp. 4537–4545.
[217] Depeng Xu et al. ‘DPNE: Differentially private network embedding’. In: *PAKDD*. 2018.
[218] Feiyu Xu et al. ‘Explainable AI: A brief survey on history, research areas, approaches and challenges’. In: *NLPCC*. 2019.
[219] Heng Xu et al. ‘Machine unlearning: A survey’. In: *ACM Computing Surveys* 56.1 (2023), pp. 1–36.
[220] Kaidi Xu et al. ‘Topology attack and defense for graph neural networks: An optimization perspective’. In: *arXiv preprint arXiv:1906.04214* (2019).
[221] Keyulu Xu et al. ‘Representation learning on graphs with jumping knowledge networks’. In: *ICML*. 2018.
[222] Keyulu Xu et al. ‘How Powerful are Graph Neural Networks?’ In: *ICLR*. OpenReview.net, 2019.
[223] Xiaojun Xu et al. ‘Characterizing malicious edges targeting on graph neural networks’. In: *OpenReview* (2018).
[224] Bobby Yan, Skyler Seto and Nicholas Apostoloff. ‘FORML: Learning to Reweight Data for Fairness’. In: *arXiv preprint arXiv:2202.01719* (2022).
[225] Cheng Yang, Jiawei Liu and Chuan Shi. ‘Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework’. In: *WWW*. 2021.

[226] Zhiju Yang et al. 'WTAGRAPH: Web Tracking and Advertising Detection using Graph Neural Networks'. In: *IEEE Symposium on Security and Privacy*. 2022.
[227] I-Cheng Yeh and Che-hui Lien. 'The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients'. In: *Expert Syst. Appl.* 36.2 (2009), pp. 2473–2480.
[228] Rex Ying et al. 'Gnnexplainer: Generating explanations for graph neural networks'. In: *NeurIPS*. 2019.
[229] Hao Yuan et al. 'Explainability in graph neural networks: A taxonomic survey'. In: *arXiv preprint arXiv:2012.15445* (2020).
[230] Hao Yuan et al. 'XGNN: Towards Model-Level Explanations of Graph Neural Networks'. In: *SIGKDD*. KDD '20. Virtual Event, CA, USA: Association for Computing Machinery, 2020, pp. 430–438. ISBN: 9781450379984. DOI: 10.1145/3394486.3403085. URL: https://doi.org/10.1145/3394486.3403085.
[231] Hao Yuan et al. 'On explainability of graph neural networks via subgraph explorations'. In: *arXiv preprint arXiv:2102.05152* (2021).
[232] Muhammad Bilal Zafar et al. 'From parity to preference-based notions of fairness in classification'. In: *Proceedings of the 31st NeurIPS (NIPS '17)*. 2017, pp. 229–239.
[233] Rich Zemel et al. 'Learning Fair Representations'. In: *ICML*. 2013, pp. 325–333.
[234] Muhan Zhang and Yixin Chen. 'Link Prediction Based on Graph Neural Networks'. In: *Proceedings of the 32nd NeurIPS (NIPS '18)*. 2018, pp. 5171–5181.
[235] Quanshi Zhang et al. 'Interpreting cnns via decision trees'. In: *CVPR*. 2019.
[236] Shaofeng Zhang et al. 'DOTIN: Dropping Task-Irrelevant Nodes for GNNs'. In: *arXiv preprint arXiv:2204.13429* (2022).
[237] Wenbin Zhang et al. 'Fairness Amidst Non-IID Graph Data: A Literature Review'. In: *arXiv preprint arXiv:2202.07170* (2022).
[238] Xiang Zhang and Marinka Zitnik. 'Gnnguard: Defending graph neural networks against adversarial attacks'. In: *NeurIPS* (2020).
[239] Xiao-Meng Zhang et al. 'Graph neural networks and their current applications in bioinformatics'. In: *Frontiers in genetics* 12 (2021), p. 690049.
[240] Xu Zhang et al. 'A Multi-view Confidence-calibrated Framework for Fair and Stable Graph Representation Learning'. In: *ICDM*. IEEE. 2021, pp. 1493–1498.
[241] Zijie Zhang et al. 'Prompt certified machine unlearning with randomized gradient smoothing and quantization'. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 13433–13455.
[242] Jieyu Zhao et al. 'Men also like shopping: Reducing gender bias amplification using corpus-level constraints'. In: *arXiv preprint arXiv:1707.09457* (2017).
[243] Chuanpan Zheng et al. 'Gman: A graph multi-attention network for traffic prediction'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 01. 2020, pp. 1234–1241.
[244] Wenyue Zheng et al. 'Graph Unlearning Using Knowledge Distillation'. In: *International Conference on Information and Communications Security*. Springer. 2023, pp. 485–501.
[245] Jie Zhou et al. 'Graph neural networks: A review of methods and applications'. In: *arXiv preprint arXiv:1812.08434* 1 (2018), pp. 57–81.

[246] Marinka Zitnik, Monica Agrawal and Jure Leskovec. ‘Modeling polypharmacy side effects with graph convolutional networks’. In: *Bioinformatics* 34.13 (2018), pp. i457–i466.
[247] Daniel Zügner and Stephan Günnemann. ‘Certifiable robustness and robust training for graph convolutional networks’. In: *SIGKDD*. 2019.
[248] Daniel Zügner and Stephan Günnemann. ‘Certifiable robustness of graph convolutional networks under structure perturbations’. In: *SIGKDD*. 2020.
[249] Daniel Zügner et al. ‘Adversarial attacks on graph neural networks: Perturbations and their patterns’. In: *TKDD* (2020).