Trade less Accuracy for Fairness and Trade-off Explanation for GNN

Yazheng Liu*, Xi Zhang*, Sihong Xie†
*Key Laboratory of Trustworthy Distributed Computing and Service (MoE), BUPT †Computer Science and Engineering Dept, Lehigh University
{liuyz,zhangx}@bupt.edu.cn, xiesihong1@gmail.com

***Abstract*—Graphs are widely found in social network analysis and e-commerce, where Graph Neural Networks (GNNs) are the state-of the-art model. GNNs can be biased due to sensitive attributes and network topology. With existing work that learns a fair node representation or adjacency matrix, achieving a strong guarantee of group fairness while preserving prediction accuracy is still challenging, with the fairness-accuracy trade-off remaining obscure to human decision-makers. We first define and analyze a novel upper bound of group fairness to optimize the adjacency matrix for fairness without significantly harming prediction accuracy. To understand the nuance of fairness-accuracy trade-off, we further propose macroscopic and microscopic explanation methods to reveal the trade-offs and the space that one can exploit. The macroscopic explanation method is based on stratified sampling and linear programming to deterministically explain the dynamics of the group fairness and prediction accuracy. Driving down to the microscopic level, we propose a path-based explanation that reveals how network topology leads to the trade-off. On seven graph datasets, we demonstrate the novel upper bound can achieve more efficient fairness-accuracy trade-offs and the intuitiveness of the explanation methods can clearly pinpoint where the trade-off is improved.**

I. Introduction

Nowadays, Graph Neural Networks (GNNs) have been playing a pivotal role in many machine learning applications, such as molecule property prediction [36], fraud detection [17], [35], social network analysis [16] and recommendation systems [39], formulated as node classification, link prediction, and graph classification [6], [11], [16], [33]. GNNs achieve the state-of-the-art predictive performance by adopting message-passing to the aggregate edge, node, and graph topology information of the local neighborhood in multiple layers.

Despite their superior performance, GNNs inherit or even amplify bias in the input graph data [8], limiting its applications that involve critical decisions about humans and organizations. The unfairness is caused by node sensitive attributes or biased topological properties such as node degree. Specifically, we focus on the degree-related group fairness for the node classification task. We put nodes of low degree in the protected group (indicated by the sensitive attribute $S = 1$), and the remaining nodes in the favored group ($S = 0$). The degree distributions of most real-world graphs follow power law [1], [7], [9], [32], and GNNs heavily rely on message aggregation so that low-degree nodes receive fewer messages than the high-degree nodes, resulting in disparate performance between the two groups [4], [32] and violation of fairness criteria, such as equal opportunity [12].

To address the fairness issues, most existing works learn a fair node or graph representation by optimizing adjacency matrices [18], [31], adversarial training [8], or fairness-oriented DeepWalk [15], [24]. One challenge is that prediction fairness and accuracy can be competing with each other, and improving one metric can hurt the other. While multi-objective optimization techniques have been proposed to explore the model parameters space and optimal trade-offs of the two metrics, only existing group fairness loss functions are used [4]. It remains unclear if a better fairness loss function can be designed to find more efficient trade-offs, i.e., improving fairness while hurting prediction accuracy less. If the answer to the question is affirmative, then a follow-up question is what leads to the *difference* in the trade-off. Existing explainable machine learning techniques can attribute the GNN predictions to node attributes or edge connections of the input graph [27], [38], but attributing the trade-offs between two metrics to the input graph as an explanation is a less studied problem.

To address the above challenges, in this work, we study a novel problem of learning fair graph neural networks with limited sensitive information. The classifier should maintain high accuracy while satisfying fairness. Besides, to understand what leads to better trade-offs, we propose the macroscopic method to select the critical nodes to explain the difference between trade-offs. See Figure 1 for details. The contributions of this work can be summarized as follows:

- We propose a novel upper bound of group fairness to train the adjacency matrix to mitigate the inequal opportunity due to degree-related biases on GNNs. The loss leads to more fairness with less harm to the accuracy.
- We formulate a novel optimization problem to find the critical nodes that can explain the difference between fairness and accuracy.
- Extensive experiments on seven datasets for node classification demonstrate our method can not only ensure fairness but also classification performance. Besides, the proposed explanation approach outperforms state-of-the-art baselines.

II. Related Work

A. Fair machine learning on graphs

Many works have been conducted to deal with the bias in the training data to achieve fairness in machine learning [12], [14], [19], [43]. However, recently, with the success of GNNs,

978-1-6654-8045-1/22/$31.00 ©2022 IEEE

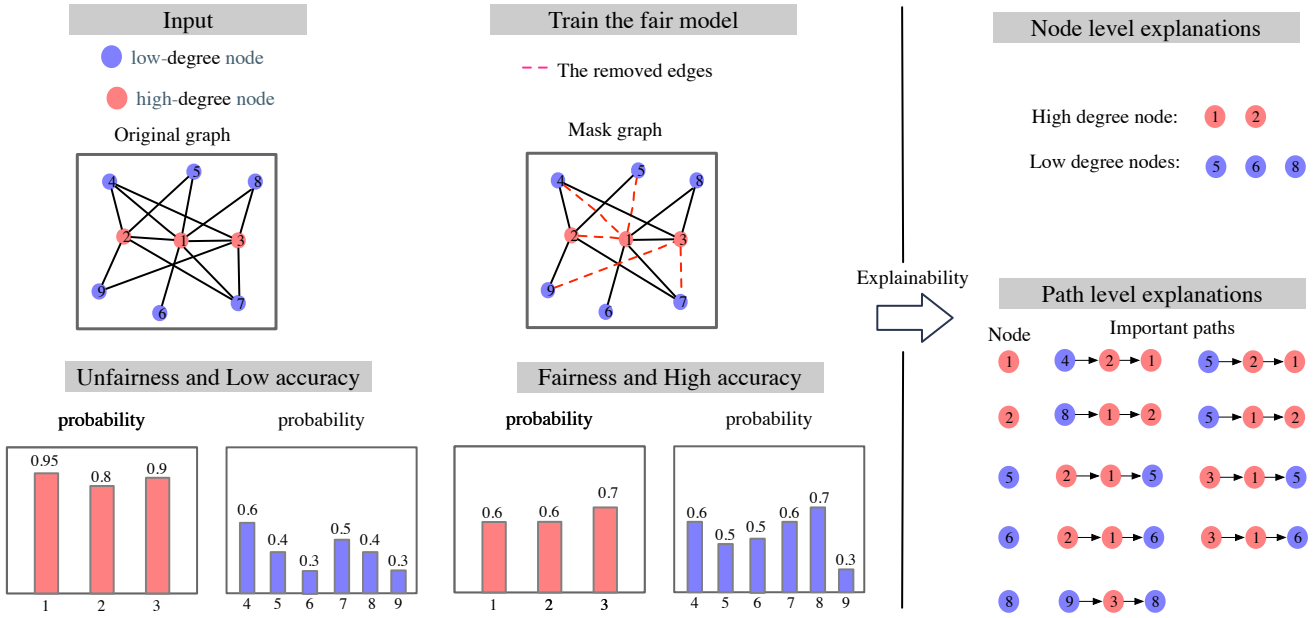

Fig. 1: The overall workflow. We adopt the GNN model for the node classification task where nodes are in protected (blue) and favored (red) groups. As an example, disparate impact concerns about the discrepancy in the predicted probabilities of nodes from the two groups. We design a novel fairness loss function to train a mask matrix for more efficient fairness-accuracy trade-offs. Finally, to understand what lead to the better trade-offs, we propose a macroscopic method to select representative nodes to explain the difference between two trade-offs. For the selected nodes, we propose a microscopic method to find a small set of paths that explain the computations that lead to the difference.

researchers started to investigate the fairness of GNNs. Due to the importance of representation learning to downstream tasks such as link prediction and node classification, some works focused on learning fair node embeddings [3], [5], [22]. Some of these approaches achieve fairness by adversarial learning. Compositional fairness constraints [3] learned a set of adversarial filters that remove dependencies on sensitive attributes. In [8], a method that learns fair GNNs with limited sensitive attribute information via adversarial attacks is proposed. For fair link prediction [22], they employed adversarial learning to ensure that inter-group links are well-represented among the predicted links. [24] proposed Fairwalk, the random walk based graph embedding method that revises the transition probability according to the vertex's sensitive attributes. [15] developed Crosswalk, the key idea is to bias random walks to cross group boundaries, which enhances fairness. In [18], [31], they proposed to learn a fair adjacency matrix that respects graph structural constraints and preserves predictive accuracy. In [32], they proposed a self-supervised learning method that mitigates the degree-related biases of GNNs.

TABLE I: Notation

Symbols	Definitions and Descriptions
J, U, V, K	Nodes in a graph
j, u, v, k	Neurons of the nodes in upper-case
N	Number of nodes in a graph
n	Index of nodes in the graph
C	The number of classes
c	A specific class
S	A sensitive attribute
$\mathcal{V}_{s,c}$	Set of nodes with $S = s$ in class c
H_0, H_1	The set of probabilities that the nodes in $\mathcal{V}_{s,c}$ are predicted to be class c in the graph G_0 and G_1
B_0, B_1	Divide the data in H_τ into the buckets. $\tau = 0, 1$, $B_\tau[l]$ denotes the l-th bucket of the histogram of H_τ
$\mathcal{V}^*_{s,c}$	The set of important nodes to explain the fairness
F	The data flow matrix.
$\mathbf{z}_J(G)$	The logits $\mathbf{z}_j(G)$, $j \in [1, \ldots, C]$, of node J
$\Delta\mathbf{z}_J(G_0, G_1)$	$\Delta\mathbf{z}_J(G_0, G_1) = \mathbf{z}_J(G_1) - \mathbf{z}_J(G_0)$
$\mathsf{Pr}_J(G)$	Class distribution of J, with elements $\mathsf{Pr}_j(G)$
$W(G)$	Paths on the computation graph of GNN
$W_J(G)$	The subset of $W(G)$ that computes $\mathsf{Pr}_J(G)$
$\Delta W_J(G_0, G_1)$	Altered paths in $W_J(G_0)$ as $G_0 \to G_1$
$C_{p,j}$	Contribution of the p-th altered path to Δz_j

B. Explanations for graph neural networks

Graph neural networks (GNNs) and their explainability are experiencing rapid developments. In the survey [41], existing GNN explanation approaches are categorized as instance-level and model-level methods. The instance-level category includes gradient/features-based methods [2], [23], perturbation-based methods [21], [26], [38], [42], decomposition-based methods [2], [27], [28], and surrogate-based methods [13], [34]. In the model-level category, [40] proposed to explain the GNNs via reinforcement learning, and the Monte Carlo search is used for exploration. Our method is similar to that in [38], where they learned a soft mask over the edges to explain the prediction of graph neural networks, while we propose a new fairness loss function that drives the optimization of the mask and shows more efficient accuracy-fairness trade-offs. In [27], a GNN prediction is attributed to the important path on the input graph, while we attribute the difference in the trade-offs to paths.

III. PRELIMINARIES

Graph neural networks. We use $G = (\mathcal{V}, \mathcal{E})$ to denote an attributed graph, where $\mathcal{V}$ is the set of a total of N nodes of $\mathcal{G}$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. A is the adjacency matrix of the graph G, where $A_{ij} = 1$ if nodes I and J are connected; otherwise, $A_{ij} = 0$. Let $\mathcal{N}(J) = \{I | (I, J) \text{ or } (J, I) \in \mathcal{E}\}$ be the neighbors of node J. Upper-case letters I,J,...,K represent certain nodes in the graph. For each node, $y \in \{1, \ldots, C\}$ is the label of the node.

We train a GNN of T layers that predicts the class distribution of each node. On layer t, $t = 1, \ldots, T$ and for node J, GNN computes hidden vector $\mathbf{h}_J^{(t)}$ using messages sent from its neighbors:

$$\mathbf{a}_J^{(t)} = f_{\text{AGG}}^{(t)}(\mathbf{h}_J^{(t-1)}, \mathbf{h}_K^{(t-1)}, K \in \mathcal{N}(J)) \quad (1)$$

$$\mathbf{z}_J^{(t)} = f_{\text{UPDATE}}^{(t)}(\mathbf{a}_J^{(t)}), \quad (2)$$

$$\mathbf{h}_J^{(t)} = \text{NOLINEAR}(\mathbf{z}_J^{(t)}), \quad (3)$$

where $f_{\text{AGG}}^{(t)}$ aggregates the messages from all neighbors and can be the element-wise sum, average, or maximum of the incoming messages. $f_{\text{UPDATE}}^{(t)}$ maps $\mathbf{a}_J^{(t)}$ to $\mathbf{z}_J^{(t)}$, using $\mathbf{z}_J^{(t)} = \left\langle \mathbf{a}_J^{(t)}, \boldsymbol{\theta}^{(t)} \right\rangle$ or a multi-layered perceptron with parameters $\boldsymbol{\theta}^{(t)}$. Let $\boldsymbol{\theta} = [\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(T)}] \in \mathbb{R}^d$ denote all the d trainable parameters of the GNN. At the input layer, $\mathbf{h}_J^{(0)}$ is the node feature vector $\mathbf{x}_J$. At layer T, the logits are $\mathbf{z}_J^{(T)} \triangleq \mathbf{z}_J(G) = [z_1(G), z_2(G), \ldots, z_C(G)]$. $\mathbf{z}_J(G)$ is mapped to the class distribution $\mathsf{Pr}_J(G)$ through softmax. The model can be trained by minimizing the loss function w.r.t. $\boldsymbol{\theta}$:

$$\mathcal{L}_C(\boldsymbol{\theta}) = -\frac{1}{|\mathcal{V}_L|} \sum_{n \in \mathcal{V}_L} \left[y_n \log \hat{y}_n + (1 - y_n) \log (1 - \hat{y}_n) \right], \quad (4)$$

where $\mathcal{V}_L$ is the set of labeled nodes of G.

Measuring fairness of predictions. For each node, we let the sensitive attribute S indicate if the node has the high ($S = 0$) or low ($S = 1$) degree. The nodes with the same value of S are grouped and we obtain the favored ($S = 0$) and protected ($S = 1$) group, respectively. Equal opportunity [12] requires the equal probability of true positives and false positives between the two groups defined by the protected attribute S. Let $\hat{y} \in \{1, \ldots, C\}$ denotes the prediction of the classifier. Multi-class classification problems are less studied in fair ML literature. Because of the multi-class prediction of node, consider the specific class c, the property of equal opportunity is defined as:

$$P(\hat{y} = c \mid S = 1, y = c) = P(\hat{y} = c \mid S = 0, y = c) \quad (5)$$

According to [20], and to facilitate gradient based optimization, we adopt the following surrogate loss function for EO for the class c [10]:

$$\mathcal{L}_{\text{EO}}^c = |\Pr(\hat{y} = c \mid S = 0, y = c) - \Pr(\hat{y} = c \mid S = 1, y = c)| \quad (6)$$

where $\Pr(\hat{y} = c \mid S = s, y = c)$ is estimated as the percentage of the nodes with sensitive attribute s and label c classified as class c by $\hat{y}$. Since the GNN has probabilistic outputs, we approximate $\Pr(\hat{y} = c \mid S = s, y = c)$ using

$$\frac{\sum_{n=1}^N \mathbb{1}\left[S_n = s, y_n = c\right] \Pr\left(\hat{y}_n = c\right)}{\sum_{n=1}^N \mathbb{1}\left[S_n = s, y_n = c\right]} \quad (7)$$

The GNN trained on G can make biased predictions because its information aggregation mechanism depends on the neighborhood structure, and node degree, indicated by S, becomes a factor deciding the informativeness and accuracy of the aggregation. In particular, with more neighbors providing useful class information, high-degree nodes are more likely to be predicted accurately, leading to the violation of equal opportunity.

Problem definition. *Given the graph $G = (\mathcal{V}, \mathcal{E})$, defined the sensitive attribute S (related to the degree of the node), learn the fair GNN with the model parameters $\boldsymbol{\theta}$ and the mask matrix M for fair node classification. The classifier should maintain high accuracy whilst satisfying the fairness standard such as equal opportunity. After obtaining the fairness model, it is necessary to find important nodes to explain the transition of the fairness of the graph model. Then, for the node, find the critical walks to explain the changed probability.*

IV. METHOD

A. A novel fair loss for better accuracy-fairness trade-off

To make the GNN more balanced in the accuracy of nodes from two different groups, we train the mask matrix M to balance the discrepancy between the informativeness of the neighbors of nodes from two distinct demographic groups. A trivial solution is to prune the neighbors of high-degree nodes ($S = 0$), so that they become the low-degree nodes and thus they are treated equally by the GNN model as the low-degree nodes ($S = 1$). However, it is likely that the predictions of the class of high-degree nodes become less accurate due to the pruning of their neighbors. In other words, the accuracy is sacrificed for a high degree of equal opportunity.

While the $\mathcal{L}_{\text{EO}}^c$ measures the difference in means between the two groups, it ignores the difference in probability distributions. For example, [0.5, 0.5, 0.5, 0.5] and [0.8, 0.8, 0.2, 0.2] are the class probabilities of nodes in two groups. They have the same mean but different probability distributions and $\mathcal{L}_{\text{EO}}^c = 0$. Using $\mathcal{L}_{\text{EO}}^c$ to train the mask matrix M can lead to little or no improvement in fairness but bring down accuracy.

Therefore, for the specific class c, we define the fairness loss based on EO for class c:

$$\bar{\mathcal{L}}_{\text{EO}}^c = \frac{\sum_{I \in \mathcal{V}_{0,c}} \sum_{J \in \mathcal{V}_{1,c}} |\Pr\left(\hat{y}_I = c\right) - \Pr\left(\hat{y}_J = c\right)|}{|\mathcal{V}_{0,c}| \times |\mathcal{V}_{1,c}|} \quad (8)$$

where $\Pr\left(\hat{y}_I = c\right), \Pr\left(\hat{y}_J = c\right)$ denote the probability that node I, J is judged to be class c. $\Pr\left(\hat{y}_I = c\right) = \left(\hat{y}_I = c \mid G = A \odot \sigma(M)\right)$. $M \in \mathbb{R}^{\mathbb{N} \times \mathbb{N}}$ denotes the mask that we need to learn, $\odot$ denotes element-wise multiplication, and σ denotes the sigmoid that maps the mask to $[0, 1]^{N \times N}$. $|\mathcal{V}_{0,c}|$ and $|\mathcal{V}_{1,c}|$ denote the size of $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$.

Note that $\bar{\mathcal{L}}^c_{\text{EO}}$ is the upper bound of $\mathcal{L}^c_{\text{EO}}$:

$$\begin{aligned}\bar{\mathcal{L}}^c_{\text{EO}} &= \frac{\sum_{I\in\mathcal{V}_{0,c}}\sum_{J\in\mathcal{V}_{1,c}}|\Pr(\hat{y}_I=c)-\Pr(\hat{y}_J=c)|}{|\mathcal{V}_{0,c}|\times|\mathcal{V}_{1,c}|}\\ &\geq \frac{|\sum_{I\in\mathcal{V}_{0,c}}\sum_{J\in\mathcal{V}_{1,c}}(\Pr(\hat{y}_I=c)-\Pr(\hat{y}_J=c))|}{|\mathcal{V}_{0,c}|\times|\mathcal{V}_{1,c}|}\\ &= \left|\frac{\sum_{I\in\mathcal{V}_{0,c}}\Pr(\hat{y}_I=c)}{|\mathcal{V}_{0,c}|}-\frac{\sum_{J\in\mathcal{V}_{1,c}}\Pr(\hat{y}_J=c)}{|\mathcal{V}_{1,c}|}\right|\\ &= \mathcal{L}^c_{\text{EO}}\end{aligned}$$

Since nodes from different classes are in disjoint sets, we can enforce fairness in all classes independently by minimizing the following weighted sum of fairness losses from all classes:

$$\bar{\mathcal{L}}_{\text{EO}} = \sum_{c=1}^{C}\lambda^c\bar{\mathcal{L}}^c_{\text{EO}}, \tag{9}$$

where the weights λ^c, $c=1,\ldots,C$, determine the importance of the fairness within each class.

Overall, we can ensure and balance the accuracy and fairness of the prediction when training the adjacency matrix mask M using the following loss function:

$$\min_M \mathcal{L} = \mathcal{L}_C + \lambda\bar{\mathcal{L}}_{\text{EO}}. \tag{10}$$

λ is a hyperparameter that balances the trade-off between the two objectives. In the experiments, we will demonstrate that the upper-bounds of the EO fairness losses will lead to more efficient trade-offs, compared to that obtained using the fairness losses themselves.

B. Contrastive explanations of accuracy-fairness trade-offs.

It is important to understand the price that one has to pay for more fairness to aid model selection and design in applications. For example, a model user may be wondering, to improve the fairness metric defined in Eq. (6) by 5%, which nodes will benefit from fairer treatment while which sample will suffer from incorrect classification; a model designer would like to dive deeper into the graph topology to understand how the different adjacency matrix and local neighborhood lead to the different accuracy-fairness trade-offs. These questions can be answered by the following two novel explanation tasks and the corresponding methods.

1) Macroscopic node-level contrastive explanations: Since the training data can be large and it may be difficult for a human user to screen all samples and analyze their contributions to the change in the trade-offs, we design an optimization problem, whose optimal solution selects the most critical nodes that can best represent and approximate the change in the trade-offs. As a result, human users only need to understand the change through these critical nodes. The selected sample are "macroscopic explanations" since they do not attribute the change to the details of the GNN computation model.

Since EO can be measured by the means of the predicted probabilities of the nodes, the change in EO can be attributed to the shift of these probabilities. Explaining the shift of probabilities of a population has not been studied before

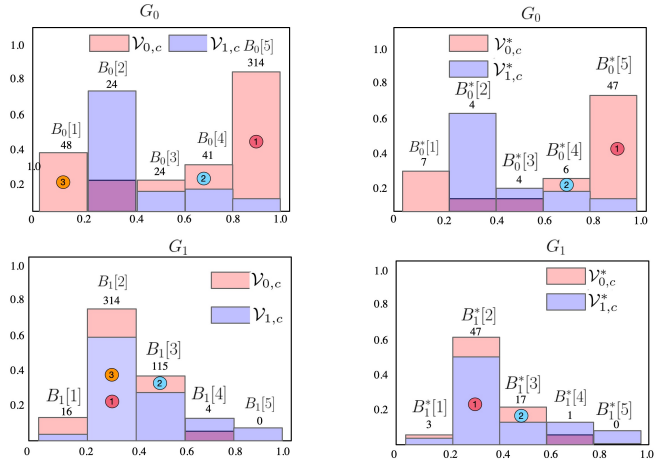

Fig. 2: Histograms about frequencies of predicted probabilities of a class c for nodes from two groups $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$. A solid circle in a bucket represents a example node classified with the corresponding class. Left column: as the graph G_0 got masked to G_1, predicted probability distributions are different and are fairer with equal opportunity. Right column: select node subsets $\mathcal{V}^*_{s,c}\subset\mathcal{V}_{s,c}$, $s=0,1$ to closely approximate (explain) the shift of the histograms of G_0 to those of G_1, and how fairness is increased.

in the explainable ML literature. We propose a node-based explanation that can preserve the shift. More broadly, the method belongs to the instance-based explanations [37] that select critical data instances to explain a phenomenon. In Fig. 2, the distribution of the predicted probabilities of nodes from group $\mathcal{V}_{s,c}$ obtained from GNN on G_0 (first column, top row) is shifted to that of the same group obtained on G_1 (first column, bottom row), and we aim to select the node subset $\mathcal{V}^*_{s,c}\subset\mathcal{V}_{s,c}$, as shown in the second column, that reproduces the distribution shift from top to bottom in the first column. The shift over each group $\mathcal{V}_{s,c}$ can be approximated independently, and we take a specific $\mathcal{V}_{s,c}$ to present an optimization problem for selecting the optimal $\mathcal{V}^*_{s,c}$.

Let H_τ be the set of probabilities for nodes in $\mathcal{V}_{s,c}$ predicted by the GNN model on G_τ, $\tau=0,1$. To make the resulting explanations more intuitive, we describe the distributions of the probabilities in H_τ by a histogram that discretizes the probabilities into a small number of buckets, each of which contains the nodes whose predicted class probabilities fall into the same interval. Let $B_\tau[l]$ denote the l-th bucket of the histogram of H_τ.

We aim to explain the shift of the histogram of H_0 to that of H_1, or more specifically, how the nodes are predicted differently due to the difference between two graphs G_0 and G_1. The difference in the probabilities is then described by the movement of nodes moving from a bucket $B_0[l]$ for G_0 to a bucket $B_1[r]$ for G_1. We track the movement using a data flow F, where $F_{l,r}$ is the set of nodes that move from $B_0[l]$ to $B_1[r]$ and $F_{l,r}[n]$ denotes the n-th node in $F_{l,r}$. Let $|B_\tau|$ denote the number of buckets in the histogram for H_τ, and for simplicity, we assume that $|B_\tau|$ is a constant $|B|$ for all τ. Similarly, $|B_0[l]|$, $|B_1[r]|$, and $|F_{l,r}|$ all denote the size of

the corresponding sets. We have the following properties:

- $\sum_{l=1}^{|B|} B_0[l] = \sum_{r=1}^{|B|} B_1[r] = |\mathcal{V}_{s,c}|$ (preserve all nodes)
- $\sum_{r=1}^{|B|} |F_{l,r}| = B_0[l]$ (preserve source nodes)
- $\sum_{l=1}^{|B|} |F_{l,r}| = B_1[r]$ (preserve destination nodes)

See the figure 2 for a demonstration.

As the original histograms can contain many nodes and tracking the flow F is non-trivial for human beings. We select $p\%$ of the original data from H_τ, denoted by H_τ^*, to represent the shift in distributions. Let $|\mathcal{V}_{0,c}^*| = |\mathcal{V}_{0,c}| \times \text{ratio}$. Randomly sampling nodes from H_0 or H_1 has two disadvantages: 1) there is randomness that is hampering human interpretation; 2) data from individual buckets in B_τ may not be well-represented, leading to misleading explanations.

To address these drawbacks, we propose the following linear program to deterministically and optimally select $p\%$ of the nodes that well-represent individual buckets B_τ and the shift in prediction probability distributions. Let the Select be the optimization variables indicating the selection of each node in each flow. In particular, $\text{Select}_{l,r}[n] \in [0,1]$ means the chance that the node indexed by $F_{l,r}[n]$ is selected to represent the shift from bucket $B_0[l]$ to $B_1[r]$. Intuitively, if we can select a small set of nodes that represent the individual flow $F_{l,r}$, the human users will be able to see how the changes in predicted probabilities lead to a fairer distribution between the favored and protected groups $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$ for class c.

Let B_τ^* be the buckets of selected nodes, and H_τ^* denote the histogram based on B_τ^*. we constrain $|B_0^*[l]| = p \times |B_0[l]|$ for $l = 1, \dots |B|$ and similarly for $|B_1^*[r]|$. These cardinality constraints are to improve the simplicity of the resulting distributions. We also want the simplified histogram of H_τ^* to be close to the original histogram of the original set of predicted probabilities H_τ, for $\tau = 0, 1$. We match the first-order moments (i.e., the means) of the original and simplified histograms $\textbf{Mean}(H_\tau^*) = \frac{\sum_{l=1}^{|B|}\sum_{r=1}^{|B|}\sum_{n=1}^{|F_{l,r}|}\text{Select}_{l,r}[n]\times \Pr_{F_{l,r}[n]}}{|\mathcal{V}_{s,c}^*|}$,

$$\min_{\text{Select}} \quad \sum_{\tau=0}^{1} |\textbf{Mean}(H_\tau) - \textbf{Mean}(H_\tau^*)| \tag{11}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{l=1}^{|B|}\sum_{n=1}^{|F_{l,r}|} \text{Select}_{l,r}[n] = B_1^*[r], \forall r = 1, \dots, |B|, \\ & \sum_{r=1}^{|B|}\sum_{n=1}^{|F_{l,r}|} \text{Select}_{l,r}[n] = B_0^*[l], \forall l = 1, \dots, |B|, \\ & \sum_{l=1}^{|B|} B_0^*[l] = \sum_{r=1}^{|B|} B_1^*[r] = |\mathcal{V}_{s,c}^*| \end{aligned} \tag{12}$$

By solving the linear programming problem, we obtain $\mathcal{V}_{\tau,c}^*$ for $\tau = 0, 1$ and $c = 1, \dots, C$. Because $\mathcal{V}_{\tau,c}^*$ can well-approximate $\mathcal{V}_{\tau,c}$ on both G_0 and G_1 for both groups and any class c, $\mathcal{V}_{\tau,c}^*$ can explain the change in fairness.

2) Microscopic path-based contrastive explanations: $G_0 = (\mathcal{V}, \mathcal{E})$, $G_1 = (\mathcal{V}, \mathcal{E}^{'})$, and $\mathcal{E}^{'}$ is obtained with some edges in $\mathcal{E}$ masked. Let $\Delta\mathcal{E} = \{e : e \in \mathcal{E} \wedge e \notin \mathcal{E}^{'}\}$ be the set of removed edges. Due to $\Delta\mathcal{E}$, some nodes will be classified with different class distributions that lead to a different accuracy-fairness trade-off. We want a more detailed explanation of the change in the probability of the selected nodes in $\mathcal{V}^* = \{J : J \in \mathcal{V}_{s,c}^*, s \in \{0,1\}, c = 1, \dots, C\}$.

The GNN for node classification at node J is generated by a computation graph, which is a spanning tree of G rooted at J of depth T. From the computation graph perspective, let a path be $(\dots, U, V, \dots, J)$, where U and V represent any two adjacent nodes on the path and J is designated as the root. For a GNN with T layers, the paths are sequences of $T+1$ nodes and we let $W(G)$ be the set of all such paths. Let $W_J(G) \subset W(G)$ be the paths ending at J. The symmetric set difference $\Delta W_J(G_0, G_1) = W_J(G_1)\Delta W_J(G_0)$ contains all paths rooted at J with at least one removed edge when some edges are masked. $\Delta W_J(G_0, G_1)$ is the changes to the computation graph that completely cause and explain the change in $\Pr_J$.

For node $J \in \mathcal{V}^*$, let the difference in its logits on graphs G_0 and G_1 be $\Delta\mathbf{z}_J(G_0, G_1) = \mathbf{z}_J(G_1) - \mathbf{z}_J(G_0) = [\Delta z_1, \dots, \Delta z_C]$. According to the method mentioned in [30], we apply it to GNN model and obtain the contribution of paths. Assume that the change in the logits can be linearly and exactly attributed to m removed propagation paths in $\Delta W_J(G_0, G_1)$. Formally, for $j = 1, \dots, C$, let $\Delta z_j = \sum_{p=1}^{m} C_{p,j}$, where $C_{p,j}$ is the contribution of the p-th removed path to Δz_j. Then

$$[z_1(G_0), \dots, z_C(G_0)] = [z_1(G_1), \dots, z_C(G_1)] - [\Delta z_1, \dots, \Delta z_C]. \tag{13}$$

We will explain the KL-divergence between $\mathsf{Pr}_J(G_1)$ and $\mathsf{Pr}_J(G_0)$, which is a well-defined distance metric of probability distributions. One can show that the KL-divergence $\text{KL}(\mathsf{Pr}_J(G_1) \| \mathsf{Pr}_J(G_0))$ is a function of Δz_j:

$$\begin{aligned} & \sum_{j=1}^{C} \mathsf{Pr}_j(G_1)\Delta z_j - \log[Z(G_1)/Z(G_0)], \\ = & \sum_{j=1}^{C} [\mathsf{Pr}_j(G_1)\Delta z_j] - \log Z(G_1) \\ & + \log \sum_{j=1}^{C} \exp(\underbrace{z_j(G_1) - \Delta z_j}_{=z_j(G_0)}), \end{aligned} \tag{14}$$

where $Z(G_\tau) = \sum_{j=1}^{C} \exp(z_j(G_\tau))$ for $\tau = 0, 1$.

Consider selecting a subset E_n of n paths from $\Delta W_J(G_0, G_1)$ as a microscopic explanation. The contributions from paths in E_n lead to a variant of Eq. (13)

$$\mathbf{z}_J(G_0) = \mathbf{z}_J(G_n) - \left[\sum_{p\in E_n} C_{p,1}, \dots, \sum_{p\in E_n} C_{p,C}\right]. \tag{15}$$

We can substitute $\mathbf{z}_J(G_0)$ in Eq. (14) with the right hand side of Eq. (15) and get

$$\sum_{j=1}^{C}\left[\mathsf{Pr}_j(G_1)\left(z_j(G_1)-z_j(G_0)-\sum_{p\in E_n}C_{p,j}\right)\right] - \log Z(G_1)+\log\sum_{j=1}^{C}\exp\left(z_j(G_0)+\sum_{p\in\Delta E_n}C_{p,j}\right) \quad (16)$$

This is the KL-divergence between two distributions $\mathsf{Pr}_J(G_1)$ and $\mathsf{Pr}_J(G_n)$ and has the minimum of 0. If it is close to 0, then the contribution from the paths in E_n, when added to G_0, can help GNN reproduce $\mathsf{Pr}(Y|G_1)$ and E_n thus succinctly explain the prediction shift when G_0 is changed to G_1.

We optimize E_n to minimize the KL-divergence in Eq. (16). Let $x_p \in [0,1]$, $p=1,\ldots,m$, represent the probabilities of selecting path p from $\Delta W_J(G_0,G_1)$ into E_n. We solve the following problem:

$$\begin{aligned}\mathbf{x}^* &= \underset{\substack{\mathbf{x}\in[0,1]^m\\ \|\mathbf{x}\|_1=n}}{\operatorname{argmin}} \sum_{j=1}^{c}\left(-\mathsf{Pr}_j(G_1)\sum_{p=1}^{m}x_pC_{p,j}\right)\\ &+ \log\sum_{j'=1}^{c}\exp\left(z_{j'}(G_0)+\sum_{p=1}^{m}x_pC_{p,j'}\right). \quad (17)\end{aligned}$$

In going from Eq. (16) to the objective, we ignore the constants $z_j(G_0)$, $z_j(G_1)$, and $\log Z(G_1)$. The problem is convex and ensures a unique optimal solution. The linear constraint ensures the total probabilities of the selected paths is n. E_n will then include the paths with the highest x_p^* values.

V. EXPERIMENT

A. Datasets and experimental settings.

TABLE II: Seven networks from three application domains.

Datasets	Classes	Nodes	Edges	Edge/Node	Features
Cora	7	2,708	10,556	3.90	1,433
Citeseer	6	3,321	9,196	2.78	3,703
PubMed	3	1,9717	44,324	2.24	500
Amazon-C	10	13,752	574,418	41.77	767
Amazon-P	8	7,650	287,326	37.56	745
Coauthor-C	15	18,333	327,576	17.87	6,805
Coauthor-P	5	34,493	991,848	28.76	8,415

We drew real-world datasets from three applications for the node classification task.

- Citeseer, Cora, and PubMed [16]: each node is a paper with a bag-of-words feature vector, and nodes are connected by the citation relationship. The goal is to predict the research area of each paper.
- Amazon-Computer (Amazon-C) and Amazon-Photo (Amazon-P) [29]: Amazon co-purchase graph, where nodes represent products and edges indicate that two products are frequently purchased together, node features are the bag-of-words vectors of the product reviews.
- Coauthor-Computer (Coauthor-C) and Coauthor-Physics (Coauthor-P): co-authorship graphs based on the Microsoft Academic Graph from the KDD Cup 2016 Challenge. We represent authors as nodes, that are connected by an edge if they co-authored a paper [29]. Node features represent paper keywords for each author's papers.

We randomly divide each graph into three portions with a ratio of *training : validation : test = 60 : 20 : 20*. The GNN parameter $\boldsymbol{\theta}$ is trained on the training set using the loss function $\mathcal{L}_C$. With $\boldsymbol{\theta}$ fixed, we optimize the mask M according to the Eq. (10). We remove edges with lower values in the mask M to obtain $\mathcal{E}^{'}$ for graph G_1. We explain the change in the fairness-accuracy trade-off as G_0 is shifted to G_1.

B. Baselines

1) Training the mask matrix M: We use the loss function $\min_M \mathcal{L}^{'} = \sum_{c=1}^{C}\lambda^c\mathcal{L}_{\text{EO}}^c + \mathcal{L}_C$ to train the mask M as a baseline. Compared with training with the upper-bounds $\tilde{\mathcal{L}}_{\text{EO}}^c$, it can be difficult to learn a satisfying M if the means between the two groups $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$ are similar. Similar to obtaining G_1, we obtain a new graph G_2 with a pruned set of edges $\mathcal{E}^{''}$.

2) Microscopic node-level contrastive explanations: We adopt the following methods as path explanation baselines. The proposed method is denoted as "**Stratified-Mean**".

- **Random sampling**. We randomly select nodes from $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$ to obtain $\mathcal{V}_{0,c}^*$ and $\mathcal{V}_{1,c}^*$. We run the baseline 1000 times to calculate the means and standard deviations of the fairness metrics.
- **Mean sampling**. We solve the problem in Eq. (11) without considering the constraints. We require $\mathcal{V}_{s,c}^*$ to have similar mean as $\mathcal{V}_{s,c}$ for $s=0,1$. The baseline may ignore the detailed differences in the histograms.
- **Stratified sampling**. Given the ratio of the selected nodes, we randomly select nodes from $F_{l,r}$ in the same ratio and obtain the $\mathcal{V}_{s,c}^*$, $s=0,1$. This baseline takes into account the details of class probability distributions through stratified sampling, but the randomness can hamper explainability. Evaluation metrics are calculated based on 1000 repetitions.

3) Microscopic path-based contrastive explanations: We adopt the following methods as path explanation baselines. The proposed method is denoted as "**AxiomPath-Convex**".

- **Gradient** (Grad) first computes the gradients of the logit of the predicted class j of the target node J, with respect to individual edges. Path importance is the sum of gradients of the edges on the path and is calculated on G_0 and G_1 independently. The contribution of a path to prediction change is the difference between the two path importance scores on G_0 and G_1 (if a path exists on just one graph, the path importance on the graph is used). The paths are ranked based on path importance to find E_n.
- **GNN-LRP** adopts the back-propagation attribution method LRP to GNN [27]. Path relevance is calculated in the same way as specified in [27] for node classification. The path ranking and selection is the same as the baseline Grad.

- **DeepLIFT** [30] can explain the change in predicted class on two graphs. For a target node, if the predicted class changes, the difference between a path's contributions to the new and original predicted classes is used to rank and select paths. If the predicted class remains the same but the distribution changes, a path's contribution to the predicted class is used. Only removed paths are ranked and selected.
- **AxiomPath-Topk** is a variant of AxiomPath-Convex. It selects the top paths with the highest contributions $\sum_{j=1}^{c} C_{p,j}$ into E_n.
- **AxiomPath-Linear** optimizes the objectives in Eq. (17) without the log terms. It is thus a linear programming problem. The resulting optimal $\mathbf{x}^*$ is processed in the same way as AxiomPath-Convex.

C. Evaluation metrics and performance

1) Training the mask matrix M: We use the $l_1(G_\tau) = \sum_{c=1}^{C} \bar{\mathcal{L}}^c_{\text{EO}}(G_\tau)$ and the $l_2(G_\tau) = \sum_{c=1}^{C} \mathcal{L}^c_{\text{EO}}(G_\tau), \tau = 0, 1, 2$ as the fairness metrics (note: G_2 is obtained by masking G_0 with the loss function $\mathcal{L}^c_{\text{EO}}$). We calculate $\Delta l_1 = l_1(G_\tau) - l_1(G_0)$ and $\Delta l_2 = l_2(G_\tau) - l_2(G_0)$ in the graph G_τ(τ=1,2) to evaluate the change in fairness. Node classification accuracy (Acc.) is also reported.

Based on table III, our method has the best performance on the Δl_2 over all datasets. On four settings (**Citeseer**, **Pubmed**, **Amazon-C** and **Coauthor-C**), it decreases the Δl_1 more than the baseline method. What's more, the baseline method only outperforms our method on two datasets for the Δl_1 and it increases the Δl_2 on the five datasets instead of reducing it. Fig. 3 shows the probability distribution of H_0 and H_1 in the graph G_0, G_1 and G_2, and we can see that our method makes the probability distribution of H_0 and H_1 closer.

TABLE III: Fairness (equal opportunity) performance (the lower (↓) the Δl_1 and the Δl_2 and the higher (↑) the accuracy, the better). The circle and the underlines denote the best method of the Δl_1 and the Δl_2. • indicates the best accuracy.

Datasets	G_0			G_1			G_2		
	l_1	l_2	Acc.	Δl_1(↓)	Δl_2(↓)	Acc.	Δl_1(↓)	Δl_2(↓)	Acc.
Cora	1.47	0.44	0.83	<u>-0.34</u>	0.02○	0.90●	0.71	0.15	0.81
Citeseer	1.74	0.77	0.74	<u>-0.53</u>	-0.34○	0.77●	0.44	-0.11	0.71
Pubmed	0.86	0.44	0.86	<u>-0.47</u>	-0.36○	0.86	0.26	-0.19	0.85
Amazon-C	4.20	1.74	0.65	<u>-2.44</u>	-1.33○	0.78●	-0.47	-1.32	0.70
Amazon-P	1.94	0.94	0.82	<u>-1.00</u>	-0.51	0.92●	-0.44	-0.55○	0.86
Coauthor-C	4.11	2.26	0.85	<u>-1.59</u>	-0.98○	0.89●	1.14	-0.64	0.80
Coauthor-P	0.88	0.38	0.92	<u>-0.34</u>	-0.17	0.96●	0.21	-0.22○	0.92

2) Macroscopic node-level contrastive explanations: The faithfulness of the macroscopic explanations of fairness changes is defined as

$$\sum_{\tau=0}^{1} |\bar{\mathcal{L}}^c_{\text{EO}}(G_\tau; \mathcal{V}_{0,c}; \mathcal{V}_{1,c}) - \bar{\mathcal{L}}^c_{\text{EO}}(G_\tau; \mathcal{V}^*_{0,c}; \mathcal{V}^*_{1,c})|, \tag{18}$$

where $\bar{\mathcal{L}}^c_{\text{EO}}(G_\tau; \mathcal{V}_{0,c}; \mathcal{V}_{1,c})$ denotes $\bar{\mathcal{L}}_{\text{EO}}$ calculated between the groups $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$ in the graph G_τ, and similarly for $\bar{\mathcal{L}}^c_{\text{EO}}(G_\tau; \mathcal{V}^*_{0,c}; \mathcal{V}^*_{1,c})$. If the selected nodes in $\mathcal{V}^*_{0,c}$ and $\mathcal{V}^*_{1,c}$ are representative of $\mathcal{V}_{0,c}$ and $\mathcal{V}_{1,c}$, respectively, the above metric should be small. We also demand the probability distribution of H^*_0 and H^*_1 is consistent with the H_0 and H_1, respectively. We calculate the EMD [25] distance between H_0 (H_1, resp.) and H^*_0 (H^*_1, resp.). The smaller the EMD distance, the more similar the two probability distributions are.

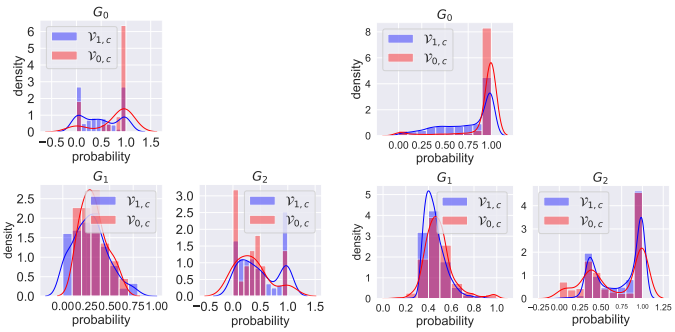

Fig. 3: The probability distribution histograms of H_0 and H_1 in the graph G_0, G_1 and G_2 on **Citeseer**(Left) and **Pubmed**(Right).

Based on Table IV, we can see that the proposed method (Stratified-Mean) achieves the best results across different classes on most datasets, with the smallest fairness change explanation faithfulness and the EMD distance. The stratified sampling method performs best in some classes on the EMD distance metric. Since it uniformly selects nodes from $F_{l,r}$, the probability distributions of H_0 (H_1, resp.) is close to H^*_0 (H^*_1, resp.). This also reveals that F plays an important role in preserving the shift in probability distribution from H_0 to H_1. For fairness change explanation faithfulness, in some classes, the mean sampling and the stratified-mean have the same effect. Although the stratified sampling performs better than stratified-mean in some classes, the gap is less significant.

We show the probability distribution of nodes located in $\mathcal{V}_{0,c}$, $\mathcal{V}_{1,c}$, $\mathcal{V}^*_{0,c}$ and $\mathcal{V}^*_{1,c}$ on Fig. 4. On the **Pubmed** and **Coauthor-C** datasets, the probability distributions of $\mathcal{V}^*_{0,c}$ and $\mathcal{V}^*_{1,c}$ obtained according to our method are consistent with $\mathcal{V}_{0,c}$, $\mathcal{V}_{1,c}$ in the two graphs G_0 and G_1. What's more, in the G_0, the nodes in $\mathcal{V}_{1,c}$ has a higher probability of being classified into class c compared to the nodes in $\mathcal{V}_{0,c}$, while in the G_1, the degree-related unfairness is almost negligible. In short, the key nodes obtained by our method can be effectively to explain why the GNN changes from unfairness to fairness.

3) Microscopic path-based contrastive explanations: For the selected nodes, we aim to find a small number of paths on computational graphs that explain the difference in predictions. We design the following metric to evaluate the faithfulness of the selected paths in explaining changes in class distributions:

$$\textbf{Fidelity}_{\text{KL}} = \frac{\text{KL}(\text{Pr}_J(\neg G_n) \| \text{Pr}_J(G_0))}{\text{KL}(\text{Pr}_J(G_1) \| \text{Pr}_J(G_0))}$$

where $\text{Pr}_J(\neg G_n)$ is the class distribution computed on the computation graph for G_1, with the selected paths in E_n added to the computational graphs on the input graph G_1, which *removed* some edges from G_0. Intuitively, if E_n indeed contains the more salient altered paths that turn G_0 into G_1, the less information the remaining paths can propagate, the more similar should $\neg G_n$ be to G_0, and thus the *smaller* the KL-divergence in the numerator. The denominator is

TABLE IV: Overall performance (the lower (↓) the Fairness metric and the EMD distance, the better). The underlines (circle, resp.) indicates the winner in the fairness metric (the EMD distance, resp.). Standard deviation are in parentheses.

Datasets	Class	Fairness metric (↓)				EMD distance(↓)			
		Random	Mean	Stratified	Stratified-Mean	Random	Mean	Stratified	Stratified-Mean
Cora	1	0.0468(± 0.0277)	0.0172	0.0268(± 0.0062)	0.0088∘	0.1304(± 0.0380)	0.1038	0.0706(± 0.0075)	0.0551
	2	0.0458(± 0.0267)	0.0052	0.0189(± 0.0036)	0.0017∘	0.1129(± 0.0388)	0.0427	0.0724(± 0.0045)	0.0322
	3	0.0215(± 0.0133)	0.0053	0.0051(± 0.0024)	0.0043∘	0.0377(± 0.0136)	0.0335	0.0189(± 0.0020)	0.0145
	4	0.0318(± 0.0193)	0.0024	0.0045(± 0.0025)	0.0023∘	0.0684(± 0.0222)	0.0363	0.0298(± 0.0036)	0.0298
	5	0.0382(± 0.0246)	0.0015	0.0433(± 0.0045)	0.0009∘	0.0787(± 0.0252)	0.0442	0.0734(± 0.0037)	0.0335
	6	0.0562(±0.0353)	0.0059	0.0792(± 0.0082)	0.0029∘	0.1292(± 0.0403)	0.0815	0.1227(± 0.0104)	0.0500
	7	0.0365(± 0.0235)	0.0099	0.0401(± 0.0033)	0.0043∘	0.0869(± 0.0263)	0.0527	0.0794(± 0.0039)	0.0321
	total	0.0313(± 0.0247)	0.0067	0.0313(± 0.0247)	0.0036∘	0.0869(± 0.0263)	0.0563	0.0668(± 0.0322)	0.0353
Citeseer	1	0.0620(± 0.0377)	0.0743	0.0398(± 0.0174)	0.0242∘	0.3693(± 0.1010)	0.3350	0.3607(± 0.0287)	0.2052
	2	0.0781(± 0.0454)	0.1105	0.0412(± 0.0114)	0.0092∘	0.2428(± 0.0751)	0.1962	0.1525(± 0.0116)	0.1027
	3	0.0549(± 0.0332)	0.0089	0.0328(± 0.0084)	0.0064∘	0.1277(± 0.0378)	0.0959	0.0771(± 0.0072)	0.0864
	4	0.0416(± 0.0269)	0.0081	0.0464(± 0.0065)	0.0046∘	0.1059(± 0.0309)	0.0737	0.0807(± 0.0079)	0.0606
	5	0.0577(± 0.0331)	0.0062	0.0344(± 0.0040)	0.0009∘	0.1170(± 0.0404)	0.0560	0.0736(± 0.0045)	0.0384
	6	0.0981(± 0.0563)	0.0213	0.0781(± 0.0129)	0.0172∘	0.1984(± 0.0720)	0.1392	0.1232(± 0.0116)	0.1265
	total	0.0654(± 0.0439)	0.0382	0.0456(± 0.0191)	0.0104∘	0.1936(± 0.1133)	0.1493	0.1447(± 0.1016)	0.1033
Pubmed	1	0.0156(± 0.0093)	0.0265	0.0067∘(± 0.0035)	0.0073	0.0605(± 0.0192)	0.0648	0.0308(± 0.0039)	0.0301
	2	0.0122(± 0.0064)	0.0126	0.0050(± 0.0022)	0.0023∘	0.0389(± 0.0089)	0.0296	0.0226(± 0.0025)	0.0171
	3	0.0123(± 0.0065)	0.0043	0.0045(± 0.0022)	0.0030∘	0.0385(± 0.0105)	0.0214	0.0209(± 0.0024)	0.0191
	total	0.0132(± 0.0077)	0.0144	0.0055(± 0.0030)	0.0042∘	0.0459(± 0.0169)	0.0386	0.0248(± 0.0053)	0.0221
Amazon-P	1	0.0412(± 0.0220)	0.0152	0.0259(± 0.0019)	0.0144∘	0.0722(± 0.0250)	0.0321	0.0350(± 0.0014)	0.0290
	2	0.0606(± 0.0342)	0.0167∘	0.0212(± 0.0012)	0.0167∘	0.1009(± 0.0460)	0.0359	0.0304(± 0.0009)	0.0273
	3	0.0517(± 0.0328)	0.0280∘	0.0288(± 0.0023)	0.0280∘	0.0906(± 0.0364)	0.0517	0.0444(± 0.0010)	0.0448
	4	0.0448(± 0.0268)	0.0081	0.0090(± 0.0022)	0.0049∘	0.0797(± 0.0364)	0.0368	0.0253(± 0.0033)	0.0245
	5	0.0531(± 0.0303)	0.0172∘	0.0229(± 0.0012)	0.0172∘	0.0891(± 0.0373)	0.0309	0.0272(± 0.0009)	0.0263
	6	0.0230(± 0.0161)	0∘	0.0127(± 0.0021)	0∘	0.0250(± 0.0150)	0.0103	0.0138(± 0.0014)	0.0068
	7	0.0237(± 0.0144)	0.0066	0.0022∘(± 0.0011)	0.0034	0.0398(± 0.0162)	0.0248	0.0081(± 0.0006)	0.0109
	8	0.0329(± 0.0207)	0.0013∘	0.0072(± 0.0012)	0.0013∘	0.0968(± 0.0445)	0.0180	0.0277(± 0.0011)	0.0164
	total	0.0412(± 0.0288)	0.0116	0.0162(± 0.0092)	0.0107∘	0.0742(± 0.0423)	0.0304	0.0265(± 0.0107)	0.0232
Amazon-C	1	0.0599(± 0.0316)	0.0093∘	0.0560(± 0.0024)	0.0093∘	0.1556(± 0.0687)	0.0317	0.0918(± 0.0016)	0.0271
	2	0.0345(± 0.0177)	0.0040	0.0026∘(± 0.0005)	0.0031	0.0651(± 0.0274)	0.0159	0.0101(± 0.0005)	0.0109
	3	0.0628(± 0.0334)	0.0162	0.0104∘(± 0.0104)	0.0140	0.1033(± 0.0438)	0.0253	0.0200(± 0.0006)	0.0230
	4	0.0506(± 0.0285)	0.0029	0.0028∘(± 0.0016)	0.0029	0.1329(± 0.0544)	0.0268	0.0212(± 0.0020)	0.0221
	5	0.0209(± 0.0114)	0∘	0.0017(± 0.0009)	0∘	0.0476(± 0.0179)	0.0366	0.0072(± 0.0008)	0.0104
	6	0.0704(± 0.0434)	0.0163∘	0.0244(± 0.0018)	0.0163∘	0.1256(± 0.0606)	0.0379	0.0414(± 0.0012)	0.0330
	7	0.0380(± 0.0210)	0.0004	0.0025(± 0.0013)	0.0001∘	0.0896(± 0.0367)	0.0232	0.0341(± 0.0021)	0.0142
	8	0.0693(± 0.0374)	0.0312∘	0.0378(± 0.0018)	0.0312∘	0.1555(± 0.0699)	0.0727	0.0584(± 0.0012)	0.0522
	9	0.0265(± 0.0159)	0.0037	0.0026∘(± 0.0008)	0.0033	0.0980(± 0.0406)	0.0445	0.0117(± 0.0009)	0.0250
	10	0.0601(± 0.0304)	0.0315∘	0.0326(± 0.0013)	0.0315∘	0.1372(± 0.0544)	0.0620	0.0524(± 0.0021)	0.0567
	total	0.0495(± 0.0336)	0.0115	0.0173(± 0.0183)	0.0111∘	0.1116(± 0.0613)	0.0376	0.0348(± 0.0254)	0.0274
Coauthor-C	1	0.0569(± 0.0338)	0.0207	0.0505(± 0.0063)	0.0063∘	0.1357(± 0.0435)	0.1152	0.0998(± 0.0082)	0.0526
	2	0.0310(± 0.0170)	0.0410	0.0169(± 0.0083)	0.0114∘	0.1080(± 0.0245)	0.0883	0.0634(± 0.0070)	0.0609
	3	0.0214(± 0.0126)	0.0138	0.0057∘(± 0.0030)	0.0095	0.0736(± 0.0185)	0.0487	0.0310(± 0.0035)	0.0429
	4	0.0461(± 0.0283)	0.0057	0.0037(± 0.0019)	0.0024∘	0.1069(± 0.0410)	0.0549	0.0346(± 0.0026)	0.0341
	5	0.0337(± 0.0220)	0.0035	0.0059(± 0.0026)	0.0007∘	0.0683(± 0.0235)	0.0352	0.0264(± 0.0031)	0.0270
	6	0.0224(± 0.0132)	0.0090	0.0039(± 0.0021)	0.0020∘	0.0577(± 0.0155)	0.0458	0.0235(± 0.0027)	0.0221
	7	0.0209(± 0.0108)	0.0076	0.0103(± 0.0040)	0.0045∘	0.0719(± 0.0211)	0.0514	0.0419(± 0.0035)	0.0278
	8	0.0410(± 0.0275)	0.0063	0.0060(± 0.0028)	0.0031∘	0.1080(± 0.0386)	0.0504	0.0373(± 0.0039)	0.0413
	9	0.0182(± 0.0105)	0.0375	0.0077(± 0.0042)	0.0046∘	0.0854(± 0.0214)	0.0864	0.0452(± 0.0051)	0.0451
	11	0.0312(± 0.0211)	0.0035	0.0047(± 0.0024)	0.0003∘	0.0532(± 0.0203)	0.0371	0.0177(± 0.0018)	0.0167
	12	0.0628(± 0.0167)	0.0102	0.0121(± 0.0035)	0.0027∘	0.0628(± 0.0189)	0.0994	0.0368(± 0.0036)	0.0299
	13	0.0302(± 0.0186)	0.0016	0.0151(± 0.0038)	0.0015∘	0.0639(± 0.0208)	0.0318	0.0312(± 0.0027)	0.0188
	14	0.0150(± 0.0093)	0.0019	0.0044(± 0.0020)	0.0006∘	0.0273(± 0.0089)	0.0366	0.0128(± 0.0013)	0.0106
	15	0.0397(± 0.0239)	0.0036	0.0167(± 0.0052)	0.0018∘	0.0728(± 0.0248)	0.0480	0.0311(± 0.0032)	0.0357
	total	0.0326(± 0.0255)	0.0118	0.0121(± 0.0122)	0.0036∘	0.0771(± 0.0377)	0.0592	0.0378(± 0.0206)	0.0332
Coauthor-P	1	0.0193(± 0.0112)	0.0087	0.0180(± 0.0027)	0.0039∘	0.0385(± 0.0113)	0.0496	0.0306(± 0.0023)	0.0286
	2	0.0247(± 0.0146)	0.0079	0.0031(± 0.0015)	0.0026∘	0.0497(± 0.0183)	0.0559	0.0185(± 0.0017)	0.0234
	3	0.0067(± 0.0042)	0.0001	0.0015(± 0.0008)	0∘	0.0114(± 0.0037)	0.0260	0.0043(± 0.0005)	0.0075
	4	0.0234(± 0.0141)	0.0039	0.0188∘(± 0.0033)	0.0020	0.0476(± 0.0154)	0.0469	0.0374(± 0.0020)	0.0234
	5	0.0800(± 0.0494)	0.0122	0.0126(± 0.0035)	0.0007∘	0.1444(± 0.0669)	0.0787	0.0385(± 0.0031)	0.0340
	total	0.0312(± 0.0371)	0.0065	0.0108(± 0.0077)	0.0018∘	0.0578(± 0.0564)	0.0514	0.0259(± 0.0131)	0.0233

for normalization since the total change can be of different scales among target nodes/edges. At one extreme, E_n contains no salient paths so that the numerator is close to $\mathrm{KL}(\mathsf{Pr}_J(G_1)\|\mathsf{Pr}_J(G_0))$ and the fidelity is 1 (the *worst*). At the other extreme, $\neg G_n$ degrades to G_0 after eliminating the effect of the most salient altered paths and thus the fidelity is 0 (the *best*). The metric is different from the objective function in Eq. (17), so that AxiomPath-Convex does not have the privilege over the baselines due to the evaluation metric. For a fair comparison, we ensure the same number of altered paths are selected into E_n for all methods. Explanation simplicity is evaluated as we vary the size of the set E_n.

From Fi. 5, we can see that AxiomPath-Convex has the best (smallest) fidelity over all levels of explanation complexities

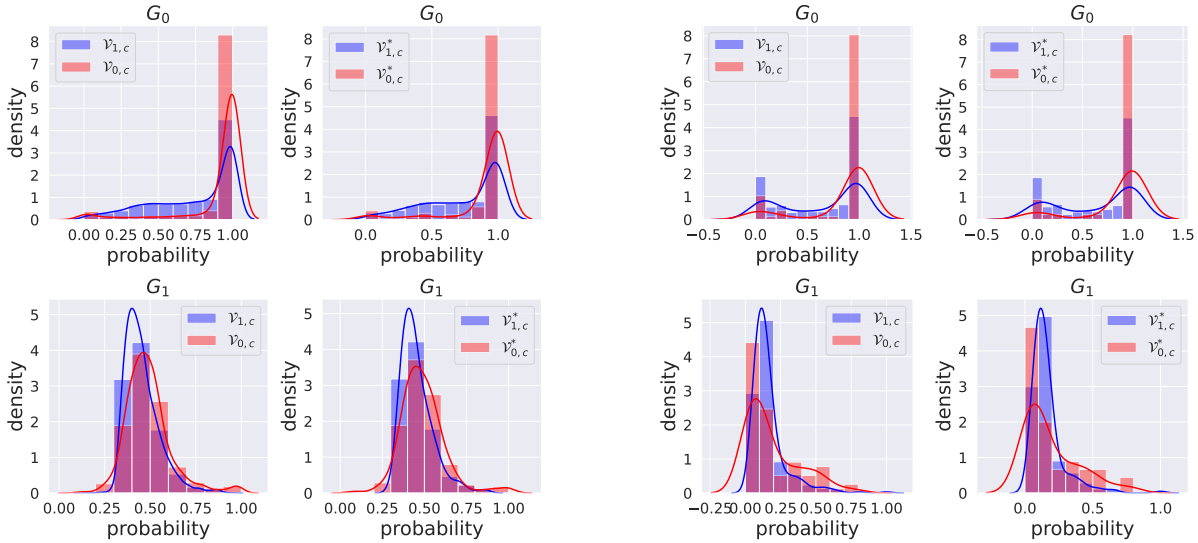

Fig. 4: The probability distribution of nodes located in $\mathcal{V}_{0,c}$, $\mathcal{V}_{1,c}$, $\mathcal{V}^*_{0,c}$ and $\mathcal{V}^*_{1,c}$ in the graph G_0 and G_1 on the predicted class (Left 4 plots: **Pubmed**; Right 4 plots: **Coauthor-C**). For each dataset, the upper left figure shows the probability distribution of nodes located in $\mathcal{V}_{0,c}$, $\mathcal{V}_{1,c}$ in graph G_0. The upper right figure shows the probability distribution of important nodes $\mathcal{V}^*_{0,c}$ and $\mathcal{V}^*_{1,c}$ obtained by our method in graph G_0. The bottom left and right figures show the probability distribution on G_1.

and over all datasets. On three settings (**Cora**, **Coauthor-P** and **Coauthor-C**), the gap between AxiomPath-Convex and the runner-up is significant. On the remaining settings, the gap is less significant but still not ignorable. AxiomPath-Topk and AxiomPath-Linear underperform AxiomPath-Convex, indicating that non-linear log term must be considered when finding optimal salient paths.

VI. Conclusions

We studied degree-related group fairness for GNN. We addressed the issues of prior works, such as the lack of explantation about why a GNN model becomes fairer or less so with a different adjacency matrix. The proposed algorithm for training the fair GNNs can not only ensure the fairness but also have the outstanding performance. What's more, for the node based explanations, we obtain the key nodes via the linear programming to explain the reason why the GNNs trend to be fair. While for the path based explanations, we optimally select a small subset of paths to explain the change in prediction change. On seven graph datasets we demonstrate the effectiveness in debiasing while keeping high accuracy. Experiments showed the superiority of the proposed node and path based explanations over state-of the-art baselines.

Acknowledgement

Sihong Xie is supported in part by the National Science Foundation under Grants NSF IIS-1909879, NSF CNS-1931042, NSF IIS-2008155, and NSF IIS-2145922. Yazheng Liu and Xi Zhang are supported by the Natural Science Foundation of China (No.61976026) and the 111 Project (B18008). Any opinions, findings, conclusions, or recommendations expressed in this document are those of the author(s) and should not be interpreted as the views of the National Science Foundation.

References

[1] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 2002.
[2] Federico Baldassarre and Hossein Azizpour. Explainability techniques for graph convolutional networks. In *ICML Workshop*, 2019.
[3] Avishek Bose and William Hamilton. Compositional fairness constraints for graph embeddings. In *International Conference on Machine Learning*. PMLR, 2019.
[4] Kai Burkholder, Kenny Kwock, Yuesheng Xu, Jiaxin Liu, Chao Chen, and Sihong Xie. Certification and trade-off of multiple fairness criteria in graph-based spam detection. In *CIKM*, 2021.
[5] Maarten Buyl and Tijl De Bie. Debayes: a bayesian method for debiasing network embeddings. In *International Conference on Machine Learning*, pages 1220–1229. PMLR, 2020.
[6] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018.
[7] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. Power-law distributions in empirical data. *SIAM review*, 2009.
[8] Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *WSDM*, 2021.
[9] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *ACM SIGCOMM computer communication review*, 1999.
[10] Golnoosh Farnadi, Behrouz Babaki, and Lise Getoor. Fairness in relational domains. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 108–114, 2018.
[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.
[12] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.
[13] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.
[14] Faisal Kamiran and Toon Calders. Classifying without discriminating. In *ICCC*. IEEE, 2009.
[15] Ahmad Khajehnejad, Moein Khajehnejad, Mahmoudreza Babaei, Krishna P Gummadi, Adrian Weller, and Baharan Mirzasoleiman. Crosswalk: Fairness-enhanced node representation learning. *arXiv preprint arXiv:2105.02725*, 2021.
[16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

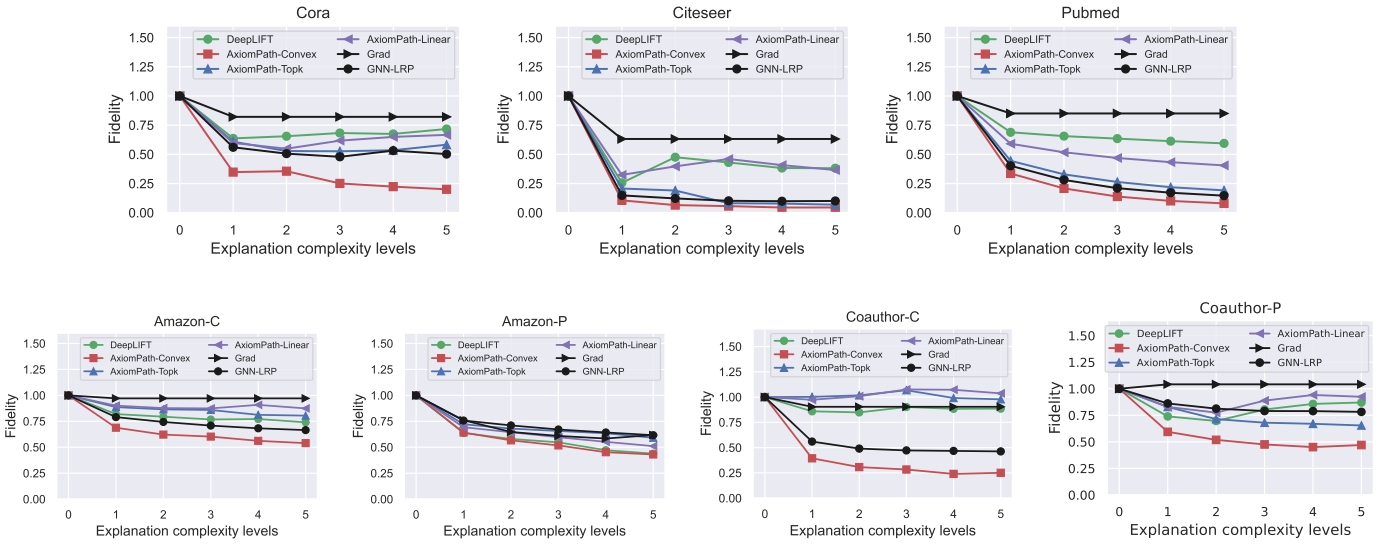

Fig. 5: Performance on the microscopic path-based contrastive explanations over the seven datasets. Each figure denotes the dataset and shows the fidelity as the number of selected paths range in pre-defined 5 levels of explanation complexity. Our method has the best performance over the baselines.

[17] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. Rev2: Fraudulent user prediction in rating platforms. In *WSDM*, 2018.

[18] Peizhao Li, Yifei Wang, Han Zhao, Pengyu Hong, and Hongfu Liu. On dyadic fairness: Exploring and mitigating bias in graph connections. In *ICLR*, 2020.

[19] Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the fairness of disentangled representations. *Advances in Neural Information Processing Systems*, 2019.

[20] Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

[21] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. In *NeurIPS*, 2020.

[22] Farzan Masrour, Tyler Wilson, Heng Yan, Pang-Ning Tan, and Abdol Esfahanian. Bursting the filter bubble: Fairness-aware network link prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 841–848, 2020.

[23] Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *CVPR*, 2019.

[24] Tahleen Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. Fairwalk: Towards fair graph embedding. 2019.

[25] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 2000.

[26] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for nlp with differentiable edge masking. *arXiv preprint arXiv:2010.00577*, 2020.

[27] Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima Kristof T. Sch¨utt, Klaus-Robert M¨uller, and Gr´egoire Montavon. Higher-order explanations of graph neural networks via relevant walks. 2020.

[28] Robert Schwarzenberg, Marc Hübner, David Harbecke, Christoph Alt, and Leonhard Hennig. Layerwise relevance visualization in convolutional text graph classifiers. *arXiv preprint arXiv:1909.10911*, 2019.

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

[30] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. In *ICML*, 2017.

[31] Indro Spinelli, Simone Scardapane, Amir Hussain, and Aurelio Uncini. Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning. *IEEE Transactions on Artificial Intelligence*, 2021.

[32] Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Yiqi Wang, Jiliang Tang, Charu Aggarwal, Prasenjit Mitra, and Suhang Wang. Investigating and mitigating degree-related biases in graph convoltuional networks. In *CIKM*, 2020.

[33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[34] Minh Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. *Advances in neural information processing systems*, 33:12225–12235, 2020.

[35] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. A semi-supervised graph attentive network for financial fraud detection. In *ICDM*, 2019.

[36] Zhenqin Wu, Bharath Ramsundarand Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay S. Pande. Moleculenet: a benchmark for molecular machine learning. 2018.

[37] Chih-Kuan Yeh, Joon Sik Kim, Ian E H Yen, and Pradeep Ravikumar. Representer Point Selection for Explaining Deep Neural Networks. In *NeurIPS*, 2018.

[38] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnn explainer: A tool for post-hoc explanation of graph neural networks. In *NeurIPS*, 2019.

[39] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018.

[40] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. 2020.

[41] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*, 2020.

[42] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. *ICML*, 2021.

[43] Lu Zhang, Yongkai Wu, and Xintao Wu. Achieving non-discrimination in data release. In *SIGKDD*, 2017.