Counterfactual Fairness for Graph Neural Networks with Limited and Privacy Protected Sensitive Attributes

Xuemin Wang XUEMINWANGBETTER@163.COM
Guangxi Key Laboratory of Trusted Software

Lei Wang 13028689617@163.COM
Guangxi Key Laboratory of Trusted Software

Tianlong Gu * GUTIANLONG@JNU.EDU.CN
Engineering Research Center of Trustworthy AI (Ministry of Education)

Xuguang Bao BAOXUGUANG@GUET.EDU.CN
Guangxi Key Laboratory of Trusted Software

Editors: Vu Nguyen and Hsuan-Tien Lin

Abstract

Graph Neural Networks (GNNs) have shown outstanding performance in learning graph representations, which increases their application in high-risk areas. However, GNNs may inherit biases from the graph data and make unfair predictions towards the protected sub-groups. To eliminate bias, a natural idea is to achieve counterfactual fairness from a causal perspective. Concretely, counterfactual fairness requires sufficient sensitive attributes as guidance, which is infeasible in the real world. The reason is that users with various privacy preferences may selectively publish their sensitive attributes and only limited sensitive attributes can be collected. Besides, the users who publish sensitive attributes still face privacy risks. In this paper, we first consider the situation in which the sensitive attributes are limited and propose a framework called PCFGR (Partially observed sensitive Attributes in Counterfactual Fair Graph Representation Learning) to learn fair graph representation from limited sensitive attributes. The framework trains a sensitive attribute estimator, which is applied to provide sufficient and accurate sensitive attributes. With these sensitive attributes, it can generate counterfactuals and eliminate the bias efficiently. Secondly, we aim to protect the privacy of the sensitive attributes and further propose PCFGR\D. Specifically, PCFGR\D first perturbs the sensitive attributes using Local Differential Privacy (LDP). Then it employs forward correction loss to train an accurate sensitive attributes estimator. We conduct extensive experiments and the experiment results show that it outperforms other alternatives in balancing utility and fairness.

Keywords: fair graph neural network; counterfactual fairness; graph representation learning; privacy protection

1. Introduce

Graph Neural Networks (GNNs) learn low-dimensional representations of nodes, which shows excellent performance on various downstream tasks such as node classification, link prediction, and graph classification Wu et al. (2020). The outstanding performance of GNN in graph representation learning increases their application in high-risk domains, such as predicting protein-protein interactions Gainza et al. (2020), drug reuse Morselli Gysi et al.

* Corresponding author.

(2021), crime prediction Jin et al. (2020), and news and product recommendations Ying et al. (2018). However, GNNs can inherit biases from graph data, and the message aggregation mechanism of the model can exacerbate these biases. For example, in graphs like social networks, nodes with similar sensitive attributes (such as race or age) are likely to be connected Agarwal et al. (2021). Since GNNs use message passing to aggregate representations of neighboring nodes, nodes with similar sensitive attributes may share similar representations. In downstream tasks, nodes with similar representations may receive similar predictions. Hence, the predictions are highly correlated with sensitive attributes, which results in bias towards the protected groups. Therefore, there is a need to develop GNNs that can learn fair graph representations.

Currently, numerous fairness metrics have been proposed for GNNs. Among them, statistical parity and equality of opportunity are widely applied. For these metrics, several fair graph neural networks have been introduced. For instance, FairGNN Dai and Wang (2021) employs a sensitive attribute estimator to impute missing sensitive attributes and subsequently utilizes adversarial training to eliminate sensitive attribute information from node representations. Moreover, existing research has begun to delve into counterfactual fairness. NIFTY Agarwal et al. (2021) generates counterfactuals by perturbing the sensitive attributes and edges of nodes, enabling the learning of counterfactually fair and robust node representations. GEAR Ma et al. (2022) automatically generates counterfactuals corresponding to the perturbation of sensitive attributes for each node and its neighboring nodes, thereby eliminating the causal relationship between node representations and sensitive attributes. Although existing counterfactual graph representation learning techniques have been relatively comprehensive, they have two limitations: 1) (Insufficient Sensitive Attributes). They all assume they can have full access to the sensitive attributes. However, they ignore the situation that the sensitives are limited. This is due to the various user's privacy preferences. Since existing methods need to generate counterfactuals with the guidance of sensitive attributes, the efficiency of fairness promotion is decreased without sufficient counterfactuals. 2)(Sensitive Attributes Leaking).Although some users agree to publish their sensitive attributes, privacy risks also exist. Hence, it is necessary to protect the privacy of sensitive attributes and promote counterfactual fairness on these private sensitive attributes.

Firstly, we only consider situations where sensitive attributes are limited and propose a framework named PCFGR. This framework includes a sensitive attribute estimator, fairness module, and utility modules. The sensitive attribute estimator accurately predicts the true sensitive attribute values of nodes with missing sensitive attributes using the feature vectors of neighboring nodes with observed sensitive attributes and non-sensitive attributes. The fairness module first computes important node pairs using the PageRank algorithm Yao et al. (2019), generates subgraphs using a subgraph generator, obtains augmented counterfactual subgraphs, learns graph representations using a Siamese neural network Zhao et al. (2020), and finally updates the loss function through the utility module to achieve the final counterfactual graph representation learning.

Secondly, we consider the privacy of partially observed sensitive attributes. We propose a new framework PCFGR\D. Here, we leverage the random response to flip the sensitive attribute values with equal probability using ε-local differential privacy protection. To acquire accurate sensitive attributes, we train the classifier using forward correction loss Shui

et al. (2022). Training estimator with forward loss not only integrates well with differential privacy protection technique but also, without disclosing personal information, directly measures the difference between the model's predictions and the actual values, clarifying the model training objective. The main contributions of this paper are as follows: 1) We proposed a new framework (PCFGR) for learning counterfactual graph representations of partially observed sensitive attributes; 2) The sensitive attribute estimator contributes to learning counterfactually fair graph representations; 3) The new framework (PCFGR\D) can learn counterfactually fair representations while protecting the privacy of sensitive attributes; 4) We conducted comparative experiments to further demonstrate the effectiveness of the proposed framework.

2. Preliminaries

Graph neural network. Current GNNs are neighborhood aggregation approaches Ying et al. (2018), that update the representations of the nodes with the representations of the neighborhood nodes. The representations after k layers' aggregation would capture the structural information of the k-hop network neighborhoods Wang et al. (2018). The updating process of the k-th layer in GNN could be formulated as $a_v^{(k)} = \text{AGGREGATE}^{(k-1)}\left(\left\{h_u^{(k-1)} : u \in \mathcal{N}(v)\right\}\right)$, $h_v^{(k)} = \text{COMBINE}^{(k)}\left(h_v^{(k-1)}, a^{(k)}\right)$ where $h_v^{(k)}$is the representation vector of the node $v \in V$ at k-th layer and $N(v)$ is a set of neighborhoods of v.

Counterfactual fairness. Counterfactual fairness Kusner et al. (2017) is derived from causal structural models and is used to assess and ensure fairness in machine learning algorithms. Causal models consist of causal graphs and structural equations. A causal graph is a directed acyclic graph (DAG) where each node represents a variable Wu et al. (2019), and each directed edge represents a causal relationship. Structural equations describe these causal relationships between variables. For variables Y and S, the counterfactual value asks the question: "What would Y be if S were set to s' ?" and is denoted as $Y_{S\leftarrow s'}$ Chiappa (2019). Based on the given causal model, counterfactual fairness is achieved if the following conditions hold for any feature $X = x$ and sensitive attribute $S = s$: when $X = x$ and $S = s$, the predicted value $\hat{Y} = f(X)$ satisfies the counterfactual condition: $P\left(\hat{Y}_{S\leftarrow s} = y \mid X = x, S = s\right) = P\left(\hat{Y}_{S\leftarrow s'} = y \mid X = x, S = s\right)$

3. PCFGR

3.1. Overview

To learn counterfactual fair graph representations with limited sensitive attributes named PCFGR. It consists of three key components: 1) Sensitive attribute estimator completes missing sensitive attribute values; 2) Counterfactual fair graph representation learning ensures the counterfactual graph representation learning for nodes; and 3) A utility module preserves the competitive accuracy of the module.

3.2. Sensitive Attribute Estimator

Bias in the graph data may be introduced into the graph representation and is amplified by the message-passing mechanism. Subsequently, the bias can lead to discriminatory pre-

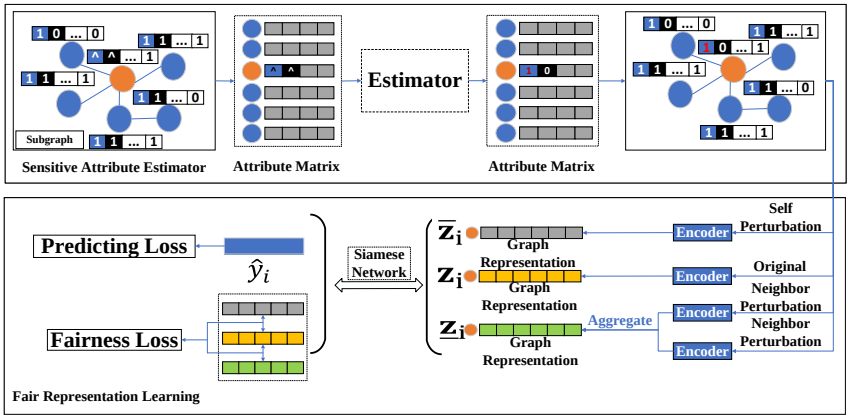

Figure 1: Schematic diagram of the framework PCFGR, where $\wedge$ indicates that there is no record for the node here.

dictions in downstream tasks. To eliminate bias, the sensitive attributes are essential as guidance to generate counterfactuals. Moreover, in real-world practice, users with various privacy preferences publish their sensitive attributes selectively, which results in limited sensitive attributes. Hence, this leads to insufficient sensitive attributes and makes it difficult to generate sufficient counterfactuals, which results in a poor improvement in fairness promotion. Since the homomorphism of the graph, nodes with similar sensitive attributes are likely to be connected Dai and Wang (2021); Hu et al. (2022). This makes it possible to predict accurate sensitive attributes of nodes in $\mathcal{V}-\mathcal{V}_S$ using both G and V_S. Therefore, we design a sensitive attribute estimator using a graph neural network $f_E : G \longrightarrow S$ to predict the sensitive attributes of nodes of private users. In our model, the sensitive attributes s_v are treated as binary variables, where $s_v \in \{0, 1\}$. The objective loss function of f_E is:

$$\min_{\theta_E} \mathcal{L}_E = -\frac{1}{|\mathcal{V}_S|} \sum_{v \in \mathcal{V}_S.} \left[s_v \log \hat{s}_v + (1 - s_v) \log \left(1 - \hat{s}_v\right) \right] \tag{1}$$

where $\hat{s}_v$ is the predicted sensitive attribute of node $v \in \mathcal{V}_S$ obtained through f_E, and θ_E is the parameter set of f_E. We can predict the sensitive attribute $\hat{s}_i$ for node $v_i \in \mathcal{V} - \mathcal{V}_S$ using f_E. Each generated sensitive attribute $\hat{s}_i$ is added into the existing set of the sensitive attributes S to gain $\hat{S}$.

3.3. Counterfactual graph representation learning

3.3.1. Subgraph generation

The true causal model of graph data is often challenging to fully capture, particularly for large-scale graphs Jiao et al. (2020). Therefore, to reduce time complexity, we generate a subgraph for each node that includes its top-k most important neighbors. This is because a node's representation is mainly influenced by its immediate neighbors Rahman et al. (2019). Specifically, we use a subgraph generator $Sub(\cdot)$ to extract the contextual information of

the central node v_i from graph G, generating a contextual subgraph $G^{(i)}$, which includes the node features $X^{(i)}$ and adjacency matrix $A^{(i)}$ of node v_i Hamilton et al. (2017) and all its top-k hops within the neighborhood. Based on these contextual subgraphs, we can obtain more sufficient information relative to the central node in terms of graph structure, thereby achieving high-quality graph representation learning and subsequent counterfactual augmentation.

To further determine which neighboring nodes are more important for the central node v_i, and to facilitate the $TOP(\cdot)$ operation Zhan et al. (2021), we calculate the importance scores for each pair of nodes: $R = \alpha(I-(1-\alpha)\overline{A})$. Where R is the importance score matrix, $R_{i,j}$ represents the importance of node j to node i, α is a parameter within the range $[0,1]$, I is the identity matrix, and $\overline{A} = A \times D^{-1}$ represents the column-normalized adjacency matrix, where D is the diagonal matrix with $D_{i,i} = \sum_j A_{i,j}$. In this way, we can select the top-k important nodes $V^{(i)}$ for each central node v_i based on the importance score matrix R, and then obtain the contextual subgraph $G^{(i)}$ of the central node v_i as follows:

$$G^{(i)} = \left\{V^{(i)}, \mathcal{E}^{(i)}, X^{(i)}\right\} = \left\{A^{(i)}, X^{(i)}\right\}, V^{(i)} = TOP(R_{i,:}, k), A^{(i)} = A_{v^{(i)},v^{(i)}}, X^{(i)} = X_{v^{(i)},:} \tag{2}$$

where the symbol (:) denotes all indices. The subgraph generated by the above process is defined as $G^{(i)} = Sub(i, G, k)$. The generated subgraphs are then input into the encoder to learn representations of the central nodes. This process has a complexity of $O(|V| \cdot k)$, where $|V|$ is the number of nodes, and k is the size of the neighborhood. This approach significantly reduces the computational burden compared to methods that operate on the full graph.

3.3.2. Counterfactual amplification

With the generated subgraphs $G^{(i)}$, we generate the counterfactuals using two types of perturbations: self-perturbation and neighbor-perturbation. **Self-perturbation.** In the subgraph $G^{(i)}$, the sensitive attribute value of the central node s_i is flipped. The generated subgraph serves as the corresponding counterfactual. A subgraph is represented as $\bar{G}^{(i)} = \left\{G^{(i)}_{S_i \leftarrow 1-s_i}\right\}$. **Neighbor-perturbation.** Similarly, in the subgraph $G^{(i)}$, the sensitive attribute values of any node except the central node are randomly perturbed, i.e., the nodes in the set $V^{(i)}_{\neg i}$. With such perturbations, a set of counterfactuals is generated as $\underline{G}^{(i)} = \left\{G^{(i)}_{S^{(i)}_{\neg i} \leftarrow SMP\left(S^{(i)}_{\neg i}\right)}\right\}$ where $SMP(\cdot)$ randomly selects specific values of sensitive attributes from the value space $\{0,1\}^{|V^{(i)}|-1}$. The operation $SMP(\cdot)$ is conducted C times.

3.3.3. Fair representation learning

For fairness in graph counterfactuals, the goal is to learn the same representations for each central node from these three kinds of subgraphs including $G^{(i)}$, $\bar{G}^{(i)}$, $\underline{G}^{(i)}$. Since siamese neural network van Knippenberg et al. (2021) is designed to compare the different inputs by learning their similarity in a shared latent space, we adopt a siamese neural network as the encoder $\phi(\cdot)$ to generate three representations z_i, $\bar{z}_i$, $\underline{z}_i$ for each node v_i. respectively, which is formulated as follows:

$$z_i = \left(\phi\big(\mathrm{X}^{(i)}, \mathrm{A}^{(i)}\big)\right)_i \tag{3}$$

$$\overline{z}_i = \mathrm{AGG}\left(\left\{\left(\phi\left(\mathrm{X}^{(i)}_{S_i \leftarrow 1-s_i}, \mathrm{A}^{(i)}\right)\right)_i\right\}\right) \tag{4}$$

$$\underline{z}_i = \mathrm{AGG}\left(\left\{\left(\phi\left(\mathrm{X}^{(i)}_{S_{\neg i} \leftarrow \mathrm{SMP}\left(S^{(i)}_{\neg i}\right)}, \mathrm{A}^{(i)}_{S_{\neg i} \leftarrow \mathrm{SMP}\left(S^{(i)}_{\neg i}\right)}\right)\right)_i\right\}\right) \tag{5}$$

where $\phi(\cdot)\colon \mathbb{R}^{k\times d} \times \mathbb{R}^{k\times k} \longrightarrow \mathbb{R}^{k\times d'}$ takes each subgraph as input and embeds each node on the input subgraph into a latent representation. Each central node i represented as z_i learned from the original data, and $Z = \{z_i\}^n_{(i=1)}$ is used for downstream tasks. For sampled counterfactual subgraphs $\bar{G}^{(i)}$ and $\underline{G}^{(i)}$ an aggregator (e.g., mean aggregator) $AGG(\cdot)$ is used to aggregate the representations of each central node, resulting in final representations $\bar{z}_i$ and $\underline{z}_i$. Then, the distance between the learned representations of central nodes from the original and counterfactual subgraphs is minimized. The loss function for graph counterfactual fairness is given by:

$$\mathcal{L}_f = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\Big((1-\lambda_s)d(z_i, \overline{z}_i) + \lambda_s d\big(z_i, \underline{z}_i\big)\Big) \tag{6}$$

where $d(\cdot)$ is a distance metric, such as cosine distance. $\lambda_s \in \{0, 1\}$ is a hyperparameter controlling the weight of neighbor disturbance.

3.4. Utility module

The utility module is designed to preserve competitive accuracy while considering counterfactual fairness. Taking node classification tasks as an example, the PCFGR framework can naturally extend to other types of tasks on graph data, such as link prediction. The labels are represented as $Y = \{y_1, \ldots, y_n\}$. The prediction loss can be expressed as:

$$\mathcal{L}_p = \frac{1}{n}\sum_{i\in[n]} l(f(z_i), y_i) \tag{7}$$

where $l(\cdot)$ is a loss function measuring prediction error (e.g., cross-entropy), $f(\cdot)$ is used to predict downstream tasks using the representations, i.e., $\hat{y}_i = f(z_i)$. Finally, the overall loss function for fair representation learning is:

$$\mathcal{L} = \mathcal{L}_p + \beta\mathcal{L}_f + \mu\|\theta\|^2 + \mathcal{L}_E \tag{8}$$

where θ is the set of model parameters, β and μ are hyperparameters controlling the weights of fairness constraints on graph counterfactuals.

3.5. Overall training algorithm for PCFGR

The algorithm for Counterfactual fair graph representation learning with limited sensitive attributes is presented in Algorithm 1. First, the graph data G and learning parameters λ and μ are fed into the model, and f_E is updated using θ_E and Equation (4) (Line 1-3). Prediction completion is performed to obtain the complete graph G, and the sensitive

attribute estimator f_E is optimized using the updated loss function to obtain the completed complete graph G (Lines 4). Subsequently, subgraphs $G^{(i)}$ are obtained using the subgraph generation component (Line 5), and self-perturbation and neighbor perturbation are applied to generate subgraphs $\bar{G}^{(i)}$ and $\underline{G}^{(i)}$ respectively (Line 6). Then, the corresponding original (counterfactual) subgraphs are obtained respectively (Lines 7). Finally, the three types of obtained subgraphs are input into the Siamese neural network to learn counterfactually fair graph representations (Lines 8).

Algorithm 1 Counterfactual Fair Graph Representation Learning with Limited Sensitive Attributes

1: **Input:** $G = (V, E, X)$, β, μ, θ_E
2: **Output:** Counterfactual fair graph representation H of graph G after completing sensitive attributes
3: Initialize f_E using θ_E and Equation (1);
4: Obtain the graph G with completed sensitive attributes estimated by f_E;
5: Obtain subgraphs $G_i = \mathrm{Sub}_i(G, k)$ using Equations (2);
6: Obtain subgraphs $\bar{G}^{(i)}$ and $\underline{G}^{(i)}$ using Equations self-perturbation and neighbor-perturbation;
7: Obtain z_i, $\bar{z}_i$ and $\underline{z}_i$ using Equations (3-5) respectively;
8: Minimize the loss function L using Equation (8) to obtain the training parameters for the optimal counterfactual fair graph representation H.

4. PCFGR\D

In this section, we further consider the privacy of the sensitive attributes and how to generate the counterfactuals from privacy-preserving sensitive attributes. A novel framework named PCFGR\D is further proposed, which adopts LDP to perturb the sensitive attributes and leverages the forward correction loss to train a sensitive attribute estimator. The specific training algorithm is shown in Figure 2, with detailed information as follows :

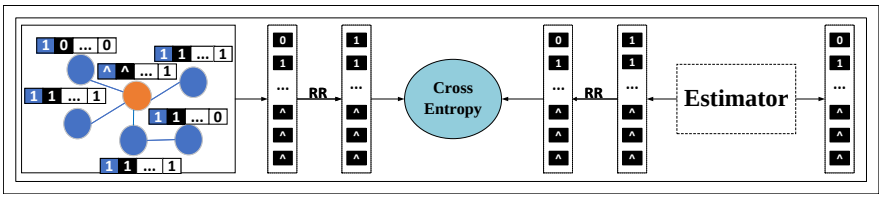

Figure 2: Schematic diagram of the sensitive attribute estimator training algorithm for PCFGR\D, where RR denotes the random response mechanism.

We first apply a random response mechanism to perturb all sensitive attribute values, achieving the goal of privacy protection. Specifically, on the binary sensitive attributes, we flip them with a probability of $\rho = \frac{1}{\exp(\epsilon)+1}$, where ϵ denotes the privacy budget. The larger

ϵ indicates the lowever privacy protection. If we directly use this graph data for counterfactual fair graph representation learning, the sensitive attribute estimator f_E deployed in Section 3.2 learns the noise label and it is difficult to predict accurate sensitive attributes. To provide sufficient and accurate sensitive attributes, it is necessary to improve the estimator to learn with noisy sensitive attributes. Since the sensitive attributes are randomly flipped using LDP, the obtained sensitive attributes only depend on the original values S_i, i.e.$P\left(\hat{s}_i \mid s_i, X\right) = P\left(\hat{s}_i \mid s_i\right)$. To mitigate the influence of noise in the sensitive attribute labels, the loss function is formulated as follows:

$$\mathcal{L}\left(P\left(\hat{s}_i \mid X\right)\right) = -\log P\left(\hat{s}_i \mid X\right) = -\log \sum_j P\left(\hat{s}_i \mid s_i = j\right) P\left(s_i = j \mid X\right) \tag{9}$$

From equation (17), it can be observed that if the objective of the sensitive attribute estimator f_E is to estimate $\hat{s}_i$, then prediction correction based on $P\left(\hat{s}_i \mid s_i\right)$ needs to be performed before the cross-entropy loss. $P\left(\hat{s}_i \mid s_i\right)$ can be represented using a noise transition matrix T , where $T_{ij} = P(\hat{s}_i = i \mid s_i = j)$. Since ϵ is known, meaning the flipping probability of sensitive attribute $s \in \{0, 1\}$ is known, the noise transition matrix T can be expressed as:

$$\mathrm{T} = \begin{bmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{bmatrix} \tag{10}$$

By incorporating the correction from equation (17) and the noise transition matrix T, the objective function for optimizing the sensitive attribute estimator f_E under the constraints of limited and differentially private sensitive attributes can be formally expressed as:

$$\min_{\theta_E} \mathcal{L}_E = \frac{1}{|S|} \sum_{s_i \in S} \left[-\log \sum_j T_{s_i j} f_E\left(\hat{s}_i = j \mid X\right) \right] \tag{11}$$

where θ_E represents the learnable parameters of the sensitive attribute estimator f_E, and $f_E(\hat{S}_i = j \mid X)$ denotes the probability predicted by f_E that instance X belongs to sensitive group i.

4.1. Overall training algorithm

The algorithm for Counterfactual fair graph representation learning with limited sensitive attributes and differential privacy protection is presented in Algorithm 2.

First, differential privacy protection is uniformly applied to the sensitive attributes of the graph data, simulating graph data that has been actually released after privacy protection (line 3). Then, the graph data along with the learning parameters are fed into the model, utilizing f_E to predict completion and obtain the complete graph G, and optimize the sensitive attribute estimator f_E using the updated loss function (17) to obtain the completed complete graph G (lines 4-5). Subsequently, utilizing the subgraph generation component, subgraphs $G^{(i)}$ are obtained (line 6), and self-perturbation and neighbor perturbation are applied to generate subgraphs $\bar{G}^{(i)}$ and $\underline{G}^{(i)}$ respectively (lines 7). Finally, the obtained three types of subgraphs are input into the Siamese neural network to learn counterfactual fair graph representations (lines 8-9).

Algorithm 2 Counterfactual Fair Graph Representation Learning with Limited Sensitive Attributes and Differential Privacy Protection

Input: $G = (V, E, X)$, λ, μ, θ_E
Output: Counterfactual fair graph representation H of graph G after completing sensitive attributes and differential privacy protection
Perturb the sensitive attributes using LDP.
Initialize f_E using θ_E and Equation (11);
Obtain the graph G with completed sensitive attributes estimated by f_E;
Obtain subgraphs $G_i = \mathrm{Sub}_i(G, k)$ using Equations (2);
Obtain subgraphs $\bar{G}^{(i)}$ and $\underline{G}^{(i)}$ using Equations self-perturbation and neighbor-perturbation;
Obtain z_i, $\bar{z}_i$ and $\underline{z}_i$ using Equations (3-5) respectively;
Minimize the loss function L using Equation (8) to obtain the training parameters for the optimal counterfactual fair graph representation H.

5. Experiment

In this section, we conducted extensive experiments to verify the effectiveness of the proposed method. Experiments were carried out on different datasets to answer the following two questions: **RQ1:** "How can counterfactually fair graph representation learning be achieved with partially observed sensitive attributes ?" **RQ2:** "How can counterfactually fair graph representation learning be achieved with perturbed sensitive attributes ?"

5.1. Dataset

To demonstrate the efficiency of the proposed solution, we conduct experiments with two datasets: 1)German Dataset: The German credit graph dataset consists of 1000 nodes, where each node represents a customer of a bank in Germany. These customers are connected based on the similarity of their credit accounts. The task involves using the gender of the customers as a sensitive attribute and categorizing them into good and bad credit categories. 2)Bail Dataset: This graph contains data on defendants released on bail in state courts in the United States. In this graph, each node represents a defendant, and each edge between a pair of nodes represents the similarity of their criminal records and demographic data. The race of the defendant is used as the sensitive attribute. The task is to decide whether to grant bail to the defendants (if released, they are not likely to commit violent crimes) or not. 3)Credit Dataset: This graph contains information on people's default payment behaviors. Each node represents a person, and each edge between a pair of nodes represents the similarity of their consumption and payment patterns. Age is used as the sensitive attribute. The task is to predict whether their default payment method is a credit card.

5.2. Experimental setup

Metrics. The proposed framework is evaluated from two aspects: predictive performance and fairness. To assess predictive performance, widely used node classification metrics are

employed, including Accuracy, F1-Score, and AUROC. To measure the fairness of representation, two commonly used metrics in statistical fairness, (Δ_{SP}) and (Δ_{EO}), as well as (δ_{CF}) evaluating counterfactual fairness of the graph These metrics are provided in previous work Ma et al. (2022).

To assess the counterfactual fairness of our proposed model, we manipulate the ratio of sensitive attribute subgroups in each dataset by randomly altering node attributes. Specifically, we randomly select $0\%, 50\%$, or 100% of the nodes, setting their sensitive attribute values to 1 and the rest to 0 . These perturbations generate counterfactual data based on causal models across the entire graph, resulting in different proportions of sensitive attribute subgroups. This process implicitly controls the distribution of sensitive attributes in each node's neighborhood. We estimate δ_{CF} by calculating the average rate at which predicted labels flip due to these perturbations. Additionally, we compute $R^2\left(\hat{Y}_i, \hat{S}_i\right)$ to quantify how well $\hat{Y}_i$ can be linearly predicted from a summary of the sensitive attributes in node i 's neighborhood. Here, $\hat{S}_i$ is derived from the sensitive attribute values of all single-hop neighbors and the average value for node v_i itself. The R^2 measure thus reflects the statistical dependency between $\hat{Y}_i$ and $\hat{S}_i$.

Implement details. Each dataset is randomly split into training (60%)/validation (20%)/testing (20%) sets. Unless otherwise specified, hyperparameters are set as follows: $\lambda = 0.6$, $C = 2$, $\lambda_s = 0.4$, $\beta = 10$, $\mu = 1e-5$, $k = 20$, $B = 4$. Learning rate l is set to 0.001, epochs are 1000, node representation dimension is 1024, and batch size is 100. The experimental results are the average of ten repeated executions. Adam optimizer is used, and the method is implemented using PyTorch.

Baseline. The proposed framework is compared with node representation learning methods GCN without fairness constraints and state-of-the-art counterfactual fairness graph representation learning method GEAR Ma et al. (2022), respectively, and the latest group fairness graph neural network FairKGD Zhu et al. (2023). Besides, we also include a privacy-preserving method LPGNNSajadmanesh and Gatica-Perez (2021).

5.3. Effectiveness of PCFGR

To answer **RQ** 1, we conducted comparative experiments between our proposed framework PCFGR, and the current optimal counterfactual fair graph representation learning framework based on graph data, GEAR. Their respective experimental results in terms of predictive accuracy and fairness performance are shown in Table 1. In summary, the following observations can be made: 1) The proposed PCFGR framework demonstrates prediction performance comparable to state-of-the-art node representation learning methods and outperforms GEAR in terms of prediction; 2) The PCFGR framework outperforms GEAR on both fairness indicators δ_{CF} and R^2. These two fairness metrics explicitly consider the causal/statistical relationship between neighboring sensitive attributes and model predictions, thus validating the effectiveness of PCFGR in mitigating neighbor bias. Additionally, PCFGR performs well on other fairness metrics and aspects.

There are differences in the experimental results of PCFGR on the German dataset at 10% and 90% sensitive attribute proportions. The main reasons are: a) the German dataset is relatively small, so when only 10% of sensitive attributes are observable, there is insufficient data for the sensitive attribute estimator, leading to differences in experimental

Table 1: Comparative experimental results under different sensitive attribute ratios in different datasets

Dataset	Ratios	Method	Accuracy	F1-score	AUC	Equality	Parity	δ_{CF}	R^2
Bail	**10%**	GEAR	$80.30_{\pm 4.81}$	$74.45_{\pm 4.76}$	$86.54_{\pm 3.64}$	$2.70_{\pm 1.62}$	$2.85_{\pm 1.64}$	$1.95_{\pm 0.40}$	$3.41_{\pm 0.90}$
		PCFGR	$80.92_{\pm 3.38}$	$75.27_{\pm 3.06}$	$87.33_{\pm 1.99}$	$3.19_{\pm 2.14}$	$2.71_{\pm 2.34}$	$1.84_{\pm 0.33}$	$3.24_{\pm 0.48}$
	30%	GEAR	$80.85_{\pm 5.20}$	$74.91_{\pm 5.24}$	$87.13_{\pm 4.01}$	$2.49_{\pm 2.02}$	$4.35_{\pm 2.35}$	$1.97_{\pm 0.45}$	$4.34_{\pm 1.35}$
		PCFGR	$82.13_{\pm 6.27}$	$76.59_{\pm 7.08}$	$87.28_{\pm 5.09}$	$3.14_{\pm 1.22}$	$2.62_{\pm 2.51}$	$1.39_{\pm 0.51}$	$3.69_{\pm 1.00}$
	50%	GEAR	$79.85_{\pm 6.52}$	$74.23_{\pm 7.25}$	$85.98_{\pm 4.92}$	$3.17_{\pm 2.77}$	$2.67_{\pm 2.75}$	$2.10_{\pm 0.63}$	$3.59_{\pm 1.20}$
		PCFGR	$81.30_{\pm 4.45}$	$75.90_{\pm 4.25}$	$87.42_{\pm 2.29}$	$2.77_{\pm 0.60}$	$2.64_{\pm 2.16}$	$1.12_{\pm 0.16}$	$3.44_{\pm 1.16}$
	70%	GEAR	$81.18_{\pm 5.02}$	$75.47_{\pm 5.66}$	$87.10_{\pm 4.20}$	$2.43_{\pm 3.19}$	$2.99_{\pm 2.31}$	$1.51_{\pm 0.36}$	$3.77_{\pm 1.04}$
		PCFGR	$82.45_{\pm 3.75}$	$76.55_{\pm 3.50}$	$87.99_{\pm 1.89}$	$3.60_{\pm 1.88}$	$3.30_{\pm 3.19}$	$1.33_{\pm 0.38}$	$3.61_{\pm 1.24}$
	90%	GEAR	$82.13_{\pm 4.54}$	$76.46_{\pm 4.87}$	$88.02_{\pm 3.38}$	$2.86_{\pm 2.24}$	$3.48_{\pm 2.54}$	$1.37_{\pm 0.28}$	$3.96_{\pm 1.37}$
		PCFGR	$83.08_{\pm 4.76}$	$77.78_{\pm 5.02}$	$89.54_{\pm 3.26}$	$3.51_{\pm 2.57}$	$3.46_{\pm 2.81}$	$1.20_{\pm 0.21}$	$3.81_{\pm 1.06}$
German	**10%**	GEAR	$66.27_{\pm 4.25}$	$76.98_{\pm 4.76}$	$62.66_{\pm 4.08}$	$11.46_{\pm 6.79}$	$8.55_{\pm 5.98}$	$2.31_{\pm 0.83}$	$1.77_{\pm 1.38}$
		PCFGR	$66.13_{\pm 4.06}$	$75.93_{\pm 4.72}$	$63.66_{\pm 3.23}$	$8.54_{\pm 2.59}$	$8.30_{\pm 0.70}$	$2.49_{\pm 1.31}$	$1.42_{\pm 0.34}$
	30%	GEAR	$65.47_{\pm 5.63}$	$75.01_{\pm 6.47}$	$63.84_{\pm 2.58}$	$13.02_{\pm 10.52}$	$12.69_{\pm 10.59}$	$4.98_{\pm 1.75}$	$3.41_{\pm 3.23}$
		PCFGR	$68.68_{\pm 4.71}$	$78.99_{\pm 4.96}$	$63.45_{\pm 4.17}$	$9.99_{\pm 4.63}$	$10.14_{\pm 6.01}$	$3.24_{\pm 0.72}$	$2.29_{\pm 0.89}$
	50%	GEAR	$60.67_{\pm 2.49}$	$68.33_{\pm 2.55}$	$63.68_{\pm 3.22}$	$15.82_{\pm 7.85}$	$19.98_{\pm 11.05}$	$2.27_{\pm 1.32}$	$5.56_{\pm 5.00}$
		PCFGR	$61.47_{\pm 2.62}$	$68.97_{\pm 1.77}$	$63.74_{\pm 2.77}$	$15.64_{\pm 8.26}$	$20.50_{\pm 4.72}$	$2.13_{\pm 1.73}$	$5.09_{\pm 2.46}$
	70%	GEAR	$61.07_{\pm 1.05}$	$68.71_{\pm 1.44}$	$64.90_{\pm 2.41}$	$16.94_{\pm 10.26}$	$13.22_{\pm 9.66}$	$2.31_{\pm 1.72}$	$6.26_{\pm 4.11}$
		PCFGR	$61.07_{\pm 2.07}$	$68.79_{\pm 1.68}$	$64.25_{\pm 3.35}$	$14.64_{\pm 8.61}$	$19.95_{\pm 9.05}$	$1.64_{\pm 0.28}$	$5.58_{\pm 4.19}$
	90%	GEAR	$62.27_{\pm 1.86}$	$69.59_{\pm 1.74}$	$65.79_{\pm 2.52}$	$17.96_{\pm 9.83}$	$24.63_{\pm 9.82}$	$2.62_{\pm 1.09}$	$7.24_{\pm 5.32}$
		PCFGR	$61.20_{\pm 1.82}$	$68.69_{\pm 0.91}$	$64.77_{\pm 3.63}$	$18.35_{\pm 9.78}$	$23.86_{\pm 9.96}$	$2.44_{\pm 0.13}$	$7.27_{\pm 5.72}$
Credit	**10%**	GEAR	$81.32_{\pm 0.41}$	$75.55_{\pm 3.46}$	$87.23_{\pm 2.67}$	$3.28_{\pm 1.92}$	$2.85_{\pm 1.84}$	$1.59_{\pm 0.32}$	$3.85_{\pm 1.78}$
		PCFGR	$81.62_{\pm 2.38}$	$76.37_{\pm 3.26}$	$87.45_{\pm 2.54}$	$3.44_{\pm 2.34}$	$2.91_{\pm 2.14}$	$1.44_{\pm 0.26}$	$3.74_{\pm 1.38}$
	30%	GEAR	$82.35_{\pm 3.20}$	$75.93_{\pm 2.44}$	$87.34_{\pm 2.88}$	$3.49_{\pm 1.89}$	$3.35_{\pm 2.24}$	$1.34_{\pm 0.38}$	$3.86_{\pm 1.25}$
		PCFGR	$83.43_{\pm 3.17}$	$76.59_{\pm 3.81}$	$87.98_{\pm 3.19}$	$3.57_{\pm 1.59}$	$3.13_{\pm 2.73}$	$1.16_{\pm 0.39}$	$3.59_{\pm 1.07}$
	50%	GEAR	$82.95_{\pm 3.52}$	$77.13_{\pm 2.35}$	$87.56_{\pm 2.78}$	$3.57_{\pm 2.47}$	$3.26_{\pm 2.75}$	$1.10_{\pm 0.63}$	$3.57_{\pm 1.19}$
		PCFGR	$83.50_{\pm 2.45}$	$77.90_{\pm 2.15}$	$88.42_{\pm 1.19}$	$3.68_{\pm 1.98}$	$3.19_{\pm 2.35}$	$0.92_{\pm 0.21}$	$3.44_{\pm 1.16}$
	70%	GEAR	$83.58_{\pm 3.02}$	$77.93_{\pm 2.65}$	$88.37_{\pm 2.50}$	$3.73_{\pm 2.24}$	$3.39_{\pm 2.13}$	$0.98_{\pm 0.26}$	$3.46_{\pm 1.18}$
		PCFGR	$84.45_{\pm 3.75}$	$78.95_{\pm 3.25}$	$89.69_{\pm 1.47}$	$3.85_{\pm 1.68}$	$3.42_{\pm 2.41}$	$0.89_{\pm 0.34}$	$3.35_{\pm 1.15}$
	90%	GEAR	$84.31_{\pm 3.45}$	$78.32_{\pm 2.47}$	$91.67_{\pm 2.58}$	$3.86_{\pm 2.44}$	$3.73_{\pm 2.24}$	$0.87_{\pm 0.28}$	$3.34_{\pm 1.09}$
		PCFGR	$85.18_{\pm 2.66}$	$80.78_{\pm 3.12}$	$92.34_{\pm 2.16}$	$3.98_{\pm 2.34}$	$3.61_{\pm 2.31}$	$0.71_{\pm 0.31}$	$3.11_{\pm 0.95}$

results; b) Similarly, when 90% of sensitive attributes are observable, there is less missing sensitive attributes, leading to overfitting of the sensitive attribute estimator, resulting in differences in experimental results.

5.4. Effectiveness of PCFGR\D

To address **RQ** 2, we conducted comparative experiments involving GEAR, the node representation learning method GCN without fairness constraints, the partially observed sensitive attribute counterfactual fair graph representation learning method PCFGR, and the counterfactual fair graph representation learning algorithm PCFGR\D with limited sensitive attributes under differential privacy protection, and the latest research on group fairness on graphs, FairGKD. The experimental results at a 70% sensitive attribute ratio and with a fixed privacy budget are shown in Table 2, where the best results are highlighted in bold and underlined.

The results in Table 2 show the most outstanding performance compared to other models, achieving the best results on the δ_{CF} metric. Although it slightly underperforms compared to PCFGR in terms of the R^2 metric, this also illustrates the bias introduced by directly using sensitive attributes protected by differential privacy for graph representation learning. The differential privacy adds noise to the original graph data, which affects the

Table 2: Results of comparative experiment at 70% sensitive attribute ratio.

	GCN	FairGKD	GEAR	LPGNN	PCFGR	PCFGR\D
Accuracy	$57.73_{\pm 7.68}$	$69.21_{\pm 4.11}$	$52.00_{\pm 16.23}$	$60.53_{\pm 1.32}$	$51.33_{\pm 6.90}$	$\mathbf{69.87_{\pm 3.09}}$
F1-score	$63.35_{\pm 13.00}$	$70.53_{\pm 3.46}$	$50.32_{\pm 31.49}$	$68.18_{\pm 0.48}$	$49.95_{\pm 13.33}$	$\mathbf{80.21_{\pm 3.29}}$
auc_roc	$63.41_{\pm 3.93}$	$65.88_{\pm 2.69}$	$64.43_{\pm 2.24}$	$64.46_{\pm 3.10}$	$66.46_{\pm 2.84}$	$\mathbf{67.17_{\pm 0.84}}$
Equality	$27.32_{\pm 8.17}$	$13.55_{\pm 3.56}$	$20.92_{\pm 22.59}$	$28.00_{\pm 9.91}$	$16.50_{\pm 8.99}$	$\mathbf{11.25_{\pm 10.05}}$
parity	$30.72_{\pm 8.82}$	$21.33_{\pm 2.79}$	$22.53_{\pm 23.48}$	$31.89_{\pm 9.87}$	$19.70_{\pm 9.70}$	$\mathbf{19.04_{\pm 10.46}}$
$\delta_{\mathbf{CF}}$	$64.00_{\pm 14.99}$	Null	$79.87_{\pm 22.53}$	$41.73_{\pm 2.51}$	$30.13_{\pm 10.87}$	$\mathbf{30.00_{\pm 29.26}}$
$\mathbf{R^2}$	$10.75_{\pm 5.95}$	Null	$11.01_{\pm 13.51}$	$11.00_{\pm 6.98}$	$\mathbf{5.17_{\pm 3.27}}$	$8.42_{\pm 4.72}$

causal relationship between sensitive attributes and prediction outcomes, resulting in a decrease in R^2. However, this does not necessarily imply that there is an absence of a causal relationship between them.

5.5. Ablation Experiment

To demonstrate the importance of each part of the proposed framework, we specifically conducted ablation studies for verification. In the ablation experiments, we removed two major components from *PCFGR\D* and compared them with node representation learning methods, GCN without fairness constraints, under the same parameter settings. We named the versions without the sensitive attribute estimator and the denoised sensitive attribute estimator as *PCFGR w/o E* and *PCFGR w/o D*, respectively. The other experiment setting is the same as that of section 5.4. The experimental results are shown in Figure 3.

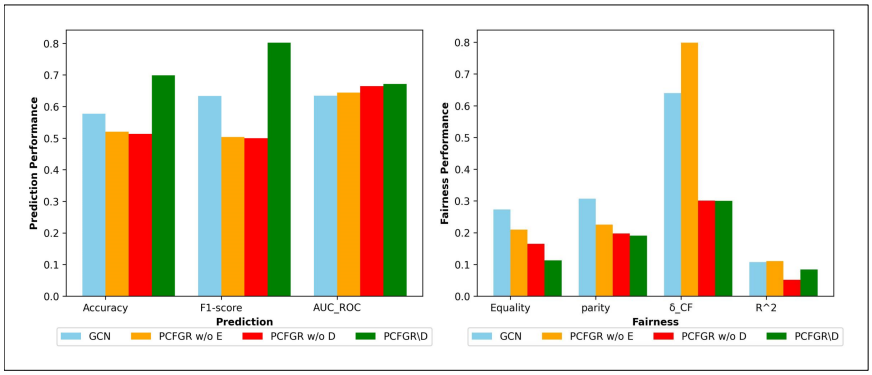

Figure 3: Results of ablation experiment at 70% sensitive attribute ratio.

From Figure 3, we find that *PCFGR\D* outperforms other models on most metrics. This is because *PCFGR\D* restores the sensitive attributes that have been processed by differential privacy protection, which leads to an increased correlation between its prediction results and the sensitive attributes, i.e., an increase in R^2. Additionally, the removal of the sensitive attribute estimator or the denoised sensitive attribute estimator impacts the learning effect of *PCFGR\D*.

5.6. Parameter sensitivity analysis

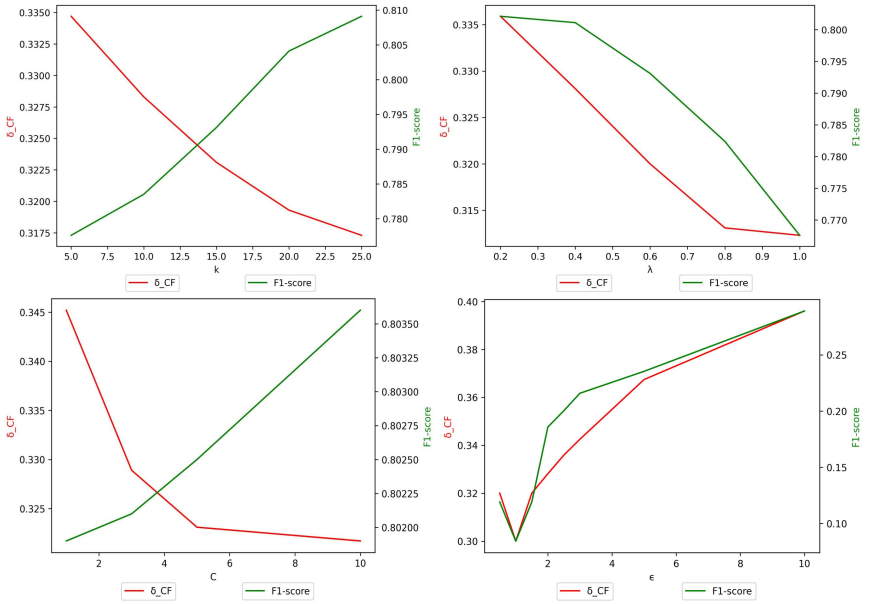

Figure 4: Parameter sensitivity analysis on German dataset.

To evaluate the model performance and stability changes of PCFGR under different parameters, We conducted a parameter sensitivity analysis experiment on the following hyperparameters, such as k, $lambda$, C, ϵ. The experimental results of different parameter settings on the German dataset are shown in Figure 4.

As the privacy budget (ϵ) increases, the strength of privacy protection gradually decreases, allowing the estimator to better estimate sensitive attributes to guide bias removal. When $0 \leq \epsilon < 1$, both the δ_{CF} and R^2 metrics gradually decrease, indicating an enhancement in counterfactual fairness and a weakening causal relationship between prediction results and sensitive attributes. When $\epsilon = 1$, the experimental results achieve optimal performance, as local differential privacy protection algorithms generally perform best at $\epsilon = 1$. When $\epsilon > 1$, although increasing the recovery of sensitive attributes leads to an improvement in counterfactual fairness, the algorithm considers scenarios under high privacy strength, resulting in an overall decrease in fairness despite the fairness improvement operations.

Unlike ϵ, as λ increases, the general trend for both δ_{CF} and R^2 is a gradual decrease. What differs between them is that the decrease in δ_{CF} is initially rapid and then gradually slows down; the trend for R^2 is in contrast to that of δ_{CF}, with R^2 decreasing slowly at first and then speeding up. As other parameters (C, k) increase, the trends for δ_{CF} and R^2 are roughly similar, with δ_{CF} gradually becoming smaller and R^2 gradually increasing.

6. Related Work

FairAGG Zhu et al. (2024) promotes group fairness by limiting the influence of sensitive attributes during the aggregation. SRGNN Zhang et al. (2024) mitigates biases by rebalancing the graph structure before learning. However, they focus on group fairness rather than counterfactual fairness. NIFTY Agarwal et al. (2021) generates counterfactuals by perturbing nodes' sensitive attributes and edges. It ensures the learned representations from these counterfactuals closely resemble those from the original graph, thereby learning fair and robust node representations. GEAR Ma et al. (2022) attains graph counterfactual fairness using counterfactual data augmentation. It generates counterfactuals by perturbing the sensitive attributes of each node and its neighbors. It eliminates the causal influence of sensitive attributes by minimizing the difference between node representations learned from the original graph and those from the counterfactuals. CAP Guo et al. (2023) selects counterfactuals directly from the training data. By utilizing these authentic counterfactuals, it learns fair node representations for node classification tasks.

However, these counterfactually fair graph neural network methods assume that sensitive attributes are fully observable, which is not the case in reality. Besides, they overlook the privacy of sensitive attributes.

7. Conclusion

In this paper, we propose a framework named PCFGR which can promote counterfactual fairness while the sensitive attributes are limited. Specifically, in PCFGR, the sensitive attribute estimator is first used to predict the missing sensitive attributes based on the observed partial sensitive attributes and non-sensitive attributes. Subsequently, counterfactual fair graph representation learning for partially observed sensitive attributes is achieved. To protect the privacy of sensitive attributes, we further propose a framework called PCFGR\D. PCFGR\D employs differential privacy protection to perturb the sensitive attributes and train the estimator using forward correction loss to predict accurate sensitive attributes. With sufficient and accurate sensitive attributes, we learn the representation that aligns with counterfactual fairness.

Acknowledgments

This paper is supported by the National Natural Science Foundation of China (Grant No. U22A2099, 62336003) and the Guangxi Natural Science Foundation under Grant 2021GXNSFBA196054.

References

Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pages 2114–2124. PMLR, 2021.

Silvia Chiappa. Path-specific counterfactual fairness. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7801–7808, 2019.

Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 680–688, 2021.

Pablo Gainza, Freyr Sverrisson, Frederico Monti, Emanuele Rodola, D Boscaini, Michael M Bronstein, and Bruno E Correia. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

Zhimeng Guo, Jialiang Li, Teng Xiao, Yao Ma, and Suhang Wang. Towards fair graph neural networks via graph counterfactual. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 669–678. ACM, 2023.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Hui Hu, Lu Cheng, Jayden Parker Vap, and Mike Borowczak. Learning privacy-preserving graph convolutional network with partially observed sensitive attributes. In *Proceedings of the ACM Web Conference*, pages 3552–3561. ACM, 2022.

Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. Subgraph contrast for scalable self-supervised graph representation learning. In *2020 IEEE international conference on data mining (ICDM)*, pages 222–231. IEEE, 2020.

Guangyin Jin, Qi Wang, Cunchao Zhu, Yanghe Feng, Jincai Huang, and Jiangping Zhou. Addressing crime situation forecasting task with temporal graph convolutional neural network approach. In *2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, pages 474–478. IEEE, 2020.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *Advances in neural information processing systems*, 30, 2017.

Jing Ma, Ruocheng Guo, Mengting Wan, Longqi Yang, Aidong Zhang, and Jundong Li. Learning fair node representations with graph counterfactual fairness. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 695–703, 2022.

Deisy Morselli Gysi, Ítalo Do Valle, Marinka Zitnik, Asher Ameli, Xiao Gan, Onur Varol, Susan Dina Ghiassian, JJ Patten, Robert A Davey, Joseph Loscalzo, et al. Network medicine framework for identifying drug-repurposing opportunities for covid-19. *Proceedings of the National Academy of Sciences*, 118(19):e2025581118, 2021.

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

Sina Sajadmanesh and Daniel Gatica-Perez. Locally private graph neural networks. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2130–2145. ACM, 2021.

Changjian Shui, Boyu Wang, and Christian Gagné. On the benefits of representation regularization in invariance based domain generalization. *Machine Learning*, 111(3):895–915, 2022.

Marijn van Knippenberg, Mike Holenderski, and Vlado Menkovski. Time-constrained multi-agent path finding in non-lattice graphs with deep reinforcement learning. In *Asian Conference on Machine Learning*, pages 1317–1332. PMLR, 2021.

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 349–357, 2018.

Yongkai Wu, Lu Zhang, and Xintao Wu. Counterfactual fairness: Unidentification, bound and algorithm. In *Proceedings of the twenty-eighth international joint conference on Artificial Intelligence*, 2019.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377, 2019.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

Huixin Zhan, Kun Zhang, Chenyi Hu, and Victor Sheng. k2-gnn: Multiple users' comments integration with probabilistic k-hop knowledge graph neural networks. In *Asian conference on machine learning*, pages 1477–1492. PMLR, 2021.

Guixian Zhang, Debo Cheng, Guan Yuan, and Shichao Zhang. Learning fair representations via rebalancing graph structure. *Inf. Process. Manag.*, 61(1):103570, 2024. doi: 10.1016/J.IPM.2023.103570. URL https://doi.org/10.1016/j.ipm.2023.103570.

Tianxiang Zhao, Xianfeng Tang, Xiang Zhang, and Suhang Wang. Semi-supervised graph-to-graph translation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1863–1872, 2020.

Yuchang Zhu, Jintang Li, Liang Chen, and Zibin Zheng. The devil is in the data: Learning fair graph neural networks via partial knowledge distillation. *arXiv preprint arXiv:2311.17373*, 2023.

Yuchang Zhu, Jintang Li, Liang Chen, and Zibin Zheng. Fairagg: Toward fair graph neural networks via fair aggregation. *IEEE Transactions on Computational Social Systems*, pages 1–12, 2024. doi: 10.1109/TCSS.2024.3385539.