GNNFairViz: Visual Analysis for Graph Neural Network Fairness

Xinwu Ye, Jielin Feng, Erasmo Purificato, Ludovico Boratto, Michael Kamp, Zengfeng Huang, and Siming Chen

***Abstract*—Recent advancements in Graph Neural Networks (GNNs) show promise for various applications like social networks and financial networks. However, they exhibit fairness issues, particularly in human-related decision contexts, risking unfair treatment of groups historically subject to discrimination. While several visual analytics studies have explored fairness in machine learning (ML), few have tackled the particular challenges posed by GNNs. We propose a visual analytics framework for GNN fairness analysis, offering insights into how attribute and structural biases may introduce model bias. Our framework is model-agnostic and tailored for real-world scenarios with multiple and multinary sensitive attributes, utilizing an extended suite of fairness metrics. To operationalize the framework, we develop GNNFairViz, a visual analysis tool that integrates seamlessly into the GNN development workflow, offering interactive visualizations. Our tool enables GNN model developers, the target users, to analyze model bias comprehensively, facilitating node selection, fairness inspection, and diagnostics. We evaluate our approach through two usage scenarios and expert interviews, confirming its effectiveness and usability in GNN fairness analysis. Furthermore, we summarize two general insights into GNN fairness based on our observations on the usage of GNNFairViz, highlighting the prevalence of the "Overwhelming Effect" in highly unbalanced datasets and the importance of suitable GNN architecture selection for bias mitigation.**

***Index Terms*—Fairness, graph neural networks, visual analytics.**

I. Introduction

OVER the past decades, the widespread adoption of machine learning (ML) systems in making human-related decisions has raised worries that biases within these models could result in decisions that disadvantage historically marginalized groups. For example, an ML model designed to predict criminal activity might make unfair decisions based on race [1], while a hiring decision model could discriminate against individuals based on gender and age (as illustrated in fig. 2). The fairness issues observed in ML systems can be traced back to biases inherent in the data used to train these models [2]. This problem becomes particularly complex in the context of graph data, where nodes and edges are not independent and identically distributed (non-IID), as is typically assumed in ML. Graph Neural Networks (GNNs), as one kind of ML models designed for graph data, represent a significant leap in diverse domains like social networks [3] and financial networks [4]. Despite their shared propensity with other ML models to inherit biases from historical datasets—particularly regarding sensitive attributes like age, gender, and ethnicity—GNNs face additional fairness issues due to the intrinsic graph structure, which can distort the effect of attribute bias [5]. Therefore, novel approaches to fairness tailored for GNNs have been proposed [6]–[8].

Ensuring fairness in GNNs is a challenging task due to the interplay between node attribute bias and structural bias. Existing bias mitigation approaches vary in focus, targeting aspects such as edges, node attributes [9], [10], or fair model architectures [11], [12]. Therefore, understanding biases in GNNs is crucial, as it informs the selection and development of effective debiasing approaches. Existing methods for interpreting fairness in GNNs [9], [13], [14] provide specific parts of data as explanations, thus falling short of being comprehensible and providing deep insights. As a solution, visual analytics approaches have shown great promise in enabling interactive and in-depth analyses of ML fairness, offering a path forward for diagnosing and understanding fairness issues in GNNs [15]. However, applying these tools directly to GNNs is challenging, as most are designed for ML models on Euclidean data and cannot adequately address the unique fairness issues in GNNs, where the graph structure often distorts the effect of attribute bias through GNNs' message-passing process.

With the primary goal of uncovering and diagnosing potential fairness issues in GNNs before proceeding with model deployment, we propose a model-agnostic visual analytics framework that analyzes GNN fairness from a data-centric viewpoint, offering GNN model developers, the intended users, insights into bias in their models. This framework is instantiated as a tool named GNNFairViz, a Python package, which seamlessly integrates into users' GNN development workflow. GNNFairViz is designed for the interactive visualization and inspection of model bias, enabling flexible node selection and supporting fairness diagnostics from the perspective of data bias. Moreover, our approach supports multi-class data and enables the simultaneous analysis of multiple sensitive attributes, revealing fairness issues that may remain hidden when sensitive attributes are analyzed individually (cf. fig. 2).

We demonstrate the effectiveness and usability of GNNFairViz in GNN fairness analysis through two usage scenarios

Xinwu Ye, Jielin Feng, Zengfeng Huang, and Siming Chen are with Fudan University, China. E-mail: {xwye23, 23110980031}@m.fudan.edu.cn, {huangzf, simingchen}@fudan.edu.cn Siming Chen is the corresponding author.

Erasmo Purificato is with Joint Research Centre, European Commission, Italy. Email: erasmo.purificato@acm.org. The author contributed to this work while affiliated with Otto von Guericke University Magdeburg, Germany. The view expressed in this paper is purely that of the author and may not, under any circumstances, be regarded as an official position of the European Commission.

Ludovico Boratto is with University of Cagliari, Italy. E-mail: ludovico.boratto@acm.org.

Michael Kamp is with Ruhr-University Bochum, Germany. E-mail: michael.kamp@uk-essen.de.

Manuscript received April 19, 2021; revised August 16, 2021.

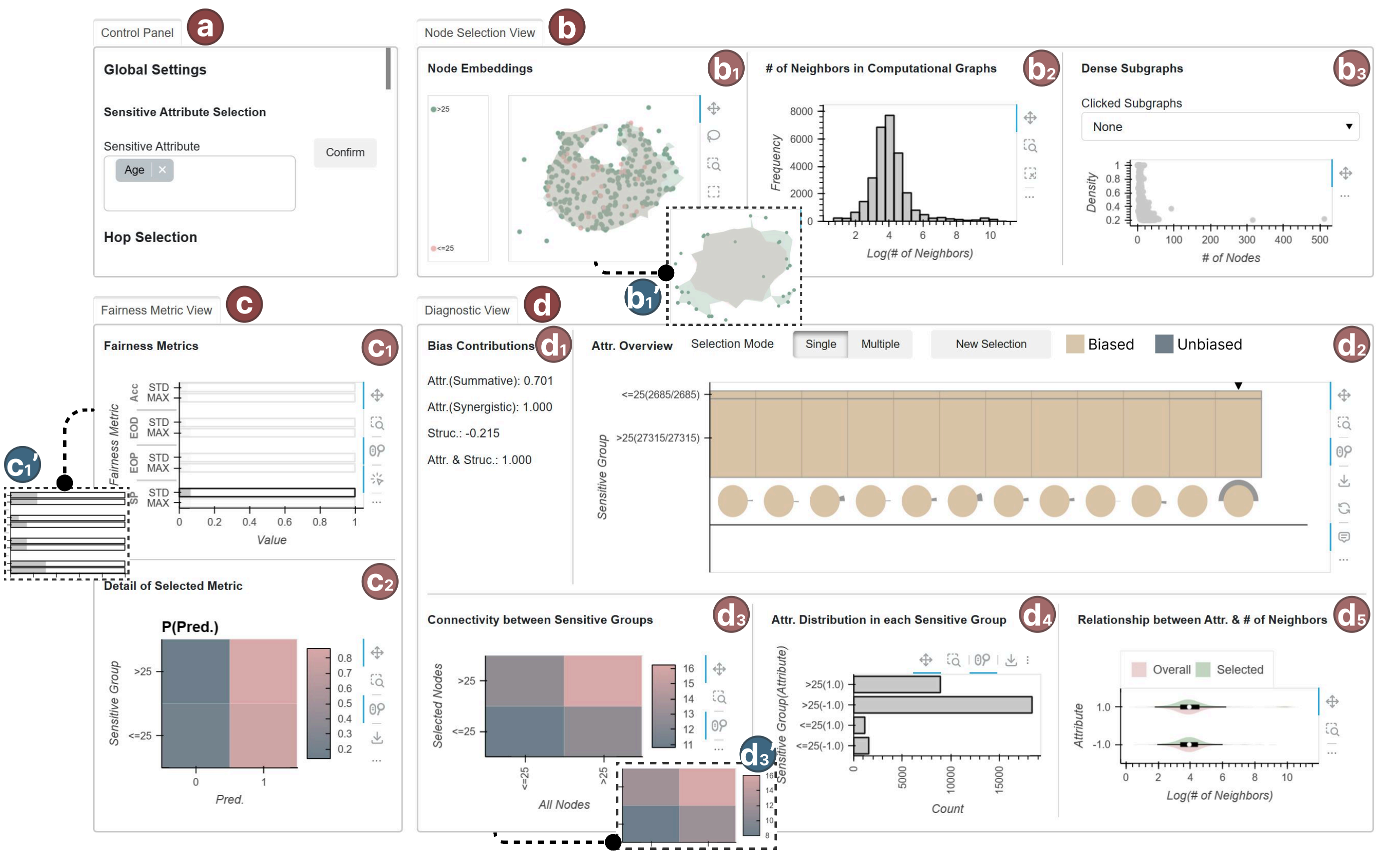

Fig. 1. GNNFairViz provides a multi-view interface for analyzing fairness in Graph Neural Network (GNN) models. The interface consists of: (a) the Control Panel for configuring sensitive attributes, hop selections, and visualization settings; (b) the Node Selection View, featuring visualization of (b_1) node embeddings, (b_2) distribution of number of neighbors in computational graphs, and (b_3) dense subgraphs; (c) the Fairness Metric View, showing (c_1) fairness metrics and (c_2) detailed breakdowns of selected metrics; and (d) the Diagnostic View, including (d_1) bias contributions, (d_2) attribute overview, (d_3) connectivity patterns between sensitive groups, (d_4) attribute distributions in each sensitive group, and (d_5) the relationship between attributes and the number of neighbors. This figure shows the use of GNNFairViz for age-related fairness analysis in default predictions with a Graph Attention Network (GAT) model trained on the Credit dataset, where the user identifies and verifies that the graph structure promotes model fairness. Key observations that support this conclusion are the numerical patterns of inter- and intra-group edges displayed in d_3 and d_3'. For further details, please refer to section VII-A.

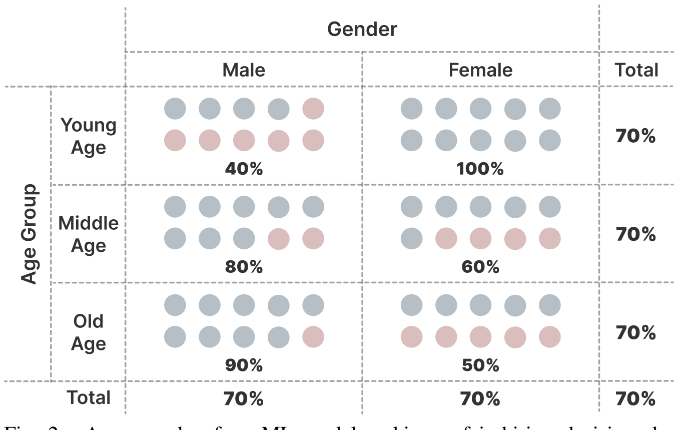

Fig. 2. An example of an ML model making unfair hiring decisions by discriminating against individuals based on gender and age. Sensitive groups are formed by combining age (young, middle-aged, and old) and gender (male and female), resulting in six distinct groups. The model outputs for each sensitive group (represented by each cell) differ, while they remain the same for each individual sensitive attribute category.

and expert interviews. Summarizing, the contributions of this paper include:

- A visual analytics framework supporting a human-in-the-loop approach for analysing fairness issues in GNNs from the perspectives of graph data structure and node attributes (section V).
- An interactive visual analysis tool based on the proposed framework, integrated into GNN model developers' original working environment (Jupyter notebook) and workflow (section VI).
- Two usage scenarios on GNN fairness analysis and interviews with experts demonstrating the usability and effectiveness of GNNFairViz (section VII), along with two general insights into GNN fairness applicable to different GNN architectures and datasets (section VIII).

II. Preliminaries

In this section, we introduce the fundamental concepts and mechanisms of GNNs and the node classification task, providing a foundation for the subsequent discussion. We then present the fairness notions applied in this work.

A. Graph Neural Networks and Node Classification Task

GNNs operate on graphs $G = (V, E, X)$, where $V = \{v_1, \dots, v_n\}$ denotes the set of $n \in \mathbb{N}$ nodes, and $E = \{(v_i, v_j) | v_i, v_j \in V\}$ denotes the set of edges. $X \in \mathbb{R}^{n \times d}$ is the node feature matrix, i.e., each node v_i has a d-dimensional feature vector associated to it. The set of edges E can be represented by an adjacency matrix $A \in \mathbb{R}^{n \times n}$, where $a_{ij} = 1$ indicates that $(v_i, v_j) \in E$, and $a_{ij} = 0$ otherwise.

In this paper, we consider the task of node classification, where each node is labeled - represented by a vector of node labels $Y \in \mathbb{R}^n$. The task is to predict a node's label based on its features and information from the graph structure. The probabilistic classification output for the n nodes is represented as $\hat{Y} = \{\hat{Y}_1, \ldots, \hat{Y}_n\}$, with $\hat{Y}_i \in \mathbb{R}^c$ and c being the number of classes. The predicted labels are denoted as $\hat{y} = \{\hat{y}_1, \ldots, \hat{y}_n\}$.

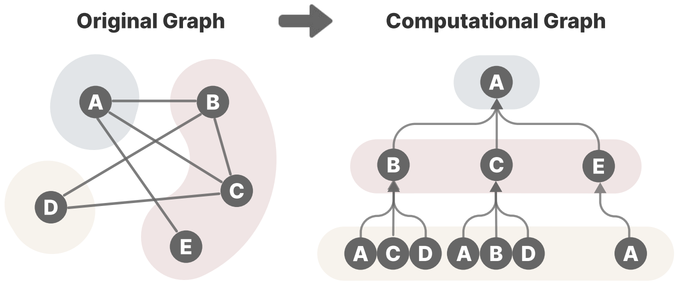

Fig. 3. An example of the computational graph of node A in GNNs. In the original graph, node A's 1-hop neighbors are nodes B, C, and E, while node D is a 2-hop neighbor. In the computational graph, which represents how node A aggregates information from its neighbors, the 1-hop neighbors remain the same (B, C, and E). However, the 2-hop neighbors include three instances of node A, one node B, one node C, and two nodes D, which are composed of 1-hop neighbors of nodes B, C, and E, respectively.

The unique aspect of GNNs lies in their ability to incorporate both node attributes and graph structure into learning [16], [17]. While different types of GNNs exist, including spectral and message-passing-based models, most state-of-the-art GNN architectures, such as Graph Convolutional Networks (GCNs), are based on the message-passing mechanism [18]. The core concept of it is the computational graph [16], where node representations are updated in layers based on the attributes of themselves and their neighboring nodes, as shown in fig. 3. At each layer k, the representation $h_v^{(k)}$ of a node v is updated based on its previous layer's representation $h_v^{(k-1)}$ and the representations of its neighboring nodes, denoted as $N(v)$. Thus, each layer represents one iteration of a message-passing process. Based on the message-passing mechanism, a computational graph $G_v = (V^v, E^v, X_{V^v})$ of node v is naturally defined for a GNN model that aggregates information K times. The edge set E^v includes edges between nodes across different levels of adjacency in the computational graph:

$$E^v = (E_{\{v\},N_1(v)}, E_{N_1(v),N_2(v)}, \ldots, E_{N_{K-1}(v),N_K(v)}),$$

where $N_k(v)$ is the set of nodes at the k-th hop to the target node v, and $E_{U,U'}$ represents the edge set consisting of edges between nodes in node set U and U'. Different GNN architectures might adopt techniques like attention mechanism used in GATs [17], resulting in modified computational graphs. Yet, throughout the rest of the paper, we refer to the original computational graph form.

B. Fairness Notions

The concept of algorithmic fairness in ML can be divided into two main categories: individual fairness and group fairness. Individual fairness advocates for comparable outcomes for individuals who are similar [8], [19]. Group fairness entails ensuring equitable treatment across various demographic groups, typically defined by sensitive attributes like gender, religion, and race [6], [7]. The majority of studies predominantly concentrate on group fairness [5], which is also the focal point of our research. In this paper, we refer to the group fairness present in a specific analyzed model as **model bias**. For group fairness in GNNs specifically, nodes are divided into different sensitive groups according to their sensitive attributes. Formally, group fairness in the context of a GNN model ensures that when a specific metric M is used to evaluate the model's outputs $\hat{y}_{g_i}$ and $\hat{y}_{g_j}$ for different node groups g_i and g_j (where $i, j = 1, 2, \ldots, m$ and m represents the number of sensitive groups), this metric should provide comparable results for each group. This is expressed as $M(\hat{Y}_{g_i}) = M(\hat{Y}_{g_j})$. The selection of a specific metric M dictates the establishment of specific definitions of fairness and the corresponding metrics.

Through an extensive review of existing literature on algorithmic fairness, we summarize a collection of notions and the corresponding metrics designed for assessing bias in node classification across various fairness principles. The fairness notions and metrics, detailed in Tab. 1 of Sec. A of the appendix, assess bias through various lenses, including Micro-F1 Score Parity (MSP), Accuracy Parity (AP), Overall Misclassification Rate Parity (OMRP), Statistical Parity (SP), Equality of Odds (EOD), and Equality of Opportunity (EOP). Specifically, each metric evaluates different aspects of fairness: MSP, AP, and OMRP analyze fairness through the related performance scores; SP assesses fairness considering the positive prediction rate parity; EOD and EOP evaluate, respectively, the parity of true positive and false positive rates (alternatively, false negative and false positive rates) and the true positive rate parity. Among them, MSP, AP, SP, and EOP metrics are applicable to scenarios involving multi-class node classification and multiple sensitive groups. Conversely, OMRP and EOD are reserved for binary scenarios; OMRP is applicable to multi-class node classification, whereas EOD is specific to binary tasks. Therefore, certain fairness notions and metrics are not applicable to multinary cases, which often arise in real-world scenarios [20], [21]. To address this, in section V-A1, we extend the fairness metrics to be used in this paper to encompass these practical settings.

ML fairness issues are primarily attributed to the potential presence of proxy variables linked to sensitive attributes of instances, i.e., different distributions of attributes with respect to sensitive attributes, a phenomenon known as **attribute bias** [22], [23]. Similarly, GNNs, which inherently rely on the attributes of the nodes, are not exempt from this challenge, consequently resulting in analogous fairness concerns [24]. The complexity of fairness in GNNs is extorted by the potential biases in graph structure (referred to as **structural bias** throughout the rest of the paper), which stands for the difference of connectivity between and within sensitive groups [10], [24], [25]. Research shows that GNN models may have greater fairness deficits than MLPs, highlighting the role of message-passing mechanisms in amplifying biases in node attributes [26]. Therefore, our approach analyzes GNN fairness from the perspectives of both attribute bias and structural bias. We denote this combined bias as **data bias**, since it captures the differences of attributed graph data distributions of different sensitive groups.

III. Related Work

This section reviews recent contributions related to our paper's topics.

A. Methods for Fairness in Graph Neural Networks

Tailored approaches to promote fairness in GNN models include pre-processing methods that mitigate bias by altering the training dataset (e.g., edge rewiring [24]), in-processing strategies that enhance fairness through mechanisms like regularization, novel objectives, or architectural designs [6]–[8], and post-processing techniques that adjust model outputs to ensure fairness [27]. However, GNN model developers must understand the factors driving model bias to make informed choices about mitigation techniques. Recent studies [9], [13], [14] have attempted to shed light on the origins of bias in GNNs. For example, REFEREE [9] provides specific edges that significantly influence model bias as explanations. However, the explanations offered by these methods are automatically generated and can often be intricate and challenging to comprehend, such as complex subgraphs of social networks. These disadvantages necessitate the use of interactive visual aids for clearer understanding and incorporating human knowledge. To this end, our paper offers an interactive approach for GNN fairness analysis.

B. Visual Analytics for General Machine Learning Fairness

Visualization has been used to analyze fairness issues in ML. Basic visualization charts, like bar charts for group accuracy [28] and scatter plots for embeddings [29], are often used to illustrate fairness in ML. Tools such as Responsibly [30], IBM AI Fairness 360 [31], and Dalex [32] calculate and display fairness metrics. However, these tools primarily provide static views, limiting their utility for in-depth fairness analysis. Interactive visual analysis tools have recently enhanced the thoroughness of fairness analysis. The What-If Tool [33] allows users to adjust classifier thresholds and compare models using fairness metrics. FAIRVIS [34], DiscriLens [1], and FairCompass [35] focus on auditing fairness in classification by analyzing groups. For ranking tasks, FairSight [36] visualizes decision impacts and fairness metrics, helping address biases and balance fairness trade-offs.

C. Visual Analytics for Graph-Based ML Fairness

In graph-based ML, FairRankVis [15] and BiaScope [37] introduce innovative methodologies for exploring and diagnosing algorithmic fairness. FairRankVis focuses on graph ranking algorithms, providing tools to explore fairness concerns specifically related to how entities are ranked based on their connections within a graph. BiaScope, meanwhile, is dedicated to identifying biases in node embeddings, helping to uncover how structural properties within the graph might influence the representation of nodes.

Although these current tools encompass a broad spectrum of fairness concerns in ML, including for graph-based tasks, they fall short in analyzing GNN fairness through the combined impact of node attributes and graph structure, which is the root of the complexity of GNN fairness problems [10], [24], [26]. Our paper centers on analyzing GNN fairness through data bias. We introduce an innovative visual analytics approach designed to support this endeavor, focusing on bias inspection, node selection, and bias diagnostics to analyze model bias from the perspective of data bias, which is further divided into attribute bias and structural bias.

IV. The Design of GNNFairViz

We initially undertook a review of literature focusing on GNNs fairness, as well as visual analytics for ML fairness and GNNs. To gain deeper insights into the typical practices and requirements in the analysis of GNN fairness, we engaged intensively with three experts (E1-E3), with E1 and E3 contributing as coauthors. E1 is a doctoral candidate with expertise in Human-Centered AI, focusing on GNN fairness, and is also involved in usability and user experience studies. E2 is a Ph.D. concentrating on trustworthy graph learning and possesses substantial knowledge regarding fairness challenges in GNNs. E3, a senior researcher, has a rich background in graph representation learning. During the collaboration, we conducted interviews with them, asking about the difficulties they encountered when analyzing GNN fairness and the aspects they believed have not yet been addressed by existing works. Concurrently, we iteratively developed GNNFairViz, incorporating their feedback to enhance its efficacy. In what follows, we discuss the challenges in GNN fairness analysis and the design requirements for our approach based on the above process. These challenges and design requirements have not been sufficiently tackled and satisfied by existing tools, and this inadequacy motivates the design of our approach.

A. Challenges

We have pinpointed three primary challenges in GNN fairness analysis through the review of the literature and discussions with experts:

C1. Evaluating model fairness from **different perspectives**. As all of our experts have highlighted, one of the fundamental challenges for a GNN practitioner is determining whether the model is fair. This task is challenging because a model might be considered fair from certain perspectives but unfair from others. As summarized in section II-B, there are various fairness notions, making it a non-trivial task to evaluate model fairness comprehensively.

C2. Analyzing model bias from the perspective of **data** (both node attributes and graph structure). GNN models, like other ML models, learn data distributions, meaning that bias present in the data will be introduced into models and reflected in their outputs, resulting in biased models [9], [13], [38]. During interviews, experts agreed with this bias introduction mechanism. They also noted that the problem is more complex for GNNs compared to general ML models due to the graph structure topology, which poses additional challenges. In particular, as stated by E2, "*we want to analyze data bias and their impact on model bias. However, the challenge lies in analyzing both data attributes and graph structure, as well as their interaction, effectively.*"

C3. Gaining **interpretable** insights into model bias. Most existing approaches for GNN fairness explanation [9], [13], [14], or GNN explainability approaches that can be adapted

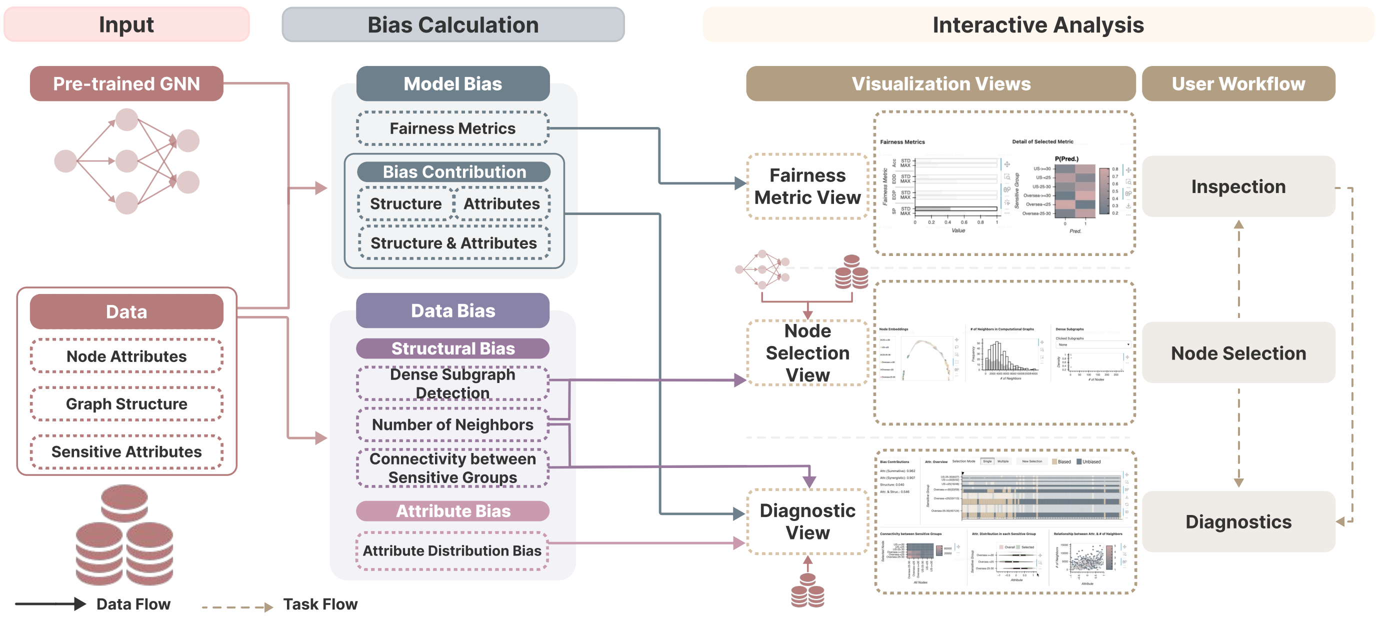

Fig. 4. GNNFairViz takes a pre-trained GNN model and data as input, i.e., node attributes, graph structure, and sensitive attributes. GNNFairViz processes a pre-trained GNN model and data—node attributes, graph structure, and sensitive attributes. The bias calculation phase computes model bias by combining the GNN's output (inferred from the attributed graph) with sensitive attributes, and data bias, split into structural and attribute bias, based on the data. The interactive analysis phase provides three views for fairness inspection, node selection, and diagnostics, respectively.

for fairness, provide explanations that are difficult for humans to interpret. As E2 and E3 both stated, the outputs of existing GNN fairness explanation mehods—often represented as subgraphs or attribute subsets—are hard to interpret, rendering them uninformative. Experts further explained that while these forms of explanations might be insightful for drug or biology applications—where the graphs are relatively small—they are not helpful for fairness problems. These problems often involve human-related networks of a larger scale, where large subgraphs as explanation provide limited insights.

B. Design Requirements

Based on these challenges, we have compiled a set of design requirements for our GNN fairness analysis approach.

R1. Supporting customizing and inspecting fairness through various perspectives. Users should be facilitated with the examination of various fairness concepts (**C1**). This requires offering a comprehensive set of fairness metrics.

R2. Providing clues and interactions for node selection. To analyze model bias from a data perspective (**C2**), users need to know which subset of nodes are important and select them. This allows them to further examine the impact of these nodes' attributes and their related edges.

R3. Allowing the progressive establishment of insights during analysis. Since static explanations in the form of graph data subsets are difficult for humans to understand, it is essential to support progressive analysis before drawing final conclusions. This allows for an in-depth diagnosis of how model bias is introduced by data bias (**C2**, **C3**).

C. Overview

The overview of our approach, as illustrated in fig. 4, encompasses three key modules across two distinct phases: the calculation of model bias (**R1**, **R3**) and data bias (**R2**, **R3**), which occur concurrently in the bias calculation phase, followed by a visualization module in the interactive analysis phase. The process begins by inputting a pre-trained GNN model along with data comprising node attributes, graph structure, and sensitive attributes. The bias calculation phase computes fairness metrics (section V-A1) and quantifies bias contributions from graph structure and attributes (section V-A2) within the model bias module. It also detects structural bias (**R2**, **R3**) and attribute bias (**R3**) within the data bias module, as detailed in section V-B1 and section V-B2, respectively. The calculations are re-performed as required, in response to user interactions. During the interactive analysis phase, results from the bias calculation phase are utilized to perform fairness analysis through three specialized views: the Fairness Metric View (**R1**), the Node Selection View (**R2**), and the Diagnostic View (**R3**). Users follow a workflow that begins with node selection, proceeds to fairness inspection, and culminates in diagnostics. This workflow abstracts the typical practices of GNN model developers in conducting fairness analyses, where they manually draw plots to identify and diagnose fairness issues, as summarized by discussions with experts. Additionally, as the analysis progresses through multiple such rounds, users can opt to bypass fairness inspection and focus more on diagnostics. After fairness analysis, users can mitigate model bias based on the gained insights.

V. Bias Calculation

This section describes the techniques used to calculate model bias (**R1**, **R3**) and data bias, including structural bias (**R2**, **R3**) and attribute bias (**R3**).

A. Model Bias

Model bias occurs when model outputs are discriminative to some sensitive groups. We provide a suite of fairness metrics extended from existing works to measure model bias (**R1**) and methods for measuring the contribution of different parts of data to the overall model bias (**R3**).

1) Fairness Metrics for GNN Node Classification: To provide a comprehensive overview of fairness issues and facilitate users to inspect them, GNNFairViz amalgamates a spectrum of fairness metrics with interactive tools. As discussed in section II-B, we have summarized several fairness notions and metrics, some of which are only suitable for binary cases. Given that binary configurations fail to capture the complexities of the real world, we develop a suite of fairness metrics by extending these fairness notions to accommodate multi-class node classification and multinary sensitive groups, and use standard deviation and maximum difference to measure them, respectively. Specifically, for the fairness notions (MSP, AP, and OMRP) that use model performance metrics not explicitly defined for each label, we directly measure standard deviation and maximal difference between sensitive groups. For those notions (SP, EOD, EOP) using model performance metrics calculated explicitly using labels, the mean of standard deviation and maximal difference calculated on each label are defined as the corresponding fairness metrics. Then we theoretically establish the equivalence of MSP, AP, and OMRP in multinary contexts, as detailed in Sec. B.1 of the appendix. Similarly, we demonstrate the equivalence of two EOD criteria, as outlined in Sec. B.2 of the appendix. Consequently, we simplify our metric suite by retaining only one form of each set of equivalent metrics, thereby proposing a set of metrics designed for multi-class node classification scenarios involving multiple sensitive groups, which is listed in table I.

Essentially, for a subset of predicted labels, sensitive groups, and ground truth labels, our metrics quantify bias by mapping these inputs to a numerical value—the lower the value, the higher the fairness. Specifically, our metrics quantify the dispersion of classification outcomes across sensitive groups, defined by categories of a single sensitive attribute or intersections of multiple sensitive attributes, as shown in fig. 2.

However, to truly understand the extent of bias, merely identifying metric values is insufficient without knowledge of their possible ranges. To address this, we offer strict mathematical proofs for these ranges, including the conditions for achieving their maximum and minimum values, detailed in Sec. B.3 - Sec. B.6 in the appendix. The ranges and definitions of the metrics are detailed in table I.

2) Bias Contribution: We measure the contributions of graph structure and attributes of a subset of nodes $V_s \subset V$ to model bias by investigating "what-if" scenarios, similar to counterfactual reasoning [39], where the question is how the outcome would have differed if the input had been different in a specific way. The processes are shown in fig. 5.

Bias Contribution of Graph Structure of a Node Set. We propose assessing how the structure of V_s contributes to model bias by answering the question: *How significantly does the bias of a GNN model's output alter when the structure of V_s is removed?* In this context, the "structure" of V_s refers to all edges connected to any vertex in V_s. To measure this, we identify the edges associated with vertices in V_s as E_{V_s}. We then construct a modified graph $G'_S = (V, E \setminus E_{V_s}, X)$ by excluding E_{V_s} from the original graph's edge set E. The model then processes G'_S for node classification, producing outputs denoted by $\hat{Y}_{G'_S}$. The change in model bias due to the exclusion

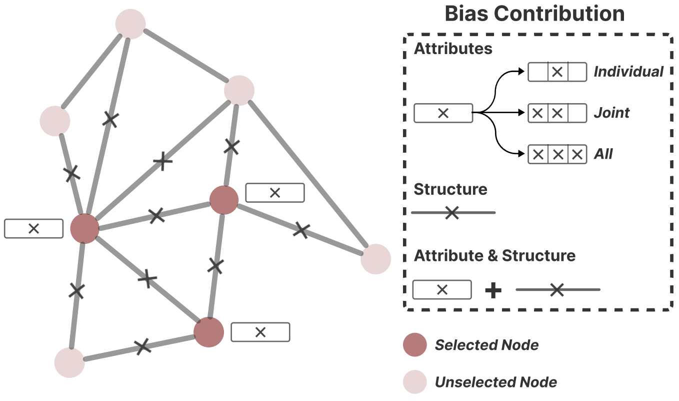

Fig. 5. We calculate the bias contributions of five aspects (individual attributes, a set of attributes, all attributes, graph structure, and the combination of all attributes and graph structure) for a set of selected nodes. These contributions are measured by the difference in model output bias before and after removing certain parts of information from the input data.

of V_s's structure is quantified by the difference between the bias measure of $\hat{Y}$ and $\hat{Y}_{G'_S}$:

$$BC_S(V_s) = \Delta Bias_g(\hat{Y}, \hat{Y}_{G'_S}) = Bias_g(\hat{Y}) - Bias_g(\hat{Y}_{G'_S}),$$

where the bias measure $Bias(\cdot)$ is calculated by a) firstly partitioning the output based on sensitive groups, b) then estimating the distributions $pdf_i, i = 1,2,...,m$ of the output of each group g_i using kernel density estimation (KDE) [40], c) finally calculating the distance between $pdf_i, i = 1,2,...,m$.

In our implementation, Jensen-Shannon Distance [41] is used to measure the distance between two distributions, and the average of distance values between all group pairs of output distributions is used as the bias measure:

$$Bias_g(\tilde{Y}) = \frac{2}{m(m-1)} \sum_{j,k=1,2,...,m, j\neq k} JSD(pdf_j, pdf_k),$$

where $\tilde{Y}$ represent the output of a model, and $JSD(\cdot,\cdot)$ is the Jensen-Shannon Distance:

$$\mathrm{JSD}(P,Q) = \sqrt{\frac{1}{2}D(P\|M) + \frac{1}{2}D(Q\|M)}$$

where D represents the Kullback-Leibler divergence [42], and $M = \frac{1}{2}(P+Q)$ is the average of distributions P and Q. The Jensen-Shannon Distance is more effective in measuring bias contribution from a node set's structure than the metrics in table I, as it directly captures changes in output probabilities caused by edge removal during inference. In contrast, these probability changes may not necessarily be reflected in the predicted labels, rendering label-based metrics less sensitive to the modifications. However, estimating Jensen-Shannon Distance can be computationally intensive due to the need to sample from distributions, given the multi-dimensional nature of model outputs. To this end, we adopt an importance sampling strategy by using outputs $\tilde{Y}_{g_j}$ and $\tilde{Y}_{g_k}$ of groups g_j and g_k as sampled points when estimating $JSD(\tilde{Y}_{g_j}, \tilde{Y}_{g_k})$.

Bias Contribution of Attributes of a Node Set. Following the same logic used to assess the contribution of the graph structure of V_s to model bias, we measure the contribution of node attributes in V_s to model bias by answering the question: *How significantly does the bias of a GNN model's*

TABLE I
THE FORMULAS OF FAIRNESS METRICS USED IN OUR APPROACH. THE (ASYMPTOTIC) RANGES OF THE METRICS ARE ALL $[0,1]$, THE FORMAL MATHEMATICAL PROOF OF WHICH ARE DETAILED IN THE APPENDIX. HERE, m DENOTES THE NUMBER OF GROUPS, AND c DENOTES THE NUMBER OF LABELS. IN THE FORMULAS, $TPR_{ij} = P(\hat{y} = l_i | y = l_i, g = g_j)$, $FPR_{ij} = P(\hat{y} = l_i | y \neq l_i, g = g_j)$, l_i REPRESENTS THE i-TH LABEL, AND g_j REPRESENTS THE j-TH SENSITIVE GROUP.

Metric	**Formula**	**Notations**
Equality of Opportunity (EOP)	$\Delta_{\text{std}} = \frac{2}{c}\sum_{i=1}^{c}\sqrt{\frac{1}{m}\sum_{j=1}^{m}(TPR_{ij} - \mu_i^{TPR})^2}$ (a)	$\mu_i^{TPR} = \frac{1}{m}\sum_{j=1}^{m} TPR_{ij}$
	$\Delta_{\max} = \frac{1}{c}\sum_{i=1}^{c}\max_{g_j,g_k\in\mathscr{G}}\left|TPR_{ij} - TPR_{ik}\right|$ (b)	
Statistical Parity (SP)	$\Delta_{\text{std}} = \frac{2}{c}\sum_{i=1}^{c}\sqrt{\frac{1}{m}\sum_{j=1}^{m}(P(\hat{y} = l_i | g = g_j) - \mu_i^{P})^2}$ (c)	$\mu_i^{P} = \frac{1}{m}\sum_{j=1}^{m} P(\hat{y} = l_i | g = g_j)$
	$\Delta_{\max} = \frac{1}{c}\sum_{i=1}^{c}\max_{g_j,g_k\in\mathscr{G}}\left|P(\hat{y} = l_i | g = g_j) - P(\hat{y} = l_i | g = g_k)\right|$ (d)	
Equality of Odds (EOD)	$\Delta_{\text{std}} = \frac{1}{c}\sum_{i=1}^{c}(\sqrt{\frac{1}{m}\sum_{j=1}^{m}(TPR_{ij} - \mu_i^{TPR})^2} + \sqrt{\frac{1}{m}\sum_{j=1}^{m}(FPR_{ij} - \mu_i^{FPR})^2})$ (e)	$\mu_i^{TPR} = \frac{1}{m}\sum_{j=1}^{m} TPR_{ij}$, $\mu_i^{FPR} = \frac{1}{m}\sum_{j=1}^{m} FPR_{ij}$
	$\Delta_{\max} = \frac{1}{2c}\sum_{i=1}^{c}(\max_{g_j,g_k\in\mathscr{G}}\left|TPR_{ij} - TPR_{ik}\right| + \max_{g_j,g_k\in\mathscr{G}}\left|FPR_{ij} - FPR_{ik}\right|)$ (f)	
Accuracy Parity	$\Delta_{\text{std}} = 2\sqrt{\frac{1}{m}\sum_{i=1}^{m}(\text{Acc}_i - \mu^{Acc})^2}$ (i)	Acc_i: Accuracy for group g_i, $\mu^{Acc} = \frac{1}{m}\sum_{i=1}^{m} Acc_i$
	$\Delta_{\max} = \max_{i,j=1,2,\ldots,m}\left|\text{Acc}_i - \text{Acc}_j\right|$ (j)	

output alter when the information of certain attributes of V_s is obscured? Unlike edges, node attributes cannot be directly removed prior the inference process of the original GNN model. Therefore, we replace the attributes of V_s with their mean values. Formally, this manipulation yields a new attribute matrix, X'_A, where:

$$X[i_v, i] = Mean(X[:, i]), \forall v \in V_s, \forall i = 1, 2, ..., d.$$

The modified graph is $G'_A = (V, E, X'_A)$. Then G'_A is fed into the GNN model for node classification, resulting the output $\hat{Y}_{G'_A}$. The difference between the bias measure of $\hat{Y}$ and $\hat{Y}_{G'_A}$ is used as the bias contribution of attributes of V_s:

$$BC_A(V_s) = \Delta Bias_g(\hat{Y}, \hat{Y}_{G'_A}) = Bias_g(\hat{Y}) - Bias_g(\hat{Y}_{G'_A}).$$

For one single attribute a, the bias contribution of a of V_s is measured by:

$$BC_a(V_s) = \Delta Bias_g(\hat{Y}, \hat{Y}_{G'_a}) = Bias_g(\hat{Y}) - Bias_g(\hat{Y}_{G'_a}),$$

where $\hat{Y}_{G'_a}$ is the output of the GNN model on the graph $G'_a = (V, E, X'_a)$, in which X'_a is created by:

$$X[i_v, i_a] = Mean(X[:, i_a]), \forall v \in V_s.$$

Similarly, the bias contribution of an attribute set $A_s = \{a_{s_1}, a_{s_2}, ..., a_{s_k}\} \subset A$ is:

$$BC_{A_s}(V_s) = \Delta Bias_g(\hat{Y}, \hat{Y}_{G'_{A_s}}) = Bias_g(\hat{Y}) - Bias_g(\hat{Y}_{G'_{A_s}}),$$

where $\hat{Y}_{G'_{A_s}}$ is the output of the GNN model on the graph $G'_{A_s} = (V, E, X'_{A_s})$, in which X'_{A_s} is created by:

$$X[i_v, s_i] = Mean(X[:, s_i]), \forall v \in V_s, \forall i = 1, 2, ..., k.$$

Joint Bias Contribution of Attributes and Graph Structure of a Node Set. We assess the joint contribution of all attributes and graph structure of V_s by answering the question: *How significantly does the bias of a GNN model's output alter when the structure and information of all attributes of V_s is obscured?* Specifically, we create a graph $G'_{S\&A} = (V, E \setminus E_{V_s}, X'_A)$, and use the output $\hat{Y}_{G'_{S\&A}}$ of the model on $G'_{S\&A}$ and $\hat{Y}$ to compute the joint bias contribution:

$$BC_{S\&A}(V_s) = \Delta Bias_g(\hat{Y}, \hat{Y}_{G'_{S\&A}}) = Bias_g(\hat{Y}) - Bias_g(\hat{Y}_{G'_{S\&A}}).$$

B. Data Bias

The bias in data is divided into structural and attribute bias, where structural bias distorts the effect of attribute bias to model bias.

1) Structural Bias: Structural bias may inherently emphasize or de-emphasize specific nodes or groups based on their topological properties. To identify it, we employ a dense subgraph detection algorithm to identify densely connected subgraphs (**R2**), which suggest the presence of uneven structure, providing recommendations for potential structural bias. Additionally, we compute the count of neighbors at various hop distances in each node's computational graph (**R2**). Finally, we measure the connectivity between selected nodes and all nodes in different sensitive groups (**R3**).

Dense Subgraph Detection. To provide users with clues regarding potential structural bias, GNNFairViz employs a dense subgraph detection algorithm to reveal dense subgraphs. In our implementation, we adopt the technique described in [43], which is selected for its computational efficiency and its flexibility in not requiring a predetermined count of dense subgraphs—a figure often unknown and challenging to estimate. Furthermore, this method enables partial clustering, recognizing that not all nodes must be part of a subgraph. This feature allows for a more realistic grouping of vertices where only a subset may be clustered. Specifically, given an undirected graph $G = (V, E)$ and its adjacency matrix A, the process first constructs a weighted adjacency matrix M based on cosine similarities between graph vertices:

$$M[i, j] = \frac{\langle A[:, i], A[:, j]\rangle}{||A[:, i]||\,||A[:, j]||}.$$

Then for a weighted graph where M represents its adjacency matrix , a top-down hierarchical clustering of vertices V is executed by progressively removing edges, prioritizing those with the lowest weights first. Finally, the resulted subgraphs $G'_i = (V'_i, E'_i), i = 1,2,...,l$, satisfy:

$$d_{G'_i} = \frac{2|E'_i|}{|V|(|V|-1)} \geq d_{thr},$$

where d_{thr} is the user-specified density threshold.

Calculating Number of Neighbors in Computational Graphs. Also as recommendations of potential structural bias, nodes with different levels of impact might have different roles in introducing model bias. Generally, the number of neighbors a node has in the original graph is utilized to measure the node's impact in the graph. However, the number does not accurately represent a node's contribution during the message-passing process of GNNs. To depict the influence of each node in the message-passing process, we calculate the number of neighbors each node possesses in its computational graph. This number directly represents the frequency with which a node's information is propagated (as shown in fig. 3, node A has 3 and 1 neighbors at 1-hop and 2-hop distances respectively in the original graph, but in its computational graph, the corresponding numbers are to 3 and 7, highlighting a substantial difference). Conversely, it also quantifies the amount of information aggregated into a node's embedding. Building on this understanding, we compute the number of neighbors each node has at successive hop distances. The calculation utilizes the adjacency matrix $\mathbf{A}$ of the graph. We begin by calculating the direct neighbors (1-hop) for each node by summing the entries of each row in $\mathbf{A}$, yielding the initial degree vector $\mathbf{d}^{(1)}$. For each subsequent hop up to a predefined maximum h_{max}, the number of neighbors at i-hops is determined iteratively:

$$\mathbf{d}^{(i)} = \mathbf{A} \cdot \mathbf{d}^{(i-1)}, i = 2,...,h_{max},$$

where each $\mathbf{d}^{(i)}$ indicates the i-hop neighborhood size for each node.

Calculating Connectivity between Sensitive Groups. Given a node set $V_s \subset V$, we measure the impact of each sensitive group within the selected nodes on groups across the entire dataset, and vice versa, by their connectivity. This is quantified by summing the number of neighbors of nodes in their computational graphs for each sensitive group. Such measurements directly reveal the structural bias, which is crucial for our analysis of GNN fairness. Specifically, the process begins by filtering A to retain only the rows corresponding to V_s, producing A_{V_s}. For the initial level of connectivity ($k = 1$), we calculate the sum of the rows in A_{V_s} that correspond to each sensitive group $g_i \in \mathscr{G}$, creating a $m \times n$ matrix S_1 by:

$$S_1[i,:] = \sum_{v \in N_{V_s}(g_i)} A_{V_s}[i_v,:], \forall g_i \in \mathscr{G},$$

where $N_{V_s}(g_i)$ are the nodes in group g_i within V_s, and i_v is the index of node v. For subsequent levels of connectivity (k>1), we propagate the connectivity information through the graph by matrix multiplication, using:

$$S_k = S_{k-1} \cdot A, k = 2,...,h_{max}.$$

To summarize the group-level connectivity at each level k, we perform column operations for each matrix S_k. For each group g_i, we extract columns corresponding to the nodes in g_i from S_k, and compute the total connectivity from each group of selected nodes to g_i by summing these columns:

$$C_{kg}[:,i] = \sum_{v \in N(g_i)} S_k[:,i_v], \forall g_i \in \mathscr{G},$$

where $N(g_i)$ are the nodes in group g_i. This results in a set of tuples (source group, destination group, connectivity strength) that quantitatively describe the connectivity patterns among the groups at each hop distance up to the maximum specified hops h_{max}.

2) Attribute Bias: Given a node set $V_s \subset V$, GNNFairViz employs a multifaceted statistical analysis to detect biases in each attribute based on sensitive groups $\mathscr{G}$ (**R3**). The approach is twofold:

One-hot or binary encoded attributes. For a binary attribute transformed from a categorical variable via one-hot or binary encoding methods, we apply the chi-square test to examine the independence between attribute values and sensitive groups. The outcome of this test serves as an indicator of whether there is bias in the distribution of this attribute. Moreover, within each sensitive group $g_i \in \mathscr{G}$, the node set V_s is divided into those that are members of g_i and those that are not. Following this, m chi-square tests are conducted to ascertain whether the distributions of attribute values within each sensitive group exhibits a significant difference when compared to the rest of the nodes.

Other attributes. For an attribute that is not binary, the Kruskal-Wallis H-test is used to test the difference in distributions of attribute values across the sensitive groups. This is succeeded by m Mann-Whitney U tests, each contrasting a pair of node distributions — those within a particular group g_i and those external, for $i = 1,2,...,m$, to detect notable disparities in distributions among the each group versus the other.

Resulting p-values are compared against a significance level α, which is set to 0.05 in our implementation, to identify significant biases in attribute distributions concerning sensitive groups.

VI. GNNFairViz

In this section, we discuss the visual analysis tool tailored to the design requirements. We detail the design of each interface component, the design choices, the user interactions, and the tool's implementation.

A. Node Selection View

To assist users in pinpointing node sets that might be influential to model bias (**R1**), we develop the Node Selection View (fig. 1, b) to visualize Node Embeddings (fig. 1, b$_1$), Number of Neighbors in Computational Graphs (fig. 1, b$_2$), and Dense Subgraphs (fig. 1, b$_3$).

Node Embeddings. The Node Embeddings plot visualizes the node embeddings generated by the GNN model, which are derived from both node attributes and the graph structure, while also highlighting the associations with sensitive groups. It presents the spatial relationships and clustering patterns of node embeddings from a user-specified embedding layer of

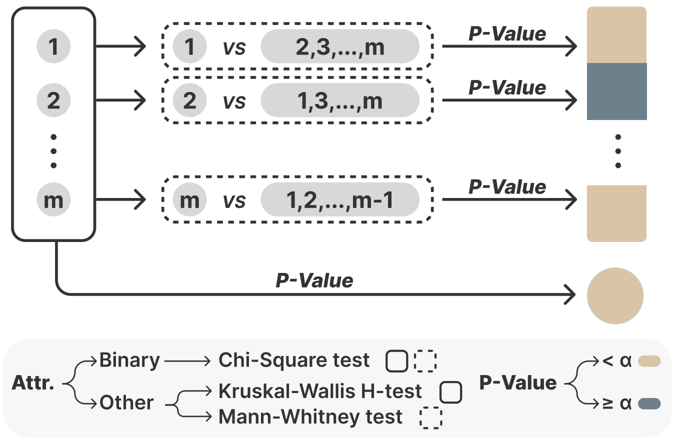

Fig. 6. Illustration of attribute bias calculation and its correspondence to the columns of glyphs in the Attribute Overview plot of the Diagnostic View. With m denoting the number of sensitive groups, for a given attribute, a solid-outlined square represents a multigroup comparison test among the m groups, the result of which is displayed as a round shape. A dotted-outlined square represents a pairwise comparison test between a specific group and the group comprising the remaining nodes, with the result shown as a square. A significant distribution difference between the tested groups, indicated by a p-value smaller than $\alpha = 0.05$, is represented by tan, while non-significant results are depicted in grayish blue.

GNN models, offering insights into the differential treatment of nodes across different sensitive groups through the lens of node embeddings. Users can project embedding spaces into two dimensions using PCA, t-SNE, or UMAP. After dimensionality reduction, nodes are shown in a color-coded scatter plot based on sensitive groups. To reduce visual clutter in large datasets, users can reduce the number of displayed nodes via random sampling, which preserves region and class density [44]. The rangeset technique [45] further aids in identifying clusters. Together, these techniques assist in highlighting disparities in node embedding distributions across sensitive groups.

Number of Neighbors in Computational Graphs. To provide users with clues for node selection by depicting the impact of each node on and from other nodes during the message-passing process, the plot of Number of Neighbors in Computational Graphs displays the distribution of number of neighbors within the computational graph for each node at a user-specified hop (calculated in the structural bias module as described in section V-B1) using a histogram. The overall neighbor count distribution is shown as hollow black frame bins, and the count distribution of selected nodes is shown as grey bins.

Dense Subgraphs. To offer recommendations of potential structural bias for node selection, dense subgraphs are identified using the dense subgraph detection algorithm described in section V-B1, and their sizes—measured by the node count—are calculated, both in the attribute bias module. Each dense subgraph is represented as a point in a scatter plot, with users able to adjust the density threshold of the detection algorithm via the Control Panel (fig. 1, a).

B. Fairness Metric View

The Fairness Metric View (fig. 1, c) visualizes the calculated fairness metrics in the model bias module (section V-A1). It consists of two plots: Fairness Metrics (fig. 1, c_1) and Detail of Selected Metric (fig. 1, c_2), supporting users to inspect GNN model bias (**R2**).

Fairness Metrics. The Fairness Metrics plot supports users to inspect the model bias by the examination of fairness metrics including SP, EOP, EOD, and AP (listed in table I). These metrics are visualized through a bar chart, where the metric values are presented as solid gray bars, and their ranges are depicted as transparent bars with black outlines.

Detail of Selected Metric. Further details for understanding the calculation of each metric can be explored in a dedicated plot by clicking on the corresponding bars. The visualization charts vary according to the metrics' calculation. For EOP and SP, the metrics are calculated based on labels and sensitive groups, following definitions (a), (b), (c), and (d) in table I. Visualization is achieved through a heat map, with the x-axis for labels, the y-axis for sensitive groups, and the color intensity in each cell representing the values of TPR_{ij} for EOP, or $P(\hat{y} = l_i | g = g_j)$ for SP. Similarly, for EOD, which is based on the TPR and FPR for nodes categorized by sensitive groups on each label (as per definitions (e) and (f) in table I), two separate heatmaps are used to visualize the details of TPR and FPR parts respectively. The AP, derived from the accuracy within each sensitive group (according to definitions (i) and (j) in table I), is visualized using a bar chart representing accuracy value of each sensitive group for its details. By inspecting the details of metrics, users can understand why the metric values are high or low and whether the values actually reflect model bias.

C. Diagnostic View

To help users analyze the cause of model bias from the perspective of data bias (**R3**), we design the Diagnostic View (fig. 1, d). The Diagnostic View consists of five components, i.e., Bias Contributions (fig. 1, d_1), Attribute Overview (fig. 1, d_2), Connectivity between Sensitive Groups (fig. 1, d_3), Attribute Distribution in Each Sensitive Group (fig. 1, d_4), and Relationship between Attributes and Number of Neighbors (fig. 1, d_5).

Bias Contributions. For the selected nodes, we calculate the bias contributions of attributes, structure, the combined effect of attributes and structure, and the summative contribution of individual attributes, in the model bias module (described in section V-A2). The results are precisely displayed in text, showing the respective contribution values.

Attribute Overview. The Attribute Overview plot is designed to offer users an overview of attribute bias, and support further exploration through interactions on it. It visualizes the result of the attribute bias module (section V-B2), as well as bias contributions of attributes from the model bias module (section V-A2). It consists of three main visual components, arranged vertically. The first component at the top is a grid where unique sensitive groups are displayed as rows and attributes as columns. Each cell in the grid contains a background and a rectangle block (as shown in fig. 10, d_2'). The cell's height shows the total number of nodes in a sensitive group, while the rectangle block's height indicates the number of selected nodes in that group. Cell backgrounds are semi-transparent, colored dark yellow (biased) or dark blue (unbiased) based on statistical tests (as detailed in section V-B2). Similarly, rectangle block colors indicate whether selected nodes are biased. Beneath the grid is a row of glyphs, each with an inner circle and a crescent. The circle's color shows if the attribute (column) for selected nodes is biased, based on statistical tests (cf. section V-B2). The crescent represents the attribute's bias contribution, with its radius (up to 180°) indicating magnitude and its position (top or bottom) showing positive or negative contribution. The visual setup allows users to quickly assess attribute bias at a glance.

The third part is user-generated through interaction with the plot. In "Multiple" selection mode, activated via a radio button, users can select attributes by clicking within columns to examine the combined bias contribution of selected nodes. Glyphs with a black inner circle and crescent, representing the attribute set's bias contribution, are added to the selected columns (see fig. 10, d_2'), calculated as per section V-A2. Clicking the "New Selection" button lets users create new combinations, adding rows for further analysis. This interactive feature enables exploration of attribute interactions beyond individual contributions, overcoming the infeasibility of computing all 2^d combinations by allowing selective, knowledge-driven comparisons.

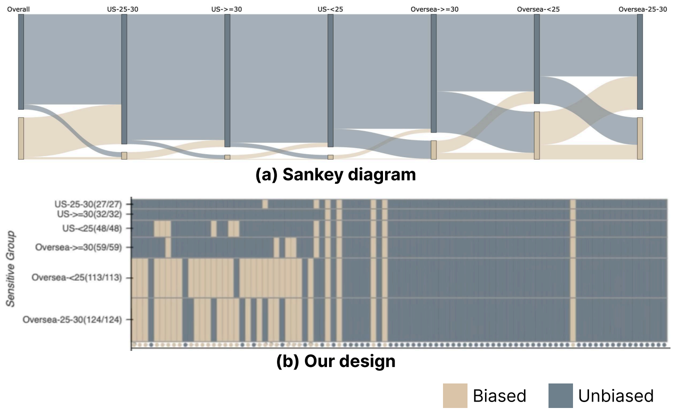

Fig. 7. Design choices for the Attribute Overview plot. (a) A Sankey diagram-based alternative design where each pair of vertical blue and tan bars represents the overall dataset or a sensitive group of nodes, and the height represents the total number of node attributes. (b) Our final design.

Connectivity between Sensitive Groups. With a user-specified hop distance, the connectivity between sensitive groups in selected nodes and groups in all nodes (calculated in the structural bias module as described in section V-B1) is displayed as a heat map plot. This plot directly reveals the potential structural bias patterns concerning sensitive groups, helping understanding how structural bias distorts the impact of attribute bias.

Attribute Distribution in Each Sensitive Group. When an attribute is selected in the Attribute Overview plot using "Single" selection mode, this plot displays its distribution across each sensitive group based on both node attributes and sensitive attributes. This visualization helps users understand the specific forms of attribute bias. For one-hot or binary encoded attributes, distributions are shown in a bar chart (fig. 1, d_4), where each bar represents the count of an attribute value within a sensitive group, with bar frames showing total node counts and grey fillings indicating selected node counts. For other types of attributes, a split violin plot (fig. 10, d_4) is used, where each split violin represents the attribute distribution within a sensitive group, with the green upper half showing selected nodes and the pink lower half showing all nodes. This comparative perspective between the overall attribute distribution and the distribution among selected nodes is crucial for evaluating how the selected nodes influence attribute bias—whether they exacerbate or mitigate it.

Relationship between Attributes and Number of Neighbors. This plot helps users inspect the interaction between the graph structure and attributes, i.e., whether edges amplify or mitigate the impact of bias in a certain attribute. In the "Single" selection mode, the relationship between values of the attribute selected in the Attribute Overview plot and the number of neighbors in computational graphs is displayed, based on node attributes and the calcualation of number of neighbors in the structural bias module (section V-B1). For one-hot or binary attributes, distributions are shown using a split violin plot (fig. 1, d_5), where each violin represents the number of neighbors for an attribute value: the green upper half shows selected nodes, and the pink lower half shows all nodes. For other attributes, a scatter plot overlays a hexbin plot, with the hexbin plot summarizing relationships for all nodes and the scatter plot, optionally sampled, highlighting selected nodes.

D. Design Choices

As highlighted by our collaborating experts, particularly those involved in the GNN and ML communities, adopting standard and widely-used visualizations is beneficial for lowering the barrier for GNN model developers to utilize GNNFairViz. Therefore, all of our visualizations are either standard or augmented from standard visualization techniques, with the exception of the Attribute Overview plot. Initially, we considered using a Sankey diagram (fig. 7, (a)) to visualize attribute bias. In this design, each pair of vertical blue and tan bars represents either the entire dataset or a specific sensitive group of nodes, with the bar heights corresponding to the total number of node attributes. This approach provides an overview of the proportion of biased and unbiased attributes across the entire dataset or within specific sensitive groups by clustering attributes along each axis. However, because the number of nodes varies among groups, this visualization fails to represent the actual attribute bias proportions. In contrast, our design (fig. 7, (b)) effectively conveys the proportion of biased and unbiased attributes through the area of color-coded patches. Additionally, while the Sankey diagram requires multiple clicks on its axes to locate specific attributes, our design enables quick attribute identification through simple interaction, such as zooming and panning. Consequently, we opted for our current design.

E. User Interactions

In addition to the interactions that have been introduced in the view introduction, GNNFairViz integrates a range of interactions to enhance user engagement with visualization charts and support the analysis process, as outlined below:

Convenient Interactions. All charts enable users to zoom in and out through scrolling or selecting a specific area, and to drag across the plot. Additionally, hovering over certain plots reveals more detailed information.

Configurations. Users can adjust settings in the Control Panel (fig. 1, a). Global Settings include toggling between original and logarithmic scales for neighbor displays. The Node Embeddings plot settings allow selecting the projection algorithm, sample size, and rangeset threshold. The Dense Subgraphs plot settings enable adjusting the density threshold.

Sensitive Attribute and Hop Selection. In the Control Panel's Global Settings, users can select sensitive attributes to define sensitive groups, aiding analysis from a user-defined perspective (**R1**). Related charts update automatically (red arrows, fig. 8). Users can also choose a hop (1 to the maximum aggregated by the GNN model), updating hop-related charts (green arrows, fig. 8, **R2**, **R3**).

Node Selection (R2). In the Node Selection View, users can select nodes via lasso or box tools in the Node Embeddings plot, x-axis bin selection in the Number of Neighbors plot, or by clicking points in the Dense Subgraphs plot to reveal and choose subgraphs from a dropdown. Multiple selections are supported, with final choices confirmed in the Control Panel, triggering updates across related plots (yellow arrows, fig. 8).

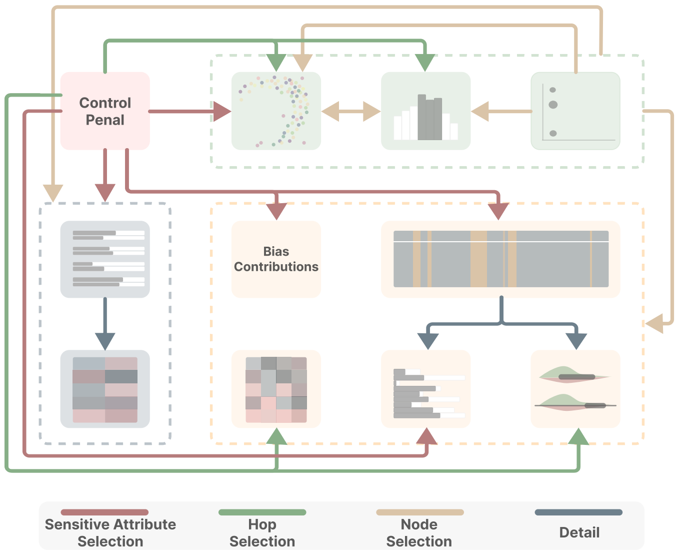

Fig. 8. Interactions between the plots. The placement of plots corresponds to the interface of GNNFairViz. **Red arrows**: Selecting a sensitive attribute updates related charts. **Green arrows**: Choosing neighborhood hops updates neighbor-related charts and the Node Embeddings plot. **Tan arrows**: Selecting nodes updates target plots with their information. **Blue arrows**: Clicking on Attribute Overview shows details of the selected attribute in target plots.

F. Implementation

GNNFairViz is a Python package developed with Bokeh and HoloViz for visualization and Pytorch, DGL, Numpy, and Pandas for data processing, aiming for seamless integration into users' development workflow in Jupyter Notebooks. Users can instantiate a Python object with a single line of code by providing a GNN model and data as parameters. The input data for GNNFairViz includes not only node attributes and graph structure but also sensitive attributes. While in principle, all attributes could be sensitive, in practice only specific attributes are considered sensitive. We assume users have already identified the potentially sensitive attributes, which is typically the case in real-world applications. With the created Python object, an interface can also be created with one line of code. It allows users to record bias-contributing factors during fairness analysis via the Control Panel. These records, retrievable with the *get_records* method, are essential for mitigating biases post-analysis.

VII. Evaluation

We present two usage scenarios and semi-structured expert interviews to assess the effectiveness and usability of our approach in facilitating GNN fairness analysis. For the usage scenarios, We demonstrate them through the perspective of Sam, a GNN model developers navigating his GNN development workflow, who aims to analyze potential fairness issues in the GNN models.

A. Usage Scenario 1: Age Fairness in Default Prediction

This usage scenario demonstrates how a GNN model developer integrates our approach into their development workflow to analyze potential fairness issues in GNN models and chooses the appropriate architecture for a fairer model based on insights from the fairness analysis. In this scenario, Sam selects the Credit Defaulter dataset. In the Credit Defaulter dataset, there are 30,000 nodes, with each node representing a credit card user. An edge between two nodes signifies a connection between credit card users based on the similarity of their purchase and payment patterns. Sam uses age (whether the users are older than 25) as the sensitive attribute. He trained a GAT model on this dataset in a Jupyter Notebook, with the node classification task being to predict future credit card payment defaults. The GAT model consists of 2 GAT layers followed by an MLP layer. He aims to assess any potential biases within the model with respect to users' age, understand their origins, and try to mitigate these biases based on his findings. Thus, he imports the Python package of GNNFairViz and initializes an object by passing the data and trained model as arguments to the class constructor, then creates the interface using the object's method—all in just two lines of code. This usage scenario shows how Sam iteratively revisits the workflow through multiple rounds.

Gaining initial insights through GNNFairViz interface (Round 1). Once the interface is created, all nodes are selected by default. Fairness metrics are automatically calculated on all nodes, with the results displayed in the plot of Fairness Metrics in the Fairness Metric View (fig. 1, c_1). By inspecting the fairness metric values, Sam observes that there exists model bias to some extent, particularly when it comes to metrics of EOP and SP (which are higher than 0.06). He then clicks on bars to check the details of these metrics. In the detail of SP metrics (fig. 1, c_2), he discovers that the model tends to predict users younger than 25 as defaulter (label 0).

Sam then seeks to understand the underlying reasons for model bias. He examines the Bias Contributions (fig. 1, d_1) in the Diagnostic View and notices that all attributes jointly contribute 100% to model bias. While the graph structure contributes negatively to the model bias, thereby promoting model fairness. Furthermore, the combined bias contribution of attributes and graph structure is also 100%. This implies that when all attribute information is removed, the structure alone does not enable the model to differentiate between users of different age groups.

Afterward, Sam aims to understand the role of attributes. In the Attribute Overview plot (fig. 1, d_2), he identifies that the attribute "HistoryOfOverduePayments" contributes 52.6% to model bias individually, a contribution significantly higher than that of other attributes. Consequently, Sam clicks on the column of this attribute to inspect its details to further explore its mechanism of action. By exploring the plot of Attribute Distribution in Each Sensitive Group (fig. 1, d_4), Sam observes that the distributions of this attribute between two sensitive groups are markedly different. This suggests that the high bias contribution of this attribute is due to its highly biased distribution with respect to age group. Additionally, Sam inspects the plot of Relationship between Attributes and Number of Neighbors (fig. 1, d_5), opting to display the number of neighbors on a logarithmic scale due to its long-tail distribution. He finds no significant relationship between them. Therefore, Sam concludes that the bias of the "HistoryOfOverduePayments" attribute plays a major role in contributing to model bias, and the graph structure does not amplify this effect significantly.

Sam proceeds to analyze the effect of graph structure. By exploring Connectivity between Sensitive Groups (fig. 1, d_3), he discovers that the graph data is not entirely homophilic. Specifically, nodes in the age group of "<=25" connect significantly more (approximately five times) with nodes in the ">25" age group than with nodes in their own age group. This observation explains the negative bias contribution of graph structure: the impact of the difference in attribute distributions between the two age groups might be amplified by the numerous edges between nodes in the ">25" age group. Intuitively, one might expect the graph structure to have a positive bias contribution. However, because the "<=25" age group has about five times more connections to the ">25" age group than to itself, the information aggregated from the ">25" group is also approximately five times greater than that from the "<=25" group. This effect results in the attributes of nodes in the ">25" age group dominating the node embeddings of the "<=25" age group, with about $\frac{5}{6}$ of the neighborhood information encoded into the node embeddings of the "<=25" group coming from the attributes of nodes in the ">25" group. Conversely, the ">25" age group has a larger number of intra-group connections than inter-group connections, ensuring that the attributes of nodes within this group predominantly influence their node embeddings. **Therefore, the presence of the graph structure reduces the differences in node embeddings between different age groups, thereby promoting model fairness.**

Verifying insights through GNNFairViz interface (Round 2). To further verify this insight, Sam selects nodes in the three large dense subgraphs identified as outliers in the Dense Subgraph plot (fig. 1, b_1), using a minimal density threshold of 0.2. After selection, he adjusts the sample size and threshold in the Control Panel to refine the Node Embeddings plot (fig. 1, b_1). He then discovers that most of the selected node belong to the ">25" age group and are primarily located in areas where the node embeddings of the two age groups do not overlap. In the Fairness Metric View, the fairness metrics updated on this node set are higher than the metrics for all nodes (fig. 1, c'_1). These observations suggest that these nodes might have a positive impact on the overall model bias. In Attribute Overview plot of the Diagnostic View, Sam notices the majority of the nodes are in the ">25" age group (913 nodes), with only 8 nodes in the "<=25" age group. This indicates that the subgraphs are highly homophilic, suggesting that dense connections may help propagate information within the same group. To further verify this observation, Sam examines the Connectivity between Sensitive Groups (fig. 1, d'_3) and finds that intra-group connections in the ">25" age group are dominant. Additionally, he notices in the Bias Contributions that the structural bias contribution is positive, albeit small. **Therefore, these observations align with the insights he has gained in Round 1.**

TABLE II
COMPARISON BETWEEN SENSITIVE GROUP CONNECTIVITY OF ORIGINAL GRAPH AND GRAPH WITH ATTENTION WEIGHTS OF EACH GAT LAYER.

	Original Adj.		Layer-1 Attn. Weights		Layer-2 Attn. Weights	
	≤ 25	>25	≤ 25	>25	≤ 25	>25
>25	16627	265679	15062	203458	1894	25421
≤ 25	5821	16627	6360	15120	795	1890

Further development with the insights in Jupyter Notebook. After the analysis, Sam gains insights into how the graph structure promotes model fairness. He then realizes that the attention mechanism, which weighs neighbors during the message-passing process, might distort this effect. To analyze whether the attention mechanism amplifies or mitigates the effect, Sam accesses the attention weights of the trained GNN model in his Jupyter Notebook. For each layer, he sums attention weights from different attention heads, then groups and sums the attention weights by the age groups of the edge sources and ends. He compares these results with the connections between groups in the original adjacency matrix, as shown in table II. Sam discovers that the attention mechanism reduces the fairness-promoting effect of graph structure by lowering the connection weight ratio of inter-group to intra-group connections for the "<=25" age group compared to the original ratio. Consequently, Sam decides to train a GCN model, which aggregates neighborhood information without any special weighting mechanism, on the same dataset. After training, he calculates fairness metrics to compare with those of the GAT model, as shown in fig. 9. While the AP metric values show no significant change, all other metric values decrease substantially, indicating that the GCN model is much fairer than the GAT model on this dataset.

B. Usage Scenario 2: Nationality and Age Fairness for NBA Player Salary

This usage scenario shows how a GNN model developer seamlessly integrates GNNFairViz into their development workflow by analyzing fairness in the interface and debiasing

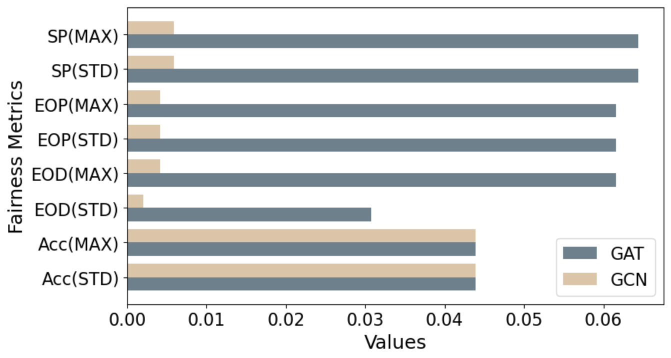

Fig. 9. Comparison of fairness metrics of a GAT and GCN model in usage scenario 1, where the user performs a fairness analysis using GNNFairViz, concludes that the model bias is likely promoted by the attention mechanism of GATs, and then decides to replace the GAT model with a GCN model to mitigate fairness issues. The results signify a significant fairness improvement.

in Jupyter Notebook cells iteratively, the whole process is shown in fig. 10. Here, Sam is analyzing an NBA dataset that includes information about 403 NBA players from the 2016-2017 season, such as their height, salary, and team affiliation. The players are also connected as nodes based on their Twitter relationships. Sam's goal is to predict if a player's salary is above the median. So he trained a GraphSAGE model on it. The model is made up of 2 GraphSAGE layers followed by an MLP layer. He is interested in whether the model discriminates against players of different nationalities and ages. Thus, Sam creates the interface by two lines of Python code.

Exploring the impact of attributes on model bias through GNNFairViz interface (Round 1). Initially, all nodes are, by default, selected. Sam chooses Country ("US" or "Oversea"), Age ("<25", "25-30", ">=30"), and the combination of Country and Age as sensitive attributes in the Control Panel, then assesses Fairness Metrics in the Fairness Metrics View. He finds that the fairness metric values are significantly higher when the sensitive groups are created by the intersection of Age and Country (fig. 10, c_1) compared to using them as individual sensitive attributes. Therefore, with Age and Country as sensitive attributes, Sam decides to analyze the contributing factors to model bias. In the Diagnostic View (fig. 10, d_1), he observes that attributes contribute 100% to model bias, while graph structure negatively contributes by a large margin (119.8%). This indicates that attribute bias is the dominant factor in introducing model bias. Sam then explores the Attribute Overview plot (fig. 10, d_2) and finds that **nearly half of the attribute distributions are biased, which explains why attributes introduce model bias**.

Establishing specific understandings through GNNFairViz interface (Round 2). Sam then aims to identify which specific attributes of certain nodes significantly contribute to model bias. He examines the Node Embeddings in the Node Selection View (fig. 10, b_1). In the plot, he notices that the node embeddings form a ribbon shape after being projected onto a 2D space. At the left end of the ribbon, most nodes belong to the "Oversea->=30" and "Oversea-<25" groups, while the right end consists mainly of nodes from the "Oversea-25-30" and "Oversea->=30" groups. In the middle area, different groups overlap, showing no significant separation. Sam selects the two distinct sets of nodes using box selection. In the Fairness Metric View, it can be observed that the metric values for the selected nodes are higher than those of all nodes (fig. 10, c'_1), which matches the observation that the two clusters have different sensitive group composition.

In the Diagnostic View, Sam notices that the joint bias contribution of all attributes is 91%, which is substantial considering that only 42.93% of the nodes are selected. In the Attribute Overview (fig. 10, d'_2), he observes that the cell colors of the selected nodes (foreground) and all nodes (background) largely match. He hypothesizes **the attribute values of the selected nodes may be amplifying the overall attribute bias**. To verify, Sam identifies four attributes with significantly high bias contributions (0.124, 0.086, 0.118, and 0.139) by zooming in and hovering on the plot. Then, he clicks on each of them successively in "Single" selection mode. In the plot of Attribute Distribution in Each Sensitive Group, he sees that the distributions of these attributes indeed make the differences between attribute distributions of different groups more pronounced. For example, in the case of the "MP" attribute, the distributions for the selected nodes are more skewed in each of the three "Oversea" groups, which are the majority, compared to all nodes, as shown in fig. 10, d_4).

Sam decides to record the contributing attributes. He notices that the summative bias contributions of individual attributes approximately equal the synergistic bias contribution of all attributes, indicating **no significant interaction between attributes in contributing to model bias**. Then he selects the four attributes in "Multiple" selection mode to inspect their joint bias contribution, which is 0.444 as displayed after selection (fig. 10, d'_3). This value closely matches the sum of their individual bias contributions, confirming his insights. Since the other attributes have no significantly large bias contribution, Sam records the four attributes and the selected nodes in the Control Panel.

Analyzing graph structure's role through GNNFairViz interface (Round 3). Sam then wants to understand the role of graph structure in promoting model fairness. He clears the selected nodes in the Control Panel to select all nodes. In the plot of Connectivity between Sensitive Groups (fig. 10, d_3) in the Diagnositc View, he observes a non-homophilic pattern: each diagonal value is smaller than the sum of other values in the same column. Specifically, the three minority groups ("US-25-30", "US->=30", and "US-<25") have many more connections to other groups than to themselves. Sam thus gains **similar insights into why the graph structure has a negative bias contribution as discussed in section VII-A**.

Further verification of insights through GNNFairViz interface (Round 4). To further verify the insights, Sam decides to select nodes with different number of neighbors. He selects the leftmost two bins in the plot of Number of Neighbors in Computational Graphs (fig. 10, b_2) in the Node Selection View. In the Diagnostic View, Sam observes a homophilic pattern where the second diagonal value is larger than the sum of other values in the same column (fig. 10, d_3). This indicates that overall, the "Oversea-<25" group has more connection to other groups within selected nodes than to

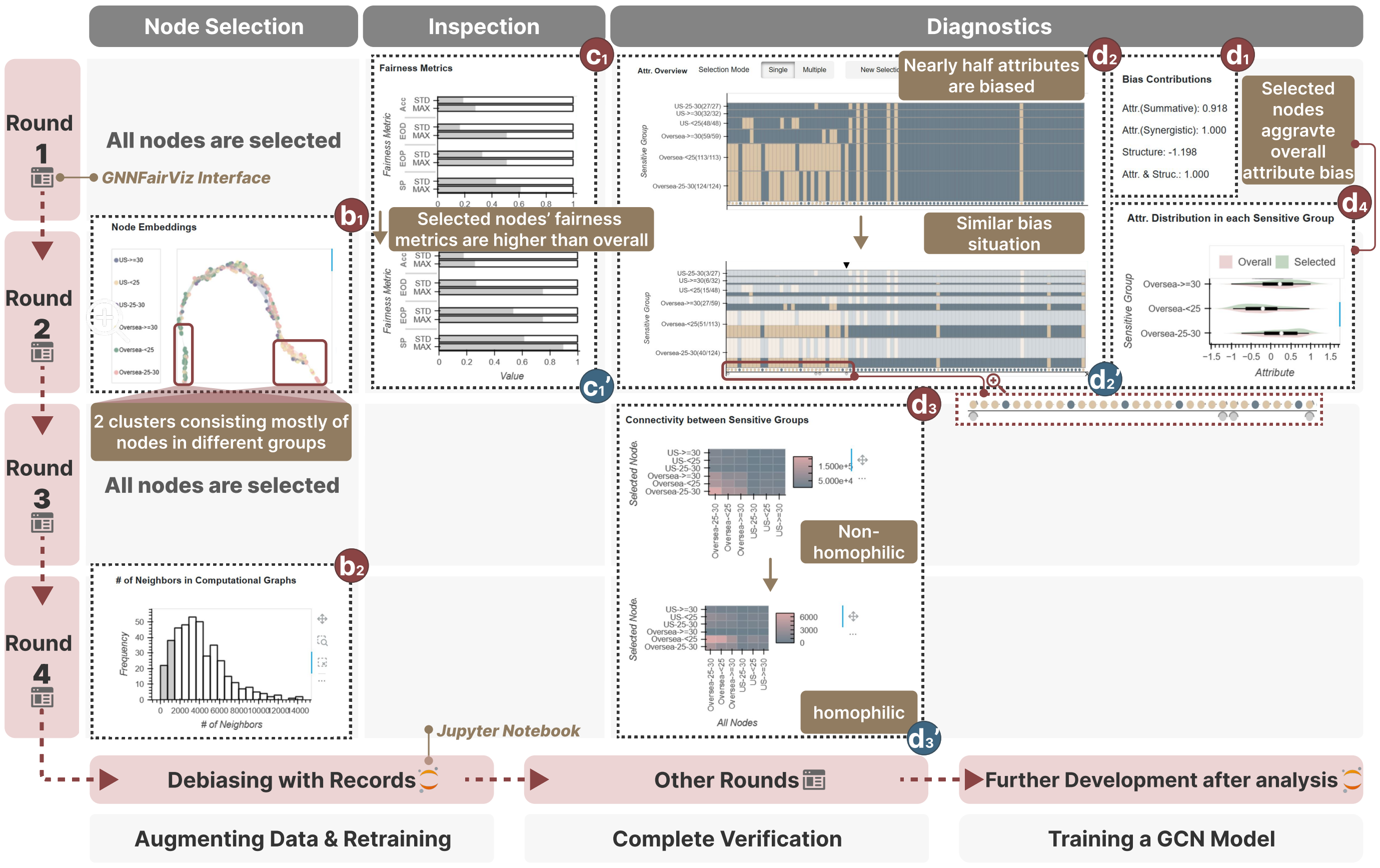
Node Selection
Inspection
Diagnostics
Round 1
Round 2
Round 3
Round 4
All nodes are selected
GNNFairViz Interface
Node Embeddings
2 clusters consisting mostly of nodes in different groups
All nodes are selected
# of Neighbors in Computational Graphs
Jupyter Notebook
Fairness Metrics
Selected nodes' fairness metrics are higher than overall
Nearly half attributes are biased
Similar bias situation
Bias Contributions
Attr.(Summative): 0.918
Attr.(Synergistic): 1.000
Structure: -1.198
Attr. & Struc.: 1.000
Selected nodes aggravte overall attribute bias
Attr. Distribution in each Sensitive Group
Connectivity between Sensitive Groups
Non-homophilic
homophilic
Debiasing with Records
Other Rounds
Further Development after analysis
Augmenting Data & Retraining
Complete Verification
Training a GCN Model

Fig. 10. Summary of usage scenario 2. **Round 1**: (c_1) Significant overall unfairness. (d_1) Graph structure and attributes have negative and positive bias contributions, respectively. (d_2) Nearly half attributes are biased. **Round 2**: (b_1) Two biased clusters of node embeddings. (c_1') The selected nodes' fairness metrics are higher than that of all nodes. (d_2') The bias situation of selected nodes' attributes is similar to all nodes' (a combination of attributes is created). (d_4) Selected nodes aggravate overall attribute bias. **Round 3**: (d_3) A non-homophilic pattern between sensitive groups. **Round 4**: (b_2) Nodes with smallest number of neighbors are seleted. (d_3') A homophilic pattern.

the same group. Furthermore, since nearly half of the selected nodes are in the "Oversea-<25" group and the number of edges in this group is large, Sam believes **they significantly amplify the effect of attribute bias**. To verify that, he checks the bias contribution of graph structure, which is 0.305. With the positive bias contribution confirming the insights, Sam records the edges connecting to the selected nodes in the related "Oversea-<25" group.

Debiasing with records in Jupyter Notebook. After recording the information important to model bias, Sam removes the recorded edges and sets the recorded attribute values to their mean values in the Jupyter Notebook, resulting in a fairness-oriented augmented dataset. Then he retrains the GraphSAGE model on this new dataset and observes that the fairness metric values are lower than those of the original model, as shown in fig. 11.

Final verification of insights through GNNFairViz interface (Other rounds). Sam returns to the interface to continue verifying his insights. He first selects the nodes with a middle number of neighbors in the plot of Number of Neighbors in Computational Graphs, and then nodes in the dense subgraph in Dense Subgraphs plot, successively. All of the selected nodes exhibit negative bias contributions and display a non-homophilic pattern. Through these multiple rounds of analysis, Sam achieves **final verification of his insights**.

Further development after analysis in Jupyter Notebook. Sam suspects that the sampling mechanism of GraphSAGE model might mitigate the effect of graph structure in promoting model fairness. This is because graph structure makes the node embeddings of minority groups contain substantial information of other groups, yet the sampling mechanism reduces the amount of information aggregated from neighborhoods, thereby increasing the impact of the nodes' own attributes. To test this, he trains a GCN model on the same dataset, resulting in lower fairness metric values except for the AP metrics (fig. 11).

C. Expert Interviews

To further evaluate our approach, we engaged in individual interviews with four experts (E4-E7), all of whom are currently engaged in the development or application of GNNs, and none of them is a coauthor of this paper. These experts include a machine learning engineer at a tech company with expertise in visual analytics and GNNs (E4), along with three researchers who possess extensive experience in the applications of GNNs (E5, E6, E7). The interviews followed a semi-structured format. Initially, we provided an overview of our approach's background and objectives without delving into algorithm specifics. Next, we conducted a tutorial on GNNFairViz using the live usage scenario examples from section VII-A and section VII-B. This introductory session lasted approximately 30 minutes. Following that, we asked the experts to explore the two datasets and models in the usage

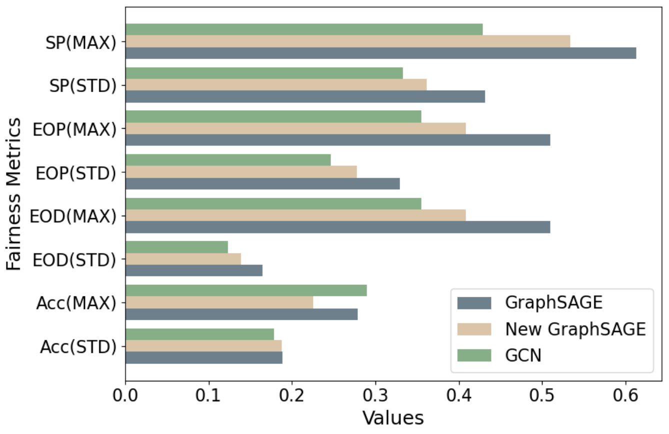

Fig. 11. Comparison of fairness metrics in usage scenario 2, where the user re-trains a GraphSAGE model on the dataset from which the edges and attributes recorded during analysis using GNNFairViz have been removed. Subsequently, the user trains a GCN model after further analysis using GNNFairViz and identifying potential side effects of GraphSAGE's sampling mechanism on model fairness. The results show that both bias mitigation strategies enhance fairness, demonstrating the effectiveness of the fairness analysis.

scenarios using GNNFairViz for 20 minutes, after which we gathered their feedback.

Effectiveness. We received positive feedback on the framework. Experts emphasized the necessity of such a tool for analyzing potential fairness issues in GNN models. They noted that the analysis would be very complex without GNNFairViz. For example, E6 acknowledged the pipeline's functionality for GNN fairness analysis, specifically stating, "*I wouldn't know how to start and approach the analysis when I encountered fairness issues without this tool. Even though I know the dataset and model well, it would be complex to check model fairness from different perspectives and understand why it happens.*" Specifically, experts highlighted the tool's capability in aiding comparisons across various aspects, including inter- and intra-group connectivity (E4, E5, E7), node embedding distributions of different groups (E5), bias in each attribute (E5, E6, E7), and the attribute distribution of selected nodes versus all nodes (E6). All experts acknowledged that the necessary information on node attributes for fairness analysis is well presented, particularly praising the Attribute Overview in the Diagnostic View. Additionally, E4 and E7 commended GNNFairViz for its intuitive display of graph structure information from the perspective of fairness. According to E7, "*this tool accurately grasps the key aspects in GNN fairness: attributes and graph structure, and effectively connects them to model fairness.*" When asked if they would use GNNFairViz to analyze fairness in GNN models when encountering such problems, they expressed enthusiasm for gaining insights that guide bias mitigation.

Tool Design. Overall, the experts agreed that the tool, designed for users in the ML community, can be easily integrated into their workflows. Regarding the visualization aspect, they appreciated that each view clearly displays its intended information, with plot titles effectively describing their functions. E7 noted that the visualization presents the necessary information comprehensively and accurately. However, E4, E5, and E6 initially found that the glyphs in the Attribute Overview become too small when the number of attributes is large, making it difficult to inspect the bias contribution of each attribute at a glance. After demonstrating the zooming and dragging interactions of the plots, they acknowledged that the interface is flexible and scalable to accommodate a large number of attributes through interactions.

Suggestions. The experts also provided several suggestions for enhancing GNNFairViz. E5 discussed the possibility of supporting comparisons between two models: "*After I gained insights into the fairness of the model and trained another model based on the insights, it would be helpful if I could compare their fairness using this tool.*" Further, E4, E6, and E7 all suggested that a guideline or tutorial document for GNNFairViz is necessary. This document should contain an introduction to the interface, illustrations of basic interactions, and usage cases. E7 stated: "*The functions and interactions of this tool are comprehensive and flexible. So there is a certain time cost to learn to use it. However, since all the visualizations and interactions are intuitive, it is going to be easy to understand and use this tool with a document-format tutorial of the interview.*"

VIII. Discussion

In this section, we discuss general insights into GNN fairness as a summary of our observations on the use of GNNFairViz and the associated qualitative principle analysis. Additionally, we summarize the limitations of our approach and propose directions for future research.

General Insights into GNN Fairness. Although the specific insights gained through fairness analysis using GNNFairViz can vary across different scenarios, we have identified two broadly applicable principles. Firstly, in real-world datasets that are highly unbalanced with respect to sensitive groups (e.g., 5% female and 95% male when gender is the sensitive attribute), the graph structure tends to promote model fairness. Theoretically, there are more inter-group edges connecting minority groups to majority groups than there are intra-group connections within each minority group due to the unbalance nature, even though nodes in each group tend to connect to nodes in the same group. Consequently, the information from the majority groups dominates the node embeddings of the minority groups. A detailed example is provided in section VII-A. This observation is likely generalizable to real-world scenarios, as the likelihood of inter-group connections to minority groups being fewer than intra-minority group connections is extremely low, a situation mostly seen in synthetic data. We term this phenomenon the "Overwhelming Effect" where the nodes in minority groups are "overwhelmed" by the influence of the majority nodes. Secondly, once insights into bias in GNN models have been obtained, it is possible to mitigate model bias by selecting suitable GNN architectures, even though they are not designed with fairness consideration. For instance, in scenarios where the condition of the Overwhelming Effect is met, enhancing model fairness can be achieved by choosing an architecture that makes the effect stronger than the current one, as demonstrated in section VII-A and section VII-B. We anticipate that these insights warrant more rigorous further analysis and can serve as a valuable reference in the field of GNN fairness.

Scalability. One limitation of GNNFairViz is its scalability. On the computational side, we have attempted to address this challenge by utilizing GPUs to accelerate calculations, optimizing techniques for sparse matrices in data processing, and implementing sampling techniques for visualization. Thus, users can have smooth experience on graphs with up to 20,000 nodes using this version of GNNFairViz. While this accommodates many popular benchmark datasets, larger graphs still present a challenge. The primary cause of these scalability problems is the quadratic increase in the time and space complexity of graph data. Future improvements could involve handling larger graphs by integrating graph database technologies and leveraging advanced hardware. On the visualization side, scalability concerns mainly arise in three charts: The Node embeddings plot, the Dense Subgraphs plot, and the Attribute Overview plot. In the Node Selection View, clutter is mitigated through the support of random sampling and the rangeset technique. For the Dense Subgraphs plot, users can adjust the density threshold to manage overlap caused by many detected subgraphs. The Attribute Overview balances scalability and the overview functionality with zooming and panning for large datasets. Despite these mitigations, scalability challenges remain. Future improvements could enhance the Dense Subgraphs plot for better handling of a large number of points while preserving interactivity and refine the Attribute Overview by designing enhanced visualizations that maintain the ability to provide a comprehensive overview.

Generalization. GNNFairViz is both model-agnostic, making it suitable for a wide array of scenarios, and flexible by enabling users to specify any combination of sensitive attributes. However, it is currently exemplified for analyzing node classification tasks and does not support other tasks like link prediction. Additionally, while the fairness metrics are derived from our literature review on GNN fairness, users might have specific requirements that necessitate further customization of these metrics. Future work will focus on extending our approach to accommodate these needs.

Learning curve. Although GNNFairViz does not feature complex visualization charts, it can be challenging for domain experts, particularly those without prior knowledge in visual analytics, as discussed in section VII-C. The high level of coordination between the visualization charts may be not familiar to GNN model developers, who are more accustomed to static visualizations common in the ML community. It takes time for these users to become familiar with the interactive features. It can be helpful to provide a tutorial document for users. However, to address the challenge from the source involves making GNNFairViz easier for domain experts to get started while ensuring it remains powerful enough for in-depth analysis, highlighting the need for continuous improvement and innovation in visual analysis tools.

IX. Conclusion

In this work, we presented a visual analytics framework for GNN fairness. This general and flexible framework analyzes model bias from the perspectives of attribute bias and structural bias. We also developed GNNFairViz, a visual analysis tool that is easy to integrate into the working environment and workflow of target users. The evaluation of our tool demonstrated its usability and effectiveness in analyzing GNN fairness and providing valuable insights for bias mitigation.

Acknowledgments

We thank anonymous reviewers for their constructive comments. This work is supported by Natural Science Foundation of China (NSFC No.62472099 and No.62202105).

References

[1] Q. Wang, Z. Xu, Z. Chen, Y. Wang, S. Liu, and H. Qu, “Visual analysis of discrimination in machine learning,” *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 1470–1480, 2021.
[2] A. Zayed, P. Parthasarathi, G. Mordido, H. Palangi, S. Shabanian, and S. Chandar, “Deep learning on a healthy data diet: Finding important examples for fairness,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 14 593–14 601.
[3] C. Xu, Y. Zhang, H. Chen, L. Dong, and W. Wang, “A fairness-aware graph contrastive learning recommender framework for social tagging systems,” *Inf. Sci.*, vol. 640, 2023.
[4] N. Rajput and K. Singh, “Temporal graph learning for financial world: Algorithms, scalability, explainability and fairness,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2022, pp. 4818–4819.
[5] Y. Dong, J. Ma, S. Wang, C. Chen, and J. Li, “Fairness in graph mining: A survey,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10 583–10 602, 2023.
[6] M. Abdelrazek, E. Purificato, L. Boratto, and E. W. De Luca, “Fairup: A framework for fairness analysis of graph neural network-based user profiling models,” in *Proc. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, 2023, pp. 3165–3169.
[7] Y. Wang, Y. Zhao, Y. Dong, H. Chen, J. Li, and T. Derr, “Improving fairness in graph neural networks via mitigating sensitive attribute leakage,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2022, pp. 1938–1948.
[8] W. Song, Y. Dong, N. Liu, and J. Li, “Guide: Group equality informed individual fairness in graph neural networks,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2022, pp. 1625–1634.
[9] Y. Dong, S. Wang, Y. Wang, T. Derr, and J. Li, “On structural explanation of bias in graph neural networks,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2022, pp. 316–326.
[10] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini, “Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning,” *IEEE Trans. Artif. Intell.*, vol. 3, no. 3, pp. 344–354, 2022.
[11] N. Navarin, L. Oneto, and M. Donini, “Learning deep fair graph neural networks,” in *Proc. Eur. Symp. Artif. Neural Netw. Comput. Intell. Mach. Learn.*, 2020, pp. 31–36.
[12] O. D. Kose and Y. Shen, “Fairness-aware graph attention networks,” in *Conf. Rec. Asilomar Conf. Signals Syst. Comput.*, vol. 2022-October, 2022, pp. 843–846.
[13] Y. Dong, S. Wang, J. Ma, N. Liu, and J. Li, “Interpreting unfairness in graph neural networks via training node attribution,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 7441–7449.
[14] Z. Wang, Q. Zeng, W. Lin, M. Jiang, and K. C. Tan, “Generating diagnostic and actionable explanations for fair graph neural networks,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, 2024, pp. 21 690–21 698.
[15] T. Xie, Y. Ma, J. Kang, H. Tong, and R. Maciejewski, “Fairrankvis: A visual analytics framework for exploring algorithmic fairness in graph mining models,” *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 368–377, 2022.
[16] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. Int. Conf. Learn. Represent.*, 2016.
[17] P. Veličković, A. Casanova, P. Liò, G. Cucurull, A. Romero, and Y. Bengio, “Graph attention networks,” in *Proc. Int. Conf. Learn. Represent.*, 2018.
[18] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, p. 4 – 24, 2021.
[19] J. Kang, J. He, R. MacIejewski, and H. Tong, “Inform: Individual fairness on graph mining,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2020, pp. 379–389.

[20] E. Purificato and E. W. De Luca, "What are we missing in algorithmic fairness? discussing open challenges for fairness analysis in user profiling with graph neural networks," in *Commun. Comput. Inf. Sci.*, vol. 1840 CCIS, 2023, pp. 169–175.

[21] E. Purificato, L. Boratto, and E. W. De Luca, "Do Graph Neural Networks Build Fair User Models? Assessing Disparate Impact and Mistreatment in Behavioural User Profiling," in *Proc. ACM Int. Conf. Inf. Knowl. Manag.*, 2022, pp. 4399–4403.

[22] J. Wu, S. M. Abrar, N. Awasthi, and V. Frías-Martínez, "Auditing the fairness of place-based crime prediction models implemented with deep learning approaches," *Comput. Environ. Urban Syst.*, vol. 102, 2023.

[23] N. Zhou, Z. Zhang, V. N. Nair, H. Singhal, and J. Chen, "Bias, fairness and accountability with artificial intelligence and machine learning algorithms," *Int. Stat. Rev.*, vol. 90, no. 3, pp. 468–480, 2022.

[24] Y. Dong, N. Liu, B. Jalaian, and J. Li, "Edits: Modeling and mitigating data bias for graph neural networks," in *Proc. ACM Web Conf.*, 2022, pp. 1259–1269.

[25] P. Li, Y. Wang, H. Zhao, P. Hong, and H. Liu, "On dyadic fairness: Exploring and mitigating bias in graph connections," in *Proc. Int. Conf. Learn. Represent.*, 2021.

[26] E. Dai and S. Wang, "Learning fair graph neural networks with limited and private sensitive attribute information," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7103–7117, 2023.

[27] Z. Zeng, R. Islam, K. N. Keya, J. Foulds, Y. Song, and S. Pan, "Fair representation learning for heterogeneous information networks," in *Proc. Int. AAAI Conf. Web Soc. Media*, vol. 15, 2021, pp. 877–887.

[28] W. G. La Cava, "Optimizing fairness tradeoffs in machine learning with multiobjective meta-models," in *Proc. Genet. Evol. Comput. Conf.*, 2023, pp. 511–519.

[29] M. Hort, R. Moussa, and F. Sarro, "Multi-objective search for gender-fair and semantically correct word embeddings," in *Proc. Genet. Evol. Comput. Conf.*, 2023, pp. 23–24.

[30] S. Hod, "Responsibly: Toolkit for auditing and mitigating bias and fairness of machine learning systems," 2018–.

[31] R. K. E. Bellamy, A. Mojsilovic, S. Nagar, K. N. Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, Y. Zhang, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, and S. Mehta, "Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias," *IBM J. Res. Dev.*, vol. 63, no. 4-5, 2019.

[32] H. Baniecki, W. Kretowicz, P. Piatyszek, J. Wisniewski, and P. Biecek, "dalex: Responsible machine learning with interactive explainability and fairness in python," *J. Mach. Learn. Res.*, vol. 22, pp. 1–7, 2021.

[33] J. Wexler, M. Pushkarna, T. Bolukbasi, M. Wattenberg, F. Viegas, and J. Wilson, "The what-if tool: Interactive probing of machine learning models," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 56–65, 2020.

[34] A. A. Cabrera, W. Epperson, F. Hohman, M. Kahng, J. Morgenstern, and D. H. Chau, "Fairvis: Visual analytics for discovering intersectional bias in machine learning," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2019, pp. 46–56.

[35] J. Liu, H. Chen, J. Shen, and K. R. Choo, "Faircompass: Operationalising fairness in machine learning," *IEEE Trans. Artif. Intell.*, pp. 1–10, 2023.

[36] Y. Ahn and Y. R. Lin, "Fairsight: Visual analytics for fairness in decision making," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 1086–1095, 2020.

[37] A. Rissaki, B. Scarone, D. Liu, A. Pandey, B. Klein, T. Eliassi-Rad, and M. A. Borkin, "Biascope: Visual unfairness diagnosis for graph embeddings," in *Proc. IEEE Vis. Data Sci.*, 2022, pp. 27–36.

[38] Y. Wang, "Fair graph representation learning with imbalanced and biased data," in *Proc. ACM Int. Conf. Web Search Data Min.*, 2022, pp. 1557–1558.

[39] L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson, "Counterfactual reasoning and learning systems: The example of computational advertising." *J. Mach. Learn. Res.*, vol. 14, no. 11, 2013.

[40] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Stat.*, vol. 33, no. 3, pp. 1065–1076, 1962.

[41] J. Lin, "Divergence measures based on the shannon entropy," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 145–151, 1991.

[42] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Stat.*, vol. 22, no. 1, pp. 79–86, 1951.

[43] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 7, pp. 1216–1230, 2012.

[44] J. Yuan, S. Xiang, J. Xia, L. Yu, and S. Liu, "Evaluation of sampling methods for scatterplots," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 1720–1730, 2021.

[45] J. T. Sohns, M. Schmitt, F. Jirasek, H. Hasse, and H. Leitte, "Attribute-based explanation of non-linear embeddings of high-dimensional data," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 540–550, 2022.

Xinwu Ye received a B.S. from Shanghai University of Finance and Economics. He is currently a master's student at the School of Data Science, Fudan University. His primary research interests are visualization, human-computer interaction, artificial intelligence, and computational biology.

Jielin Feng is currently a Ph.D. at the School of Data Science, Fudan University. Her primary research focuses on visualization and visual analytics, with a particular emphasis on AI-driven generative visualization.

Erasmo Purificato is a Scientific Project Officer at the European Commission's JRC, focusing on trustworthy algorithmic systems. He holds a Ph.D. from OVGU, specializing in fairness in GNNs, and has worked as a Research Assistant at OVGU and the Leibniz Institute, and as an ML Engineer at Blue Reply. He has co-organized workshops, served as a guest editor, and contributed to conferences like RecSys and UMAP. More at erasmopurif.com.

Ludovico Boratto is an Associate Professor of CS at the University of Cagliari. His research interests focus on recommender systems and their impact. He got his Ph.D. at the University of Cagliari, where he was a research assistant until May 2016. From May 2016 to April 2021, he joined Eurecat. He was a visiting researcher at Yahoo! Research. He is a member of ACM and IEEE.

Michael Kamp leads the Trustworthy Machine Learning group at the Institute for AI in Medicine, University Medicine Essen. Previously, he was a postdoc at CISPA Helmholtz Center and Monash University. Holding a Ph.D. in CS from Bonn University, he has over 40 publications, including 15+ in top venues. Michael is on the editorial board of the Springer Machine Learning Journal, an ELLIS member, and has served in various organizing chairs. More at https://michaelkamp.org/.

Zengfeng Huang is currently a Professor in SDS, FDU. He obtained his doctoral degrees from HKUST respectively. His research mainly focuses on Big Data algorithms, ML, and theoretical CS. He has published over fifty papers in top-tier international journals and conferences. His research has earned multiple awards, including recognition at ICML, WAIC, and ACM PODS.

Siming Chen is an Associate Professor at Fudan University's School of Data Science. Previously a Research Scientist at Fraunhofer IAIS, he earned his Ph.D. from Peking University. His research focuses on visualization and visual analytics, emphasizing Human-AI Collaboration. With over 100 publications, including 40+ in top venues like IEEE VIS and ACM CHI, he has received 10+ best paper and honorable mention awards. More at http://simingchen.me.