Learning Fair Node Representations with Graph Counterfactual Fairness

Jing Ma[1], Ruocheng Guo[2], Mengting Wan[3], Longqi Yang[3], Aidong Zhang[1], Jundong Li[1*]
[1]University of Virginia, Charlottesville, VA, USA 22904
[2]City University of Hong Kong, Hong Kong SAR, China
[3]Microsoft, Redmond, WA, USA 98052
{jm3mr, aidong, jundong}@virginia.edu, ruocheng.guo@cityu.edu.hk, {mengting.wan, Longqi.Yang}@microsoft.com

ABSTRACT

Fair machine learning aims to mitigate the biases of model predictions against certain subpopulations regarding sensitive attributes such as race and gender. Among the many existing fairness notions, counterfactual fairness measures the model fairness from a causal perspective by comparing the predictions of each individual from the original data and the counterfactuals. In counterfactuals, the sensitive attribute values of this individual had been modified. Recently, a few works extend counterfactual fairness to graph data, but most of them neglect the following facts that can lead to biases: 1) the sensitive attributes of each node's neighbors may causally affect the prediction w.r.t. this node; 2) the sensitive attributes may causally affect other features and the graph structure. To tackle these issues, in this paper, we propose a novel fairness notion – graph counterfactual fairness, which considers the biases led by the above facts. To learn node representations towards graph counterfactual fairness, we propose a novel framework based on counterfactual data augmentation. In this framework, we generate counterfactuals corresponding to perturbations on each node's and their neighbors' sensitive attributes. Then we enforce fairness by minimizing the discrepancy between the representations learned from the original graph and the counterfactuals for each node. Experiments on both synthetic and real-world graphs show that our framework outperforms the state-of-the-art baselines in graph counterfactual fairness, and also achieves comparable prediction performance.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Applied computing** → *Law, social and behavioral sciences.*

KEYWORDS

Counterfactual fairness; graph; fairness; node representation

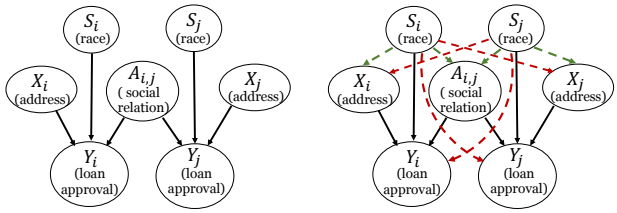
ACM Reference Format:

Jing Ma, Ruocheng Guo, Mengting Wan, Longqi Yang, Aidong Zhang, Jundong Li. 2022. Learning Fair Node Representations with Graph Counterfactual Fairness. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498391

*Corresponding Author.

WSDM '22, February 21–25, 2022, Tempe, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498391



(a) M' **(b)** M

Figure 1: Causal models generally used in existing works (M') and in this work (M). We use S_i, X_i, Y_i to denote the sensitive attribute, features, and label of any node i, and $A_{i,j} \in \{0, 1\}$ denotes the edge between node pair (i, j). Each arrow denotes a causal relation. The dashed lines denote the causal relations that the existing works do not consider.

1 INTRODUCTION

Representation learning on graphs aims to map nodes into a latent embedding space. These node representations are often used to power downstream predictive tasks, and have become the new state-of-the-art in multiple real-world applications [24, 37, 43, 44]. However, these node representation learning approaches may overlook potential biases buried in the graph data, thus introducing algorithmic biases against subpopulations defined by certain sensitive attributes such as race, gender, and age. Consequently this may raise ethical and societal concerns, especially in high-stake decision-making scenarios such as ranking of job applicants [29] and credit scoring [34]. For example, it would become a serious ethical issue if a bank's decision on the loan application was affected by the applicant's and their close contacts' race information.

To tackle the above problem, several approaches were proposed to assess and address the fairness of node representation learning on graphs. The majority of these methods aim to learn node representations which can elicit *statistically* fair predictions across the population [1, 6, 12, 27]. In addition, the concept of *counterfactual fairness* has been extended to graph-structured data recently [1, 2]. Different from the previous statistical notions, counterfactual fairness extends Pearl's causal structural models [32] and aims to encourage the predictions made from different versions of the same individual (a.k.a. *counterfactuals*) to be equal. For example, the prediction for one's loan application being approved should be the same regardless this applicant being Black or White.

This paper falls under the umbrella of counterfactual fairness but focuses on addressing two critical limitations of existing studies [1, 2] of counterfactual fairness on graphs: 1) biases induced by **one's neighboring nodes** and 2) biases induced by **the causal relations from the sensitive attributes to other features as well as the graph structure**. We follow the previous loan application example to explain these limitations in details: i) As illustrated in Fig. 1(a), existing studies mostly focus on mitigating the causal influence from the sensitive attribute (race information S_i) of the i-th applicant on the prediction of the label (loan approval decision Y_i), but neglect the fact that the race information of the applicant's social contacts (S_j) can also causally affect the fairness of the prediction (as labeled using red dashed edges in Fig. 1(b)). ii) On the other hand, existing methods may implicitly assume the sensitive attribute (S_i) has no causal effect on other variables such as node features (X_i) and the graph structure ($A_{i,j}$) so that they can safely simplify the counterfactual data generation mechanism as by just flipping the sensitive attribute values. However, we question the applicability of this assumption since such causal effect is ubiquitous in real-world scenarios. For example, one's race can causally influence their social relations as well as the residential neighborhood they live in (as labeled using green dashed edges in Fig. 1(b)).[1]

We argue that biases in model predictions can be induced by the aforementioned pathways. In this paper, we propose a more comprehensive fairness notion on graphs – *graph counterfactual fairness*, which considers the potential biases regarding the sensitive attributes of each node and its neighboring nodes, as well as the biases led by the causal effect from sensitive attributes on other variables. With this notion, learning node representations towards graph counterfactual fairness is still challenging. It is because the causal relations among variables (as showed in Fig. 1(b)) are often required to obtain the counterfactuals, but these causal relations are often unknown in practice. Manually constructing the entire causal model requires extensive domain knowledge and human efforts, especially for large-scale graph data. To address the above challenge, we propose a novel framework to learn **G**raph count**E**rfactually f**A**ir node **R**epresentations (**GEAR**). GEAR aims to learn node representations towards graph counterfactual fairness, and maintain high performance for downstream tasks such as node classification. GEAR includes the following modules: 1) **Subgraph generation.** To reduce the costs of modeling the causal relations on large graphs, we first develop an algorithm to automatically infer the importance scores among nodes. For each individual node, we then prune the range of the causal model to an ego-centric subgraph which contains only the node itself and its most influential neighboring nodes. 2) **Counterfactual data augmentation.** For each node, we leverage the graph auto-encoder technique [23] and fairness constraints to generate two types of counterfactuals as data augmentation: 1) self-perturbation: the counterfactuals where each node's own sensitive attribute value had been modified; 2) neighbor-perturbation: the counterfactuals where the sensitive attribute values of neighbors had been modified. 3) **Node representation learning.** To learn node representations towards graph counterfactual fairness, we leverage a Siamese network [7] to minimize the discrepancy between the representations learned from the original subgraph and those learned from counterfactuals. The main contributions of this work can be summarized as follows:

- **Problem.** We propose a new fairness notion — graph counterfactual fairness, which considers the potential biases brought by different causal pathways from sensitive attributes to the graph model predictions.
- **Method.** We propose a novel framework GEAR to learn node representations towards graph counterfactual fairness. Specifically, for each node, we minimize the discrepancy between the representations learned from the original data and the augmented counterfactuals with different sensitive attribute values.
- **Experiments.** We conduct extensive experiments on both synthetic and real-world graphs. The results show that the proposed method outperforms existing baselines in multiple fairness notions, and achieves comparable prediction performance in downstream tasks.

[1]There may also exist causal relations between non-sensitive features and the graph structure, although we do not show them in Fig.1 for simplicity of illustration.

Table 1: Notation.

Notation	Definition
$\mathcal{G}, \mathcal{V}, \mathcal{E}$	the original graph, the set of vertices/edges
$\mathbf{X}, \mathbf{x}_i$	features of all nodes/the i-th node
$\mathbf{A}$	adjacency matrix
n	the number of nodes
$\mathbf{S}, s_i$	sensitive attribute values of all nodes/the i-th node
$\mathbf{Z}, \mathbf{z}_i$	representations of all nodes/the i-th node
$\Phi(\cdot), \phi(\cdot)$	encoder/subgraph encoder
$\texttt{AGG}(\cdot)$	aggregator
$f(\cdot)$	downstream classifier/predictor
$(U)_{V \leftarrow v}$	counterfactual of variable U when V had been set to v
$\neg i$	the indices which are not i
d, d'	dimension of features/representations
$(\cdot)^{(i)}$	information of the subgraph with central node i
$\texttt{SUB}(\cdot)$	subgraph generation operator
$\texttt{SMP}(\cdot)$	sampling operation of sensitive attribute
$\widetilde{\mathbf{S}}$	the summary of neighboring sensitive attribute values
$\overline{\mathcal{G}}^{(i)}, \underline{\mathcal{G}}^{(i)}$	the set of counterfactual subgraphs of $\mathcal{G}^{(i)}$ under self-perturbation/neighbor-perturbation
C	the sampling number in neighbor-perturbation

2 PROBLEM DEFINITION

Notations. Given a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ denotes the node features ($n = |\mathcal{V}|$), and $\mathbf{x}_i \in \mathbb{R}^d$ represents the features of node i. $\mathbf{A} \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix of the graph $\mathcal{G}$, where $\mathbf{A}_{i,j} = 1$ if edge $i \rightarrow j$ exists, otherwise $\mathbf{A}_{i,j} = 0$. Without loss of generalization, we assume $\mathcal{G}$ is undirected and unweighted, but this work can be naturally extended to directed or weighted settings. Each node i has a sensitive attribute $s_i \in \{0, 1\}$ (we assume one single, binary sensitive attribute for simplicity, but our model can also be easily extended to multivariate or continuous sensitive attributes). $\mathbf{S} = \{s_i\}_{i=1}^n$, and s_i is included in $\mathbf{x}_i$. We denote the non-sensitive features as $\mathbf{X}^{\neg s} = \{\mathbf{x}_1^{\neg s}, ..., \mathbf{x}_n^{\neg s}\}$, where $\mathbf{x}_i^{\neg s} = \mathbf{x}_i \backslash s_i$.

Traditional node representation learning methods train an encoder $\Phi(\cdot) : \mathbb{R}^{n \times d} \times \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times d'}$ to map each node to a latent representation. The learned representations for the n nodes are

denoted by $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^n$, where $\mathbf{z}_i = (\Phi(\mathbf{X}, \mathbf{A}))_i$, $\mathbf{z}_i \in \mathbb{R}^{d'}$ for any node i, and d' is the dimensionality of node representations. These representations can be used in various downstream tasks like node classification [4], link prediction [28], and graph classification [46]. $f(\cdot)$ denotes the downstream classifier/predictor. In the node classification task, let y_i denote the true label of the node i, $f(\cdot)$ takes the representation $\mathbf{z}_i$ as input, and outputs the predicted label $\widehat{y}_i$.

Counterfactual fairness. Counterfactual fairness [25] is a fairness notion based on Pearl's structural causal model [32]. A *causal model* consists of a *causal graph* and *structural equations*. A causal graph is a directed acyclic graph (DAG), where each node represents a variable, and each directed edge represents a causal relationship. Structural equations describe these causal relations among variables. For variables Y, S, the value of the *counterfactual* "what would Y have been if S had been set to s?" is denoted by $Y_{S\leftarrow s}$. Based on a given causal model, a predictor $\widehat{Y} = f(X)$ is *counterfactually fair* [25] if under any features $X = x$ and sensitive attribute $S = s$,

$$P(\widehat{Y}_{S\leftarrow s} = y | X = x, S = s) = P(\widehat{Y}_{S\leftarrow s'} = y | X = x, S = s), \quad (1)$$

for all y and $s' \neq s$. Here $\widehat{Y}_{S\leftarrow s} = f(X_{S\leftarrow s}, s)$ denotes the prediction made on the counterfactual when S had been set to s. Intuitively, it aims to minimize the difference between predictions made on each individual and its counterfactuals with different sensitive attribute values. Ideally, the counterfactuals should be generated based on the ground truth causal model. Different from the statistical fairness notions such as equality of opportunity (EO) [19, 47] and demographic parity (DP) [48], counterfactual fairness aims to eliminate the biases led by the causal effect from the sensitive attribute on the observed variables used for model training. However, most existing works of counterfactual fairness focus on i.i.d. data.

Existing notion of counterfactual fairness on graph. Recent works [1, 2] have extended counterfactual fairness to graphs. Given a graph $X = \mathbf{X}, A = \mathbf{A}$, these works consider that an encoder $\Phi(\cdot)$ satisfies counterfactual fairness if for any node i:

$$(\Phi(X_{S_i=0}, A))_i = (\Phi(X_{S_i=1}, A))_i, \quad (2)$$

where $X_{S_i=0}$ and $X_{S_i=1}$ denote the node features after setting S_i as 0 and 1, respectively, while everything else does not change[2]. This notion considers fairness as minimizing the discrepancy between the representations of each node with different values of its sensitive attribute (while everything else is fixed). This notion has the following limitations: 1) it does not consider the potential biases led by the causal effect from the sensitive attribute of other nodes in the graph on the prediction of each node; 2) it implicitly assumes that the sensitive attribute has no causal effect on other features or the graph structure. In a nutshell, this fairness notion is more limited than the general counterfactual fairness notion.

Graph counterfactual fairness. To address the above limitations, in this work, we propose a novel fairness notion on graphs:

Definition 2.1. *(Graph counterfactual fairness). An encoder $\Phi(\cdot)$ satisfies graph counterfactual fairness if for any node i:*

$$P((Z_i)_{S\leftarrow \mathbf{s}'} | X = \mathbf{X}, A = \mathbf{A}) = P((Z_i)_{S\leftarrow \mathbf{s}''} | X = \mathbf{X}, A = \mathbf{A}), \quad (3)$$

for all $\mathbf{s}' \neq \mathbf{s}''$, where $\mathbf{s}', \mathbf{s}'' \in \{0, 1\}^n$ are arbitrary sensitive attribute values of all nodes, $Z_i = (\Phi(X, A))_i$ denotes the node representations for node i. In other words, given a graph $X = \mathbf{X}, A = \mathbf{A}$, $\Phi(\cdot)$ should minimize the distribution discrepancy between the representations $(\Phi(X_{S\leftarrow \mathbf{s}'}, A_{S\leftarrow \mathbf{s}'}))_i$ and $(\Phi(X_{S\leftarrow \mathbf{s}''}, A_{S\leftarrow \mathbf{s}''}))_i$ for any node i.

Intuitively, this notion encourages the representations learned from the original graph and counterfactuals to be equal. The counterfactuals correspond to different cases when the sensitive attribute of the n nodes had been set to any values. For notation simplicity, in the following sections, we use $\mathbf{X}_{S\leftarrow \mathbf{s}'}$ to denote a specific value of the counterfactual "what would the node features have been if the sensitive attribute of the n nodes had been set by $\mathbf{s}'$, given the original data, i.e., node features $\mathbf{X}$ and graph structure $\mathbf{A}$?". We also use notation $\mathbf{A}_{S\leftarrow \mathbf{s}'}$ in a similar way.

In this work, we aim to develop a framework which learns node representations on graph towards graph counterfactual fairness, and maintains a good prediction performance simultaneously.

3 THE PROPOSED FRAMEWORK — GEAR

In this section, we propose a novel framework GEAR which aims to learn node representations for graph counterfactual fairness. As the illustration shown in Fig. 2, GEAR mainly includes three key components: 1) subgraph generation; 2) counterfactual data augmentation; 3) fair representation learning. In subgraph generation, GEAR extracts a context subgraph for each node, which contains the local graph structure including the node itself (central node) and its nearest neighbors with respect to precomputed importance scores. In counterfactual data augmentation, we generate counterfactuals in which the sensitive attribute of nodes in these subgraphs had been perturbed. Based on the augmented counterfactuals, the fair representation learning component leverages Siamese networks [7] to minimize the distance between the representations learned from the original data and the counterfactuals w.r.t. the same node.

3.1 Subgraph Generation

True causal models for graph data are often difficult to be completely obtained, especially for large-scale graphs. Based on a common observation [18, 21] that each node is mostly influenced by its nearest neighbors, we extract a subgraph $\mathcal{G}^{(i)}$ with node features $\mathbf{X}^{(i)}$ and adjacency matrix $\mathbf{A}^{(i)}$ for each node i. This subgraph extracts the context information of the central node i on $\mathcal{G}$, i.e., the subgraph of $\mathcal{G}$ which only contains the top k neighbors of node i (including itself). These top-k neighbors are usually within several hops from the central node. Specifically, for each node i on the graph, we generate its context subgraph $\mathcal{G}^{(i)}$ with a subgraph generator $\mathtt{Sub}(\cdot)$. Based on these context subgraphs, we learn the representations for their corresponding central nodes. This is based on a commonly used assumption [18] that each node has a low dependency with the nodes outside its context subgraph. Therefore, each subgraph is expected to be informative enough with respect to the graph structure relative to the central node for high-quality representation learning and counterfactual data augmentation afterwards.

Inspired by recent subgraph based node representation learning methods [21, 50], we first compute the importance scores for every node pair with personalized pagerank algorithm [20]. The importance scores can be calculated as: $\mathbf{R} = \alpha(\mathbf{I} - (1-\alpha)\overline{\mathbf{A}})$, where $\mathbf{R}$ is the importance score matrix, and each entry $\mathbf{R}_{i,j}$ describes how important node j is for node i, and $\mathbf{R}_{i,:}$ denotes the importance

[2] We use italicized uppercase letters (e.g., S_i, X, A) to denote random variables, and use italicized lowercase letters (e.g., s_i), non-italicized bold lowercase/uppercase letters (e.g., $\mathbf{x}_i$ and $\mathbf{X}$) to denote specific realization of scalars or vectors/matrices, respectively.

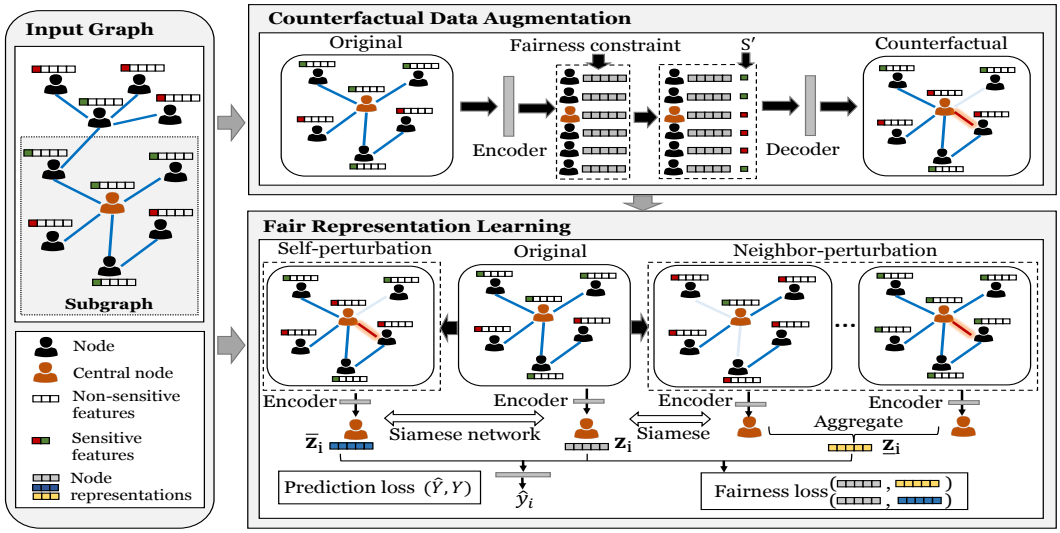


Figure 2: An illustration of the proposed framework GEAR.

score vector for node i. α is a parameter in the range of $[0, 1]$, $\mathbf{I}$ is the identity matrix. $\overline{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$ denotes the column-normalized adjacency matrix, where $\mathbf{D}$ is the corresponding diagonal matrix with $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{i,j}$. We compute $\mathbf{R}$ in a preprocessing stage before model training for efficiency. With the importance scores, we use a $\texttt{TOP}(\cdot)$ operation to select the top-k important nodes $\mathcal{V}^{(i)}$ for each central node i, then formulate the context subgraph $\mathcal{G}^{(i)}$ as follows:

$$\mathcal{G}^{(i)} = \{\mathcal{V}^{(i)}, \mathcal{E}^{(i)}, \mathbf{X}^{(i)}\} = \{\mathbf{A}^{(i)}, \mathbf{X}^{(i)}\}, \tag{4}$$

$$\mathcal{V}^{(i)} = \texttt{TOP}(\mathbf{R}_{i,:}, k), \tag{5}$$

$$\mathbf{A}^{(i)} = \mathbf{A}_{\mathcal{V}^{(i)},\mathcal{V}^{(i)}}, \ \mathbf{X}^{(i)} = \mathbf{X}_{\mathcal{V}^{(i)},:}, \tag{6}$$

where the symbol : means all the indices. The above subgraph generation process (Eq. (4) to (6)) is defined as $\mathcal{G}^{(i)} = \texttt{Sub}(i, \mathcal{G}, k)$. Then the generated subgraphs are fed into encoders to learn representations of the central nodes.

3.2 Counterfactual Data Augmentation

To achieve graph counterfactual fairness, we pretrain a counterfactual data augmentation module before node representation learning. Here we consider a relatively simple but general causal model (as shown in Fig. 1(b)) to generate counterfactuals for each subgraph. Based on common observations [25], we assume that the sensitive attribute (e.g., race) is exogenous, i.e., it has no parent variables in the causal graph, and it would causally influence the other node features, the graph structure, and the labels. Based on the causal model we assume, once we intervene on the sensitive attribute, we need to model how the other variables change accordingly. To achieve this goal, we use a graph variational auto-encoder (GraphVAE) [23] based module, which takes each context subgraph as input and encodes each node in the subgraph into a latent embedding $\mathbf{h}_i$, then a decoder reconstructs the original subgraph with the latent embeddings $\mathbf{H} = \{\mathbf{h}_1, ..., \mathbf{h}_k\}$ and the sensitive attribute values of the k nodes in this subgraph. The reconstruction loss $\mathcal{L}_r$ is as follows (we leave out the superscript $(\cdot)^{(i)}$ for notation simplicity):

$$\mathcal{L}_r = \mathbb{E}_{q(H|\mathbf{X},\mathbf{A})}[-\log(p(\mathbf{X}, \mathbf{A}|H, \mathbf{S}))] + \mathrm{KL}[q(H|\mathbf{X}, \mathbf{A})\|p(H)], \tag{7}$$

where $p(H)$ is a standard Normal prior distribution. We sample the embeddings $\mathbf{H}$ from $q(H|\mathbf{X}, \mathbf{A})$.

As the sensitive attribute is assumed to be exogenous, we can mitigate the causal effect from the sensitive attribute on the embeddings by removing the statistical dependency between them. To achieve this target, we use an adversarial learning method to learn embeddings which are invariant to different sensitive attribute values of each node and their neighbors. Specifically, we use a discriminator here to predict the summary of neighboring sensitive attribute values. Here we take the summary $\widetilde{s}_i$ as the mean aggregation over all the nodes in the subgraph $\mathcal{G}^{(i)}$, i.e., $\widetilde{s}_i = \frac{1}{|\mathcal{V}^{(i)}|}\sum_{j\in\mathcal{V}^{(i)}} s_j$. We divide the summary into B ranges to formulate it as a multivariate classification task for the discriminator $D(\cdot)$. We use a fairness constraint as follows: $\mathcal{L}_d = \sum_{b\in[B]} \mathbb{E}[\log(D(\mathbf{H}, b))]$,where the discriminator $D(\mathbf{H}, b)$ predicts the probability of whether the summary of sensitive attribute values is in range b. Based on the theoretic analysis in [6, 36], $\mathcal{L}_d$ is a regularizer to minimize the mutual information between the summary of sensitive attribute values and the embeddings. The final loss of the counterfactual data augmentation is: $\mathcal{L}_a = \mathcal{L}_r + \beta\mathcal{L}_d$, where β is a hyperparameter for the weight of fairness constraint. We use alternating stochastic gradient descent for optimization: 1) we minimize $\mathcal{L}_a$ by fixing the discriminator and updating parameters in other parts; 2) we minimize $-\mathcal{L}_a$ with respect to the discriminator while other parts fixed. To achieve graph counterfactual fairness, we expect the embeddings $\mathbf{H}$ can capture the latent variables which are informative of the input subgraph but not causally influenced by the sensitive attribute of the nodes in the subgraph. We pretrain the counterfactual data augmentation module to better disentangle different components of the framework. If more prior knowledge of the causal model is provided, we can incorporate it in counterfactual data augmentation, e.g., directly generate counterfactuals with a given causal model, and do not need to change other components in the framework.

Based on the above techniques, we conduct perturbations on the original subgraphs and obtain different types of counterfactuals. For each context subgraph $\mathcal{G}^{(i)}$, we generate two kinds of perturbations

on it, including self-perturbation on the sensitive attribute of the central node, and neighbor-perturbation on the sensitive attribute of other nodes in the subgraph.

Self-perturbation. In the subgraph $\mathcal{G}^{(i)}$, we take its node embeddings, flip the sensitive attribute value of the central node s_i, then feed the embeddings and the perturbed sensitive attribute into the decoder of the pretrained counterfactual data augmentation module, and take the reconstructed subgraph as the corresponding counterfactual. The set containing the subgraphs after self-perturbation is denoted by $\overline{\mathcal{G}}^{(i)} = \{\mathcal{G}^{(i)}_{S_i \leftarrow 1-s_i}\}$.

Neighbor-perturbation. Similarly, in the subgraph $\mathcal{G}^{(i)}$, we randomly perturb the sensitive attribute values of any nodes except the central node, i.e., the nodes in the set $\mathcal{V}^{(i)}_{\neg i}$. After such perturbation, we generate a set of counterfactuals $\underline{\mathcal{G}}^{(i)} = \{\mathcal{G}^{(i)}_{\mathbf{S}^{(i)}_{\neg i} \leftarrow \mathtt{SMP}(\mathbf{S}^{(i)}_{\neg i})}\}$, where $\mathtt{SMP}(\cdot)$ randomly samples specific values of the sensitive attribute out of the value space $\{0,1\}^{|\mathcal{V}^{(i)}|-1}$. We use a parameter C to denote the number of $\mathtt{SMP}(\cdot)$ operations in neighbor-perturbation.

3.3 Fair Representation Learning

Based on the above counterfactual data augmentation, we learn fair representations which are expected to elicit the same predicted label across different counterfactuals w.r.t. the same node. To achieve this goal, we leverage Siamese networks [7] to encode the three kinds of subgraphs: original subgraphs $\mathcal{G}^{(i)}$, counterfactual subgraphs $\overline{\mathcal{G}}^{(i)}$ and $\underline{\mathcal{G}}^{(i)}$ for each central node i. For graph counterfactual fairness, we expect to learn the same representations for each central node from the three kinds of subgraphs. We train a subgraph encoder $\phi(\cdot)$ to generate the representations $\mathbf{z}_i, \overline{\mathbf{z}}_i, \underline{\mathbf{z}}_i$ for each central node i on these three kinds of subgraphs, respectively. Then we minimize the distance between the central node representations learned from the original subgraph and from the counterfactuals. We formulate the loss for graph counterfactual fairness as:

$$\mathcal{L}_f = \frac{1}{|\mathcal{V}|}\sum\nolimits_{i\in\mathcal{V}}((1-\lambda_s)d(\mathbf{z}_i,\overline{\mathbf{z}}_i) + \lambda_s d(\mathbf{z}_i,\underline{\mathbf{z}}_i)), \tag{8}$$

where $d(\cdot)$ is a distance metric such as cosine distance. $\lambda_s \in [0,1]$ is a hyperparameter which controls the weight of neighbor-perturbation. From the original subgraph and the counterfactuals, we obtain the node representations in the following way:

$$\mathbf{z}_i = (\phi(\mathbf{X}^{(i)}, \mathbf{A}^{(i)}))_i, \tag{9}$$

$$\overline{\mathbf{z}}_i = \mathtt{AGG}(\{(\phi(\mathbf{X}^{(i)}_{S_i\leftarrow 1-s_i}, \mathbf{A}^{(i)}_{S_i\leftarrow 1-s_i}))_i\}), \tag{10}$$

$$\underline{\mathbf{z}}_i = \mathtt{AGG}(\{(\phi(\mathbf{X}^{(i)}_{\mathbf{S}^{(i)}_{\neg i}\leftarrow \mathtt{SMP}(\mathbf{S}^{(i)}_{\neg i})}, \mathbf{A}^{(i)}_{\mathbf{S}^{(i)}_{\neg i}\leftarrow \mathtt{SMP}(\mathbf{S}^{(i)}_{\neg i})}))_i\}), \tag{11}$$

where $\phi(\cdot) : \mathbb{R}^{k\times d} \times \mathbb{R}^{k\times k} \rightarrow \mathbb{R}^{k\times d'}$ takes each subgraph as input, and embeds each node on the input subgraph into a latent representation. We take the representations of each central node i learned from the original data as $\mathbf{z}_i$, and we use $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^n$ for downstream tasks. For the sampled counterfactual subgraphs in $\overline{\mathcal{G}}^{(i)}$ and $\underline{\mathcal{G}}^{(i)}$, we use an aggregator (e.g., mean aggregator) $\mathtt{AGG}(\cdot)$ to aggregate the representations of each central node i, and obtain the final representations $\overline{\mathbf{z}}_i$ and $\underline{\mathbf{z}}_i$.

To encode useful information of node features and graph structure into the representations, we use labels as supervision. We use the task of node classification as an example, but our framework can be naturally extended to other kinds of tasks on graph data such as link prediction. We denote the class labels as $\mathbf{Y} = \{y_1, ..., y_n\}$ for the n nodes. The prediction loss can be formulated as:

$$\mathcal{L}_p = \frac{1}{n}\sum\nolimits_{i\in[n]} l(f(\mathbf{z}_i), y_i), \tag{12}$$

where $l(\cdot)$ is the loss function (e.g., cross-entropy) which measures the prediction error, $f(\cdot)$ makes predictions for downstream tasks with the representations, i.e., $\widehat{y}_i = f(\mathbf{z}_i)$. Finally, the overall loss function for fair representation learning is:

$$\mathcal{L} = \mathcal{L}_p + \lambda\mathcal{L}_f + \mu\|\theta\|^2, \tag{13}$$

where θ is the set of model parameters, λ and μ are hyperparameters controlling the weight of the graph counterfactual fairness constraint, and L_2 norm regularization, respectively.

Table 2: Detailed statistics of the datasets.

Dataset	Synthetic	Bail	Credit
$\lvert\mathcal{V}\rvert$	2,000	18,876	30,000
$\lvert\mathcal{E}\rvert$	4,120	311,870	137,377
Feature dimension	26	18	13
Average degree	5.120	34.044	10.158
# of intra-group edges	2,379	162,821	120,750
# of inter-group edges	1,741	149,049	16,627

4 EXPERIMENTS

We evaluate the proposed method on both synthetic and real-world graphs. The detailed statistics of these datasets are shown in Table 2, including the number of nodes, the number of edges, the dimension of features, the average degree, and the number of intra-group and inter-group edges with respect to the sensitive attribute.

4.1 Datasets

A synthetic dataset and two real-world datasets are used in the experiments. In the synthetic dataset, we create a causal model with which we can fully manipulate the data generation process. More specifically, in this synthetic dataset, we generate the features, latent embeddings, graph structure, and labels as below:

$$S_i \sim \text{Bernoulli}(p),\ Z_i \sim \mathcal{N}(0,\mathbf{I}),\ X_i = \mathcal{S}(Z_i) + S_i\mathbf{v}, \tag{14}$$

$$P(A_{i,j}=1) = \sigma(\cos(Z_i,Z_j) + a\mathbf{1}(S_i=S_j)), Y_i = \mathcal{B}(\mathbf{w}Z_i + w_s\frac{\sum_{j\in\mathcal{N}_i}S_j}{|\mathcal{N}_i|}), \tag{15}$$

where we sample the sensitive attribute with Bernoulli distribution, where $p = 0.4$ is the probability of $S_i = 1$. We sample latent embeddings $Z_i \in \mathbb{R}^{d_z}$ from a Gaussian distribution, where $d_z = 50$. Z_i influences the node features and the graph structure for each node i, and $\mathcal{S}(\cdot)$ denotes a sampling operation which randomly selects $d = 25$ dimensions out of the latent embeddings to form the observed features X_i. $\mathbf{v} \in \mathbb{R}^d, \mathbf{v} \sim \mathcal{N}(0,\mathbf{I})$ controls the influence of the sensitive attribute on other features. We simulate the probability of each edge (i,j) based on the cosine similarity between Z_i and Z_j, as well as whether their sensitive attribute values are equal. Here $\mathbf{1}(\cdot)$ is an indicator function which outputs 1 when the input statement is true and 0 otherwise. We set parameter $a = 0.01$. Then we sum up the above similarity between (Z_i, Z_j) and the indicator function's output of $(S_i = S_j)$, and map it into a range

Table 3: Comparison of the performance of node representation learning methods with respect to prediction and fairness.

Dataset	Method	Prediction Performance			Fairness			
		Accuracy (↑)	F1-score (↑)	AUROC (↑)	$\triangle_{EO}$ (↓)	$\triangle_{DP}$ (↓)	δ_{CF} (↓)	R^2 (↓)
Synthetic	GCN	0.686 ± 0.015	0.687 ± 0.020	0.758 ± 0.017	0.050 ± 0.030	0.060 ± 0.033	0.101 ± 0.030	0.085 ± 0.050
	GraphSAGE	0.712 ± 0.012	0.714 ± 0.021	0.789 ± 0.018	0.049 ± 0.036	0.053 ± 0.042	0.172 ± 0.056	0.011 ± 0.011
	GIN	0.682 ± 0.021	0.691 ± 0.022	0.741 ± 0.021	0.077 ± 0.053	0.081 ± 0.055	0.301 ± 0.080	0.011 ± 0.009
	C-ENC	0.665 ± 0.023	0.671 ± 0.031	0.732 ± 0.028	0.030 ± 0.024	0.048 ± 0.026	0.633 ± 0.013	0.085 ± 0.016
	FairGNN	0.668 ± 0.020	0.672 ± 0.026	0.735 ± 0.022	**0.025 ± 0.021**	**0.042 ± 0.033**	0.678 ± 0.014	0.091 ± 0.021
	NIFTY-GCN	0.618 ± 0.035	0.640 ± 0.037	0.672 ± 0.042	0.172 ± 0.110	0.199 ± 0.106	0.208 ± 0.090	0.105 ± 0.081
	NIFTY-SAGE	0.664 ± 0.041	0.682 ± 0.073	0.755 ± 0.021	0.031 ± 0.027	0.048 ± 0.027	0.147 ± 0.071	0.008 ± 0.005
	GEAR	**0.718 ± 0.018**	**0.724 ± 0.022**	**0.793 ± 0.014**	0.052 ± 0.038	0.064 ± 0.038	**0.002 ± 0.002**	**0.007 ± 0.006**
Bail	GCN	0.838 ± 0.017	0.782 ± 0.023	0.885 ± 0.018	0.023 ± 0.019	0.075 ± 0.014	0.132 ± 0.059	0.075 ± 0.028
	GraphSAGE	**0.854 ± 0.026**	**0.804 ± 0.032**	**0.905 ± 0.021**	0.039 ± 0.022	0.086 ± 0.039	0.088 ± 0.047	0.069 ± 0.011
	GIN	0.731 ± 0.058	0.656 ± 0.084	0.773 ± 0.069	0.041 ± 0.023	0.065 ± 0.034	0.143 ± 0.069	0.047 ± 0.036
	C-ENC	0.842 ± 0.047	0.792 ± 0.014	0.889 ± 0.033	0.038 ± 0.022	0.069 ± 0.020	0.040 ± 0.025	0.078 ± 0.024
	FairGNN	0.835 ± 0.024	0.784 ± 0.021	0.882 ± 0.035	0.046 ± 0.013	0.074 ± 0.026	0.042 ± 0.032	0.086 ± 0.016
	NIFTY-GCN	0.752 ± 0.065	0.669 ± 0.050	0.799 ± 0.051	0.019 ± 0.015	**0.036 ± 0.022**	0.031 ± 0.017	0.025 ± 0.018
	NIFTY-SAGE	0.823 ± 0.048	0.723 ± 0.103	0.876 ± 0.043	**0.014 ± 0.006**	0.047 ± 0.015	0.013 ± 0.011	0.044 ± 0.020
	GEAR	0.852 ± 0.026	0.800 ± 0.031	0.896 ± 0.016	0.019 ± 0.023	0.058 ± 0.017	**0.003 ± 0.002**	**0.038 ± 0.012**
Credit	GCN	0.698 ± 0.028	0.794 ± 0.027	0.684 ± 0.019	0.087 ± 0.035	0.108 ± 0.031	0.042 ± 0.029	0.022 ± 0.014
	GraphSAGE	0.739 ± 0.009	0.821 ± 0.008	**0.756 ± 0.011**	0.094 ± 0.033	0.109 ± 0.030	0.062 ± 0.036	0.014 ± 0.004
	GIN	0.713 ± 0.018	0.805 ± 0.016	0.706 ± 0.010	0.121 ± 0.042	0.130 ± 0.037	0.123 ± 0.060	0.025 ± 0.012
	C-ENC	0.695 ± 0.011	0.786 ± 0.012	0.683 ± 0.018	0.098 ± 0.025	0.104 ± 0.042	0.100 ± 0.024	0.048 ± 0.012
	FairGNN	0.683 ± 0.053	0.780 ± 0.042	0.680 ± 0.021	0.175 ± 0.035	0.187 ± 0.036	0.105 ± 0.053	0.056 ± 0.018
	NIFTY-GCN	0.697 ± 0.007	0.792 ± 0.007	0.685 ± 0.007	0.097 ± 0.024	0.106 ± 0.021	0.004 ± 0.004	0.017 ± 0.003
	NIFTY-SAGE	0.751 ± 0.023	0.833 ± 0.020	0.730 ± 0.011	**0.075 ± 0.021**	**0.094 ± 0.019**	0.004 ± 0.003	0.011 ± 0.003
	GEAR	**0.755 ± 0.011**	**0.835 ± 0.008**	0.740 ± 0.008	0.086 ± 0.018	0.104 ± 0.013	**0.001 ± 0.001**	**0.010 ± 0.003**

of $[0, 1]$ with a Sigmoid function $\sigma(\cdot)$ to compute the link probability between (i, j). $\mathbf{w} \in \mathbb{R}^{d_z}$ contains parameters sampled from Normal distribution. We average each node's and their one-hop neighbors' sensitive attribute values and use it into label generation with weight $w_s = 0.5$. In $\mathcal{B}(\cdot)$, we map Y_i into a binary value. Specifically, we first compute the mean value of Y_i over all nodes, and set $Y_i = 1$ if it is larger than the mean value, otherwise $Y_i = 0$.

As for the real-world graphs, we use: 1) *Bail* [1]: This graph contains the data of defendants who got released on bail at the U.S state courts. In this graph, each node represents a defendant, each edge between a pair of nodes represents their similarity of criminal records and demographics. We use the defendants' race as the sensitive attribute. The task is to classify defendants into bail (not tend to commit a violent crime if released) or no bail. 2) *Credit defaulter* [1]: This graph contains people's default payment information. In this graph, each node represents an individual, each edge between a pair of nodes represents the similarity of their spending and payment patterns. We use their age as the sensitive attribute, and the task is to predict that their default ways of payment is credit card or not.

To evaluate the graph counterfactual fairness of the proposed method, we need to generate the ground-truth counterfactuals with the perturbations on different nodes' sensitive attribute. On the synthetic dataset, the counterfactuals can be generated based on the predefined causal model. On the real-world graphs, the ground-truth causal models are unknown, so we use a simple causal model and fit the observed data, and use the learned parameters in the fitted causal model to generate the counterfactuals for the whole graph. More specifically, we first use a Naïve Bayes model to learn $P(X_i|S_i)$, and then update the counterfactual features by $(\mathbf{x}_i)_{S_i \leftarrow 1-s_i} = \mathbb{E}[X_i|S_i = 1 - s_i] - \mathbb{E}[X_i|S_i = s_i] + \mathbf{x}_i$. We use $(\cdot)^{CF}$ to denote any counterfactuals. Then we generate the counterfactual graph edge for each (i, j) based on the following rules:

$$P(A_{i,j}^{CF} = 1) = \sigma(\cos(X_i^{CF} \backslash S_i^{CF}, X_j^{CF} \backslash S_j^{CF}) + \gamma \mathbf{1}(S_i^{CF}, S_j^{CF})), \quad (16)$$

where $\cos(\cdot)$ is cosine similarity. We use $\sigma(\cdot)$ to map its input into a range $[0, 1]$ to compute the link probability for any node pairs in the counterfactual graph. We fit the data with this causal model and learn the parameter γ. For evaluation, we use the counterfactual data generated by the causal model and learned parameters.

As discussed in [33], there might be multiple possible causal models in the real-world data, so we have tried different causal models to fit the real-world data. Due to the space limit, we only show the results based on the causal model as described above, but the observations over all the experiments are generally consistent.

4.2 Experiment Settings

Metrics. We evaluate the proposed framework with respect to two aspects: prediction performance and fairness. To evaluate the prediction performance, we use the widely-used node classification metrics: accuracy, F1-score, and AUROC. To measure the fairness of the representations, we first use two metrics which are commonly used in statistical fairness: $\triangle_{SP} = |P(\widehat{Y}_i|S_i = 0) - P(\widehat{Y}_i|S_i = 1)|$, and $\triangle_{EO} = |P(\widehat{Y}_i|Y_i = 1, S_i = 0) - P(\widehat{Y}_i|Y_i = 1, S_i = 1)|$. To evaluate graph counterfactual fairness, we design a metric δ_{CF}:

$$\delta_{CF} = |P((\widehat{Y}_i)_{S \leftarrow \mathbf{s}'}|X = \mathbf{X}, A = \mathbf{A}) - P((\widehat{Y}_i)_{S \leftarrow \mathbf{s}''}|X = \mathbf{X}, A = \mathbf{A})|, \quad (17)$$

where $\mathbf{s}', \mathbf{s}'' \in \{0, 1\}^n$ are arbitrary values of sensitive attribute of all nodes. As there are too many different counterfactuals (e.g., there are 2^n cases for a graph with n nodes), it is difficult to evaluate the difference of predictions under all these counterfactuals. Therefore, we evaluate the graph counterfactual fairness of the

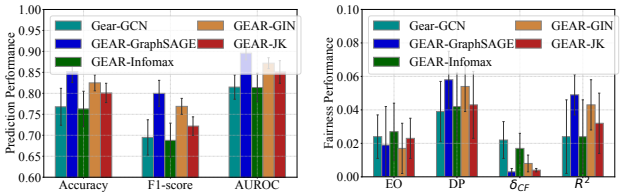
Table 4: Comparison of the performance of different variants of GEAR.

Dataset	Method	Prediction Performance			Fairness			
		Accuracy (↑)	F1-score (↑)	AUROC (↑)	$\triangle_{EO}$ (↓)	$\triangle_{DP}$ (↓)	δ_{CF} (↓)	R^2 (↓)
Synthetic	GEAR-NS	0.722 ± 0.023	0.726 ± 0.025	0.794 ± 0.028	0.061 ± 0.044	0.071 ± 0.024	0.005 ± 0.002	0.011 ± 0.006
	GEAR-NN	0.725 ± 0.026	**0.727 ± 0.016**	0.794 ± 0.024	0.066 ± 0.048	0.086 ± 0.033	0.008 ± 0.004	0.016 ± 0.005
	GEAR-NP	**0.729 ± 0.022**	**0.727 ± 0.027**	**0.796 ± 0.016**	0.094 ± 0.051	0.116 ± 0.063	0.012 ± 0.005	0.023 ± 0.018
	GEAR-NC	0.720 ± 0.019	0.725 ± 0.018	0.793 ± 0.018	0.058 ± 0.042	0.069 ± 0.028	0.006 ± 0.003	0.012 ± 0.004
	GEAR	0.718 ± 0.018	0.724 ± 0.022	0.793 ± 0.014	**0.052 ± 0.038**	**0.064 ± 0.038**	**0.002 ± 0.002**	**0.007 ± 0.006**
Bail	GEAR-NS	0.854 ± 0.020	0.802 ± 0.014	0.897 ± 0.020	0.027 ± 0.024	0.066 ± 0.020	0.014 ± 0.007	0.056 ± 0.018
	GEAR-NN	0.855 ± 0.024	**0.804 ± 0.024**	**0.898 ± 0.020**	0.032 ± 0.027	0.068 ± 0.023	0.022 ± 0.009	0.058 ± 0.016
	GEAR-NP	**0.860 ± 0.022**	**0.804 ± 0.031**	**0.898 ± 0.022**	0.041 ± 0.028	0.073 ± 0.028	0.027 ± 0.010	0.064 ± 0.019
	GEAR-NC	0.853 ± 0.024	0.801 ± 0.019	0.896 ± 0.021	0.025 ± 0.027	0.064 ± 0.014	0.007 ± 0.004	0.053 ± 0.014
	GEAR	0.852 ± 0.026	0.800 ± 0.031	0.896 ± 0.016	**0.019 ± 0.023**	**0.058 ± 0.017**	**0.003 ± 0.002**	**0.049 ± 0.012**
Credit	GEAR-NS	0.749 ± 0.014	0.831 ± 0.024	0.741 ± 0.011	0.089 ± 0.018	0.109 ± 0.038	0.016 ± 0.027	0.012 ± 0.005
	GEAR-NN	0.751 ± 0.012	0.832 ± 0.018	0.742 ± 0.009	0.092 ± 0.034	0.114 ± 0.043	0.020 ± 0.047	0.013 ± 0.004
	GEAR-NP	0.753 ± 0.018	**0.836 ± 0.017**	**0.749 ± 0.010**	0.099 ± 0.043	0.122 ± 0.049	0.028 ± 0.054	0.016 ± 0.007
	GEAR-NC	0.749 ± 0.015	0.830 ± 0.011	0.741 ± 0.009	0.088 ± 0.012	0.106 ± 0.011	0.004 ± 0.002	0.013 ± 0.004
	GEAR	**0.755 ± 0.011**	0.835 ± 0.008	0.740 ± 0.008	**0.086 ± 0.018**	**0.104 ± 0.013**	**0.001 ± 0.001**	**0.010 ± 0.003**

proposed model in the following way: on each dataset, we control the rate of sensitive subgroup population and randomly perturb the sensitive attribute of all nodes. More specifically, we randomly select 0%, 50%, 100% nodes, and set their sensitive attribute values to be 1, while set the sensitive attribute of other nodes to be 0. With such perturbations, we generate counterfactual data for the whole graph with different ratios of sensitive subgroup, based on the causal model described in Section 4.1. Intuitively, these perturbations implicitly control the distribution of the sensitive attribute in each node's neighborhood, and we take the averaged ratio of nodes which flip their predicted labels as an estimation for δ_{CF}. Besides, we also compute the R-square $R^2(\widehat{Y}_i, \widetilde{S}_i)$ to measure how well a linear regression predictor for $\widehat{Y}_i$ can be explained by the summary of the neighboring sensitive attribute values for any node i. Here we use the mean aggregator over the sensitive attribute values of all one-hop neighbors and each node i itself to compute the sensitive attribute summary $\widetilde{S}_i$. This R-square metric can reflect the statistical dependency between $\widehat{Y}_i$ and $\widetilde{S}_i$.

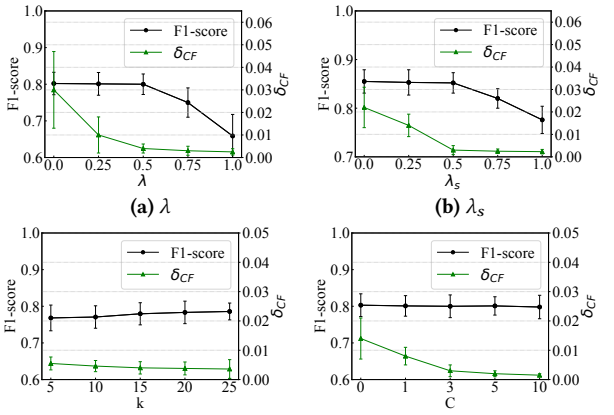
Baselines. We compare the proposed framework with several state-of-the-art node representation learning methods. We divide them into two categories: 1) node representation learning methods without fairness constraints: these methods only aim to encode useful information from the input graph and improve the prediction performance in downstream tasks. We use graph convolutional network (**GCN**) [24], **GraphSAGE** [18], and Graph Isomorphism Network (**GIN**) [44] as baselines; 2) fair representation learning methods on graphs: these methods target on learning fair node representation on graphs. Among them, **C-ENC** [6] and **FairGNN** [12] enforces fairness with an adversarial discriminator to predict the sensitive attribute; **NIFTY** [1] enforces fairness by maximizing the similarity of representations learned from the original graph and their augmented counterfactual graphs, where the sensitive attribute values of all nodes are flipped, while other parts remain unchanged. We use two variants of it with GCN or GraphSAGE encoders, denoted by **NIFTY-GCN** and **NIFTY-SAGE**, respectively.

Setup. Each dataset is randomly splitted into 60%/20%/20% training/validation/test set. Unless otherwise specified, we set the hyperparameters as $\lambda = 0.6$, $C = 2$, $\lambda_s = 0.4$, $\beta = 10$, $\mu = 1e-5$, $k = 20$, $B = 4$. The learning rate is 0.001, the number of epochs is 1,000, the representation dimension is 1,024, batch size is 100. Experimental results are averaged over ten repeated executions. We use the Adam optimizer and implement our method with Pytorch [31].



(a) Prediction **(b) Fairness**

Figure 3: Comparison of the performance of different subgraph encoders in GEAR on Bail dataset.



(a) λ **(b)** λ_s **(c) Subgraph size** k **(d) # of neighbor-perturbations** C

Figure 4: Parameter study on Bail dataset.

4.3 Prediction Performance and Fairness

The performance of prediction and fairness is shown in Table 3. The best results are shown in **bold**, and the runner-up results are underlined. Generally speaking, we have the following observations: 1) The proposed model GEAR shows comparable prediction

performance with the state-of-the-art node representation learning methods, and it outperforms all the fair node representation learning methods in prediction; 2) The proposed model outperforms all the other fair node representation learning methods in δ_{CF} and R^2. These two fairness metrics explicitly consider the causal/statistical relation between the neighboring sensitive attribute and the model prediction, thus this observation validates the effectiveness of our framework in mitigating the biases from neighbors. Besides, GEAR also performs well in other fairness metrics $\triangle_{EO}$ and $\triangle_{DP}$. The baseline NIFTY also has good performance in graph counterfactual fairness, because NIFTY also generates counterfactuals during training. Although NIFTY does not explicitly consider the causal effect from neighbors' sensitive attribute on each node, its counterfactuals still implicitly promote graph counterfactual fairness. However, our method still outperforms all these fairness methods mainly for two reasons: a) GEAR generates multiple versions of counterfactuals with self-perturbation and neighbor-perturbation. It has better coverage of the space of possible counterfactuals, while NIFTY only generates one counterfactual by flipping all nodes' sensitive attribute values, here the influence from the neighbors' sensitive attribute may counteract with each other; b) GEAR generates counterfactuals which include changes in both features and graph structure after modifying the sensitive attribute, rather than simply changing the sensitive attribute. More specifically, the counterfactual augmentation component in GEAR removes biases caused by misusing the descendants of the sensitive attribute in node representation learning.

4.4 Model Structure & Parameter Study

To investigate the model performance under different options of model structure, we vary the encoder of subgraphs, including GCN [24], GraphSAGE (mean aggregator) [18], Informax [38], GIN [44], and Jumping Knowledge (JK) [45]. Due to the space limit, we only show the results in Bail in Fig. 3, but the observations are consistent in other datasets. Generally, all the subgraph encoders show good performance in both prediction and fairness. Among them, GraphSAGE encoder shows the best performance over all the variants.

To evaluate the robustness of GEAR under different parameters, we vary the parameters λ (the weight of constraint for graph counterfactual fairness), λ_s (the weight for neighbor-perturbation), subgraph size k, and the number of neighbor-perturbations C in $\texttt{SMP}(\cdot)$ to investigate how the model performance varies. The results over different settings of parameters in Bail are shown in Fig 4. We observe that: 1) The model achieves better fairness with larger λ and λ_s while with slight sacrifice of the prediction performance; 2) The model achieves better fairness with relatively larger subgraph size, but this improvement becomes less significant when the subgraph size is over 20. 3) When the number of neighbor-perturbations in $\texttt{SMP}(\cdot)$ becomes larger, the predictions become more fair. These observations generally match our expectation.

4.5 Ablation Study

In ablation study, we compare different variants of GEAR to verify the effectiveness of different components. We first remove self-perturbations, denote this variant as GEAR-NS. Next, we remove neighbor-perturbations, denoted by GEAR-NN. We then remove all the perturbations, denote this variant as GEAR-NP. We remove the counterfactual data augmentation module, just flip the sensitive attribute values, and denote this variant as GEAR-NC. The model performance of these variants is shown in Table 4. We observe that all the variants perform worse than GEAR with respect to fairness. These results validate the effectiveness of different components in GEAR for learning fair node representations.

5 RELATED WORK

Graph representation learning. Many efforts have been made for graph representation learning [16, 17, 41] in recent years. Among them, neural network methods encode the attributes and graph structure into a latent space as representations to capture useful information. These methods include the well-known graph convolutional networks (GCNs) [24], variational graph auto-encoders (VGAE) [23], GraphSAGE [18], and structural deep network embedding (SDNE) [40]. Recently, subgraph-based methods [11, 21, 30] utilize the correlation between central nodes and their sampled subgraphs to capture regional structure information and improve the model scalability. Despite the success of these methods in different domains, they may exhibit biases against certain sensitive groups.
Fairness on graphs. Fairness in machine learning has attracted significant attention recently [5, 26, 39]. Typical fairness notions include group fairness [10, 13, 19, 47, 48], individual fairness [15, 35], and counterfactual fairness [9, 25, 33, 42]. Recent works [1, 6, 8, 12, 14, 22] promote fairness in node representation learning. Most of works are based on adversarial learning [3, 49], aiming to prevent the learned representations from accurately predicting the corresponding sensitive attribute. These works focus on removing the statistical dependency between the sensitive attribute and predictions elicited by the learned representations, but do not consider the biases in the features, graph structure or labels due to the causal effect from the sensitive attribute on them. Differently, few works [1, 2] extend counterfactual fairness to graphs. However, most of these works do not consider the potential biases brought by the sensitive attribute of neighboring nodes, and the causal effect from the sensitive attribute to other node features and graph structure.

6 CONCLUSION

In this paper, we propose a novel fairness notion of graph counterfactual fairness, which explicitly considers the causal influence from the neighboring nodes' sensitive attribute to each node, as well as the causal effect from the sensitive attribute to other features and the graph structure. We propose a novel framework GEAR to learn node representations which can achieve graph counterfactual fairness and good prediction performance simultaneously. Specifically, in GEAR, we use a counterfactual data augmentation module to generate counterfactuals with interventions on the sensitive attribute of different nodes. GEAR then maximizes the similarity between the node representations learned from the original data and different counterfactuals. Experimental results on synthetic and real-world graphs validate the effectiveness of our framework with respect to both prediction performance and fairness.

ACKNOWLEDGEMENTS

This material is supported by the National Science Foundation (NSF) under grants #1955151, #1934600, #2006844, a JP Morgan Chase Faculty Research Award, and a Cisco Faculty Research Award.

REFERENCES

[1] Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. 2021. Towards a Unified Framework for Fair and Stable Graph Representation Learning. *UAI* (2021).

[2] Chirag Agarwal, Marinka Zitnik, and Himabindu Lakkaraju. 2021. Towards a Rigorous Theoretical Analysis and Evaluation of GNN Explanations. *arXiv preprint arXiv:2106.09078* (2021).

[3] Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. 2017. Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075* (2017).

[4] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social network data analytics*. 115–148.

[5] Reuben Binns. 2020. On the apparent conflict between individual and group fairness. In *FAT*.

[6] Avishek Bose and William Hamilton. 2019. Compositional fairness constraints for graph embeddings. In *ICML*.

[7] Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. *IJPRAI* (1993).

[8] Maarten Buyl and Tijl De Bie. 2020. Debayes: a bayesian method for debiasing network embeddings. In *ICML*.

[9] Silvia Chiappa. 2019. Path-specific counterfactual fairness. In *AAAI*.

[10] Alexandra Chouldechova. 2017. Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. *Big data* (2017).

[11] Maxwell Crouse, Ibrahim Abdelaziz, Cristina Cornelio, Veronika Thost, Lingfei Wu, Kenneth Forbus, and Achille Fokoue. 2019. Improving graph neural network representations of logical formulae with subgraph pooling. *arXiv preprint arXiv:1911.06904* (2019).

[12] Enyan Dai and Suhang Wang. 2020. FairGNN: Eliminating the Discrimination in Graph Neural Networks with Limited Sensitive Attribute Information. *arXiv preprint arXiv:2009.01454* (2020).

[13] William Dieterich, Christina Mendoza, and Tim Brennan. 2016. COMPAS risk scales: demonstrating accuracy equity and predictive parity. *Northpoint Inc* (2016).

[14] Yushun Dong, Jian Kang, Hanghang Tong, and Jundong Li. 2021. Individual Fairness for Graph Neural Networks: A Ranking based Approach. In *KDD*.

[15] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *ITCS*.

[16] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. *ICLR Workshop on Representation Learning on Graphs and Manifolds* (2019).

[17] William L Hamilton. 2020. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning* (2020).

[18] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*.

[19] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *NIPS*.

[20] Glen Jeh and Jennifer Widom. 2003. Scaling personalized web search. In *IW3C2*.

[21] Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. 2020. Sub-graph contrast for scalable self-supervised graph representation learning. In *IEEE ICDM*.

[22] Jian Kang, Jingrui He, Ross Maciejewski, and Hanghang Tong. 2020. Inform: Individual fairness on graph mining. In *ACM SIGKDD*.

[23] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning* (2016).

[24] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ICLR* (2017).

[25] Matt J Kusner, Joshua R Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. *NIPS* (2017).

[26] Marta Z Kwiatkowska. 1989. Survey of fairness notions. *Information and Software Technology* (1989).

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

[28] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *JASIST* (2007).

[29] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM CSUR* (2021).

[30] Dang Nguyen, Wei Luo, Tu Dinh Nguyen, Svetha Venkatesh, and Dinh Phung. 2018. Learning graph representation via frequent subgraphs. In *SIAM*.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *NIPS* (2019).

[32] Judea Pearl et al. 2009. Causal inference in statistics: an overview. *Statistics surveys* (2009).

[33] Chris Russell, M Kusner, C Loftus, and Ricardo Silva. 2017. When worlds collide: integrating different counterfactual assumptions in fairness. In *NIPS*.

[34] Harjit Singh Sekhon, Sanjit Kumar Roy, and James Devlin. 2016. Perceptions of fairness in financial services: an analysis of distribution channels. *International Journal of Bank Marketing* (2016).

[35] Saeed Sharifi-Malvajerdi, Michael Kearns, and Aaron Roth. 2019. Average individual fairness: Algorithms, generalization and experiments. *NIPS* (2019).

[36] Yuhang Song, Wenbo Li, Lei Zhang, Jianwei Yang, Emre Kiciman, Hamid Palangi, Jianfeng Gao, C-C Jay Kuo, and Pengchuan Zhang. 2020. Novel human-object interaction detection via adversarial domain generalization. *arXiv preprint arXiv:2005.11406* (2020).

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

[38] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. *ICLR (Poster)* (2019).

[39] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *FairWare*. 1–7.

[40] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *ACM SIGKDD*.

[41] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *AAAI*.

[42] Yongkai Wu, Lu Zhang, and Xintao Wu. 2019. Counterfactual fairness: Unidentification, bound and algorithm. In *IJCAI*.

[43] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE TNNLS* (2020).

[44] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks?. In *ICLR*.

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

[46] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *NIPS*.

[47] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2017. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *WWW*.

[48] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *ICML*.

[49] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating unwanted biases with adversarial learning. In *AIES*.

[50] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. 2020. Graph-bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140* (2020).