BeMap: Balanced Message Passing for Fair Graph Neural Network

Xiao Lin[1], **Jian Kang**[2], **Weilin Cong**[3], **Hanghang Tong**[1]
[1]University of Illinois Urbana-Champaign, [2]University of Rochester, [3]Penn State University
[1]{xiaol13, htong}@illinois.edu, [2]jian.kang@rochester.edu, [3]wxc272@psu.edu

Abstract

Fairness in graph neural networks has been actively studied recently. However, existing works often do not explicitly consider the role of message passing in introducing or amplifying the bias. In this paper, we first investigate the problem of bias amplification in message passing. We empirically and theoretically demonstrate that message passing could amplify the bias when the 1-hop neighbors from different demographic groups are unbalanced. Guided by such analyses, we propose BeMap, a fair message passing method, that leverages a balance-aware sampling strategy to balance the number of the 1-hop neighbors of each node among different demographic groups. Extensive experiments on node classification demonstrate the efficacy of BeMap in mitigating bias while maintaining classification accuracy.

1 Introduction

Graph neural network (GNN) learns node representations via message passing, which aggregates the information from the local neighborhood of each node, to capture the topological and attributive characteristics of graph data [1–4]. Despite the substantial research progress, concrete evidence has shown that GNNs could carry certain biases, which lead to unfair learning results. For example, a graph-based recommender system may discriminate individuals from certain demographic groups by recommending fewer career opportunities [5, 6]. As such, the widespread use of GNNs in safe-critical and life-changing applications, such as employment systems [7], is in imminent need of fairness considerations.

In this paper, we study group fairness, which is one of the most intuitive and fundamental fairness notions. To date, researchers have developed a collection of algorithms to ensure group fairness for graph neural networks [8–12]. A typical approach is to impose the fairness consideration as a regularization term from the optimization perspective. Though achieving promising performance in bias mitigation, these methods often do not explicitly take into account the bias introduced or amplified by message passing during model training. To our knowledge, only a few works consider bias mitigation during message passing [12, 13]. They either manipulate the graph structure empirically [13, 14] or completely change the message passing procedure to solve a complex minimax problem [12, 15]. Several fundamental questions remain nascent in the context of algorithmic fairness on GNN: *(Q1) Does the message passing in GNN introduce or amplify bias? (Q2) If so, how to ensure fairness during message passing without changing the GNN architecture in a principled way?*

To answer these questions, we study the problems of (Q1) bias amplification in message passing and (Q2) fair message passing. For (Q1) bias amplification in message passing, we both empirically and theoretically show that message passing can indeed amplify bias when the numbers of neighboring nodes from different demographic groups are unbalanced. The key idea is to measure the bias as the reciprocal of the expected squared Euclidean distance of each node to the centroid of the demographic group it belongs to.[1] Based on the distance-based bias definition, for each node, if the numbers of neighbors from different demographic groups are unbalanced, the expected squared distance to the corresponding centroid would shrink after message passing, meaning that the dependence of the model on the sensitive attributes increases, and the bias in predictions is thus amplified. For (Q2) fair message passing in GCNs, guided by the theoretical insights from Q1, we propose BeMap that

[1]The more biased the model, the smaller the distance.

Lin et al., BeMap: Balanced Message Passing for Fair Graph Neural Network. *Proceedings of the Second Learning on Graphs Conference (LoG 2023)*, PMLR 231, Virtual Event, November 27–30, 2023.

leverages a balance-aware sampling strategy to selectively sample a subset of neighbors to achieve a balanced number of neighbors from different demographic groups for bias mitigation.

In summary, our main contributions are as follows:

- **Problems.** We study the problems of bias amplifications in message passing and fair message passing from the perspective of neighborhood balance of each node.
- **Analyses.** Our analyses *both empirically and theoretically* reveal that the message passing schema could amplify the bias, if the demographic groups with respect to a sensitive attribute in the local neighborhood of each node are unbalanced.
- **Algorithm.** Guided by our analyses, we propose an easy-to-implement sampling-based method named BeMap, which leverages a balance-aware sampling strategy to mitigate bias.
- **Evaluations.** We conduct extensive experiments on real-world datasets, which demonstrate that BeMap could significantly mitigate bias while maintaining a competitive accuracy compared with the baseline methods.

2 Related Work

Graph neural network (GNN) has demonstrated state-of-the-art performance in various learning tasks, including anomaly detection [16], crime rate prediction [17], and recommender systems [18]. [1] proposes the Graph Convolutional Network (GCN) by utilizing a localized first-order approximation of spectral graph convolutions. [2] introduces graph neural networks for inductive learning via neighborhood sampling and aggregation. [19] leverages the self-attention mechanism to learn the attention weights for neighborhood aggregation. [3] scales up the training on large graphs by importance sampling. [20] learns node representations by randomly dropping nodes to augment data and enforcing the consistency of predictions among augmented data. Similarly, [21] randomly drops nodes for several runs and aggregates the predictions for the final prediction by ensemble. Different from [3, 20, 21] that drop nodes for scalable training or improving the generalization and/or expressiveness of GNN, our work drops edges (i.e., dropping nodes in the local neighborhood of a node) to mitigate bias in GNN. [22] randomly drops edges to perform data augmentation and prevents over-smoothing. Different from [22] that randomly drops edges to alleviate over-smoothing, our work selectively removes edges to obtain balanced graph structures to improve the model fairness. For comprehensive literature reviews on graph neural networks, please refer to [23–27].

Fairness in graph neural networks has been actively studied. A common strategy to learn fair graph neural networks is through optimizing a regularization-based optimization problem [8, 9, 11, 28, 29]. For group fairness, [8] ensures group fairness by minimizing the mutual information between sensitive attributes and node embeddings via adversarial learning. [9] leverages a similar debiasing strategy to learn fair graph neural networks with limited sensitive attributes. For individual fairness, [30] mitigates individual bias using the Laplacian regularization on the learning results. [31] adopts learning-to-rank for ranking-based individual fairness. In terms of counterfactual fairness, [28] learns counterfactually fair node embeddings through contrastive learning. [29] generates the counterfactual graph with variational graph auto-encoder [32]. Different from the aforementioned techniques, our work does not require any regularization in the objective function. Other than regularization-based formulation, [11] preprocesses the input graph by minimizing the Wasserstein distance between the embedding distributions of majority and minority groups. [14] generates fair neighborhoods by neighborhood rewiring and selection. [12] proposes a novel message passing schema that solves a minimax problem w.r.t. group fairness. [13] promotes the existence of edges connecting nodes in different demographic groups. However, [12] bears little resemblance to the original message passing schema, thus completely changing the training procedures; [14] lacks a theoretical understanding on the connection between balanced neighborhood and group fairness; [11] requires a pre-trained model to pre-process the input graph; and [13] cannot guarantee a balanced neighborhood consequently. Compared to [11–14], our work fills the gap between a balanced neighborhood and group fairness and tunes the message passing in a easy-to-implement node sampling strategy with theoretical guarantee. Further discussion on prior works related to fairness in GNNs is provided in Appendix I.

3 Preliminaries

In this section, we first briefly introduce the Graph Convolutional Network (GCN) and the commonly used group fairness definitions. Then, we formally define the problem of bias amplification in message passing and fair message passing in GCN.

Notation Convention. We use bold uppercase/lowercase letters for matrices/vectors (e.g., $\mathbf{A}$, $\mathbf{x}$), italic letters for scalars (e.g., d), and calligraphic letters for sets (e.g., $\mathcal{N}$). For matrix indexing, the

i-th row of a matrix is denoted as its corresponding bold lowercase letter with subscript i (e.g., the i-th row of matrix $\mathbf{X}$ is $\mathbf{x}_i$). Notations are summarized in Table 3.

Graph Convolutional Networks. In this paper, we study the message passing schema in Graph Convolutional Network (GCN) [1], which is one of the most classic graph neural networks. Let $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$ denote a graph with a node set of n nodes $\mathcal{V} = \{v_1, \ldots, v_n\}$, a binary adjacency matrix $\mathbf{A}$, and node feature matrix $\mathbf{X}$. For any node v_i, we denote its degree and the feature as d_i and $\mathbf{x}_i$. [2] For the l-th hidden layer in an L-layer GCN, we denote its weight matrix as $\mathbf{W}^{(l)}$ and the input and output representations of node v_i as $\mathbf{h}_i^{(l)}$ and $\mathbf{h}_i^{(l+1)}$, respectively.[3] Then the message passing schema in GCN is $\widehat{\mathbf{h}}_i^{(l)} = \sum_{v_j \in \widehat{\mathcal{N}}(v_i)} \alpha_{ij} \mathbf{h}_j^{(l)}$, where $\widehat{\mathcal{N}}(v_i) = \mathcal{N}(v_i) \cup \{v_i\}$ is the self-augmented neighborhood (i.e., the union of node v_i and its 1-hop neighborhood $\mathcal{N}(v_i)$) and α_{ij} is the aggregation weight with the source node being v_i and the target node being v_j (e.g., $\alpha_{ij} = \frac{1}{\sqrt{d_i+1}\sqrt{d_j+1}}$ for symmetric normalization and $\alpha_{ij} = \frac{1}{d_i+1}$ for row normalization). Based on the message passing schema, the graph convolution for v_i in GCN can be formulated as $\mathbf{h}_i^{(l+1)} = \sigma\left(\widehat{\mathbf{h}}_i^{(l)} \mathbf{W}^{(l)}\right)$, where $\sigma(\cdot)$ is the nonlinear activation (e.g., ReLU).

Group Fairness. Group fairness aims to ensure the parity of model predictions among the demographic groups of data points, where the demographic groups are often determined by a sensitive attribute (e.g., gender and race). Specifically, we adopt two widely used fairness criteria, i.e., statistical parity [33] and equal opportunity [34], which are defined in Definitions 1 and 2, respectively.

Definition 1. ***(Statistical parity [33])*** *Given any label $y \in \{0, 1\}$, any sensitive attribute $s \in \{0, 1\}$ and the prediction $\widehat{y} \in \{0, 1\}$ inferred by a model, a model satisfies statistical parity if and only if the acceptance rate with respect to the model predictions are equal for different demographic groups. Mathematically, statistical parity can be expressed as*

$$Pr(\widehat{y} = 1 \mid s = 0) = Pr(\widehat{y} = 1 \mid s = 1) \tag{1}$$

where $Pr(\cdot)$ refers to the probability of an event.

Definition 2. ***(Equal opportunity [34])*** *Following the settings of Definition 1, a model satisfies equal opportunity if and only if the true positive rate with respect to the model predictions are equal for different demographic groups. Mathematically, equal opportunity can be expressed as*

$$Pr(\widehat{y} = 1 \mid y = 1, s = 0) = Pr(\widehat{y} = 1 \mid y = 1, s = 1) \tag{2}$$

where $Pr(\cdot)$ refers to the probability of an event.

Given Definitions 1 and 2, the bias with respect to statistical parity and equal opportunity are naturally defined as the discrepancies in the acceptance rate and the true positive rate across different demographic groups. Mathematically, the quantitative measures of bias with respect to statistical parity and equal opportunity are defined as Eq. (3) and Eq. (4), respectively.

$$\Delta_{\text{SP}} = |Pr(\widehat{y} = 1 \mid s = 1) - Pr(\widehat{y} = 1 \mid s = 0)| \tag{3}$$

$$\Delta_{\text{EO}} = |Pr(\widehat{y} = 1 \mid y = 1, s = 1) - Pr(\widehat{y} = 1 \mid y = 1, s = 0)| \tag{4}$$

From the information-theoretic perspective, minimizing Eq. (3) and Eq. (4) is essentially equivalent to eliminating the statistical dependency between the model prediction $\widehat{y}$ and the sensitive attribute s. Consequently, to ensure group fairness, existing works [8, 9, 11] propose to impose the statistical dependency (e.g., mutual information) between $\widehat{y}$ and s as regularization in the optimization problem. Nevertheless, these works could not explicitly consider the bias caused by the message passing schema. To this end, we seek to understand the role of message passing in the context of algorithmic fairness. Based on that, we define the problem of bias amplification in message passing as follows:

Problem 1. *Bias amplification in message passing.*

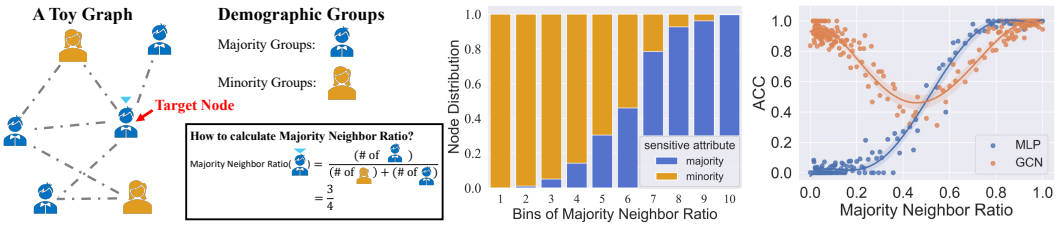
Given: (1) An undirected graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$; (2) an L-layer GCN; (3) the vanilla message passing schema in any l-th hidden layer $\widehat{\mathbf{h}}_i^{(l)} = \sum_{v_j \in \mathcal{N}(v_i) \cup \{v_i\}} \alpha_{ij} \mathbf{h}_j^{(l)}$; (4) a sensitive attribute s; (5) a bias measure *bias* for statistical parity.

Find: A binary decision regarding whether or not the bias will be amplified after message passing.

Based on the answer to Problem 1, our goal is to develop a generic fair message passing such that the bias measure in Problem 1 will decrease after message passing. We define the problem of fair message passing as follows:

[2] Although we only consider binary adjacency matrix, our theoretical analysis and proposed method can be naturally generalized to graph with weighted edges by replacing 0/1 edge weight to other values.

[3] For notation simplicity, we denote $\mathbf{h}_i^{(1)} = \mathbf{x}_i$ for all $v_i \in \mathcal{V}$.



(a) A toy example of calculating majority neighbor ratio.

(b) The correlation between the sensitive attribute of a node and its neighbors.

(c) Empirical evidence of bias amplification in GCN.

Figure 1: The empirical evidence of bias amplification in GCN on the Pokec-z dataset. Best viewed in color. In (b), majority neighbor ratio is grouped into 10 equal-width bins with width being 0.1, i.e., $[0, 0.1), [0.1, 0.2), \ldots, [0.9, 1.0]$.

Problem 2. *Fair message passing.*

Given: (1) An undirected graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$; (2) an L-layer GCN; (3) a sensitive attribute s; (4) a bias measure Δ_{SP}.

Find: A fair message passing $\widehat{\mathbf{h}}_i^{(l)} = MP\left(\mathbf{h}_i^{(l)}, \mathcal{G}\right)$ such that Δ_{SP} decreases after message passing.

4 Bias Amplification in Message Passing

In this section, we provide both the empirical evidence and the theoretical analysis on bias amplification in message passing.

4.1 Empirical Evidence

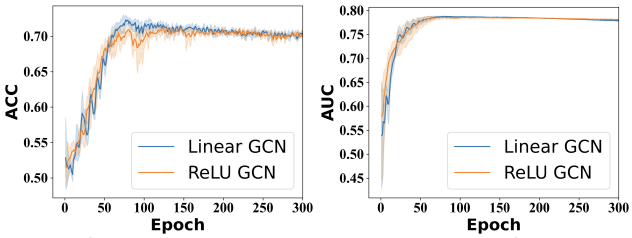
We first empirically illustrate that message passing could amplify the bias. The design of our experiment is based on the assumption that the learning results (e.g., class probabilities) from a fair model (e.g., FairGNN [9]) should be independent of the sensitive attribute [9]. Therefore, we use a logistic regression classifier to predict the sensitive attribute of a node using its corresponding learning result output by a model, where the accuracy of the sensitive attribute estimator naturally serves as an indicator of how biased a model is.

We investigate the effect of bias amplification of message passing on the Pokec-z dataset by comparing the behavior of a two-layer multi-layer perception (MLP) and a two-layer GCN.[4] Note that the only difference between the MLP and GCN is whether the message passing is utilized or not, while the hidden dimensions and the nonlinear activation function are the same with each other. The details for the empirical evaluation are as follows. First, we train the MLP and GCN to predict labels using the node features without sensitive attributes as input. When the graph neural networks reach the best performance, we freeze the parameters of the first layer of both the MLP and GCN. Hence, the first layer of the MLP and GCN can serve as embedding extractors. Second, we train two logistic regression classifiers to predict sensitive attributes with the embeddings extracted from the MLP and GCN, respectively. If the classifier can accurately predict the sensitive attribute, it implies that those embeddings contain rich sensitive information. In this way, we use the accuracy as a proxy to measure the dependency between sensitive attributes and embeddings extracted from the two models.

The evaluation results are presented in Figure 1. From the figure, we have three key observations. First, from Figure 1(b), we can observe a strong positive correlation between the sensitive attribute of a node and the sensitive attribute of its neighbors, i.e., a node tends to have the same sensitive attribute as the majority of its neighbors. Second, as shown in Figure 1(c), the MLP predicts the sensitive attribute of all nodes as the sensitive attribute of the majority demographic group, even when a node and all its neighbors do not belong to the majority demographic group. Third, in Figure 1(c), the GCN makes much more accurate predictions than the MLP when the neighbors of a node are more likely to be in the minority group. The first and second observations imply that the embeddings extracted from the MLP have little correlation with sensitive attributes, thus leading to the less biased predictions. On the contrary, the GCN extracts the embeddings containing rich sensitive information for those nodes that belong to the same demographic group as their neighbors, as indicated in the third observation, regardless of the minority group and the majority group. These results demonstrate that message passing could aggregate the sensitive information from the neighborhood, which further amplifies the bias and guides the logistic regression classifier to correctly predict the sensitive attribute.

4.2 Theoretical Analysis

[4] Details about the Pokec-z dataset is deferred to Section 6 and Appendix G.1.



(a) Classification accuracy (ACC) for each training epoch.

(b) AUC score (AUC) for each training epoch.

Figure 2: Node classification accuracy (ACC) and AUC score (AUC) curves of linear GCN (Linear GCN) vs. nonlinear GCN (ReLU GCN) on the NBA dataset.

For analysis purposes, we consider a binary sensitive attribute and an L-layer linear GCN with row normalization (i.e., $\alpha_{ij} = \frac{1}{d_i+1}$) for binary node classification, where linear GCN is a special type of GCN model without the nonlinear activation (e.g., [4]).[5] Although our analysis relies on linearity, recent works have shown that lack of nonlinearity in GCN enjoys almost the same expressive power as nonlinear GCN [35, 36]. Moreover, as shown in Figure 2, linear GCN exhibits similar or even slightly better node classification accuracy than its nonlinear counterpart, which also demonstrates that linearizing GCN is a good alternative to understand the behavior of GNN models.

To analyze the connection between the bias amplification phenomenon and graph topology, we make the assumption (Assumption 1) concerning the generation of graphs. Specifically, we use random graphs to analyze the key properties in Proposition 1 of GNNs, which has been widely adopted in previous works [12, 37, 38]. For example, FMP [12] investigates the connection between group fairness and graph properties using random graphs as the basis for analysis. OOD-GNN [38] examines the out-of-distribution generalization abilities of GNNs on specific random graphs with distribution shifts. And Keriven et al. [37] explores the convergence and stability of GCNs by analyzing their behavior on standard models of random graphs.

Assumption 1. *The graphs discussed in theoretical analysis are generated by the Gilbert random graph model [39], in which each edge is added independently with a fixed possibility P.*

To measure how biased a GNN is after message passing, we further make the following assumption on the existence of ideal fair node embedding. The intuition is that a fair graph neural network would output fair node embeddings without sensitive information after the last hidden layer. Following this intuition, a node embedding output by any GNN can be decomposed into a fair node embedding and the residual (i.e., for a fair GNN, the residual would be 0).

Assumption 2. *The output node embedding $\mathbf{z}$ from the last hidden layer is a linear combination of fair embedding $\mathbf{z}_t$ and bias residual $\mathbf{z}_b$, i.e., $\mathbf{z} = \mathbf{z}_t + \mathbf{z}_b$, where the fair embedding gives fair prediction while the bias residual corresponds to the bias in predictions. Written in matrix form, we have $\mathbf{Z} = \mathbf{Z}_t + \mathbf{Z}_b$.*

Supposing that Assumption 2 holds, we show in Lemma 1 that for any $v_i \in \mathcal{V}$, the output embedding from any hidden layer can be viewed as a linear combination of fair embedding and bias residual. Proof is deferred in Appendix B.

Lemma 1. ***(Linear decomposition of a node embedding)*** *Suppose that Assumption 2 holds. Given an input graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$ and an L-layer linear GCN with row normalization, we have*

$$\mathbf{h}_i^{(l)} = \left(\mathbf{t}_i^{(l)} + \mathbf{b}_i^{(l)}\right) \mathbf{W}^{(1)} \dots \mathbf{W}^{(l-1)} \tag{5}$$

for any $v_i \in \mathcal{V}$ and any hidden layer $l \in \{1, \dots, L\}$, where $\mathbf{t}_i^{(l)} = \widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^L)^\dagger \mathbf{z}_t \mathbf{W}^\dagger$ is the fair embedding, $\mathbf{b}_i^{(l)} = \widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^L)^\dagger \mathbf{z}_b \mathbf{W}^\dagger$ is the bias residual, $\widetilde{\mathbf{A}} = \mathbf{D}^{-1}\mathbf{A}$ is the row-normalized adjacency matrix, and $\mathbf{W}^\dagger$ is the Moore–Penrose inverse of $\mathbf{W} = \mathbf{W}^{(1)}\mathbf{W}^{(2)} \dots \mathbf{W}^{(L)}$.

Lemma 1 offers a fresh perspective on the interplay between message passing and fairness. In particular, we will show that message passing alters the distribution of bias residuals associated with all nodes, thereby influencing fairness. Specifically, when the bias residuals of all nodes converge to the same point after message passing, the output node embeddings from the last hidden layer equals to the sum of fair node embeddings and a constant vector, resulting in fair prediction results. Conversely, if the differences between the bias residuals from different demographic groups are magnified through message passing, the unfairness in prediction results will be also amplified. Therefore, the distributional shift in the bias residual reveals how message passing would affect the fairness.

[5] Our analysis can be generalized to non-binary sensitive attribute and multi-class classification.

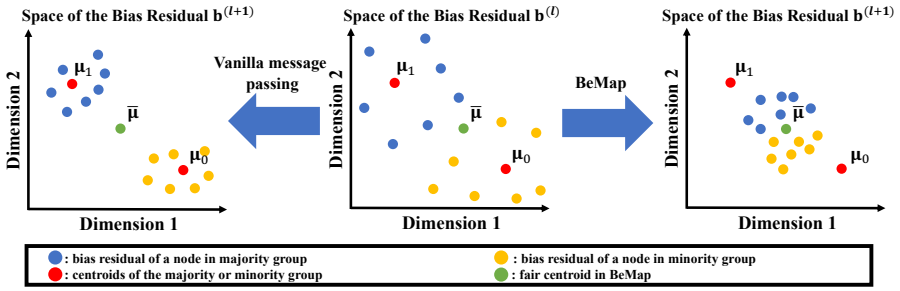


Figure 3: An illustrative example of BeMap. After BeMap, the bias residuals will move towards the fair centroid (the green point), whereas, after the vanilla message passing in GCN, they will move towards the centroid of the majority group and the minority group (the red points).

For GCNs with row normalization, message passing essentially performs a weighted average over the bias residuals from local neighborhoods. Since the summation of the weight from all the neighbors is 1 for row normalization, the centroid of the distribution of the bias residual for any demographic group remains unchanged during message passing. Please see the detailed proof in Appendix C. In this case, an increase in the difference of bias residuals is directly related to the shrinkage in the expected distance between the bias residual of any node and the centroid of its corresponding demographic group. For example, if the expected distance is much larger than the distance between centroids, then the distributions of bias residuals from various demographic groups likely exhibit substantial overlapping areas. Within this overlap, identifying the demographic group to which a specific bias residual belongs becomes challenging. This suggests that when the distance between the centroids of two demographic groups is fixed, an increase in the expected distance results in a considerable expansion of the overlapping area. Consequently, the difference between the two distributions diminishes. Meanwhile, an extreme unfair situation is that the bias residual of any node exactly falls into the centroid of its demographic group. Then the bias residual essentially becomes an indicator for sensitive attributes. As a result, even if the input data excludes any information related to sensitive attributes, GNNs can readily distinguish the sensitive attribute associated with each node. It grants GNNs a convenient path to infer labels through the leaked information of sensitive attributes, thus leading to severe unfairness.

To better understand the change of the expected distances during message passing, we introduce a distance-based bias that captures the model bias. Let us denote the centroid of the bias residuals from the majority as μ_0 and denote the centroid from the minority as μ_1.

Definition 3. ***(Distance-based bias)*** *Suppose that we have an input graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$ and an L-layer GCN. For any l-th hidden layer, the distance-based bias is defined as the reciprocal of the expected squared Euclidean distance from the bias residual of each node to the centroid of its corresponding demographic group, which is shown below.*

$$bias^{(l)}(\mathcal{G}, \mathbf{B}^{(l)}, s) = \frac{1}{\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l)} - \mu(v_i)\|_2^2]} \tag{6}$$

where $\mathbf{b}_i^{(l)}$ is the i-th row of $\mathbf{B}^{(l)}$, and $\mu(v_i)$ is the centroid of the bias residuals of all nodes belonging to the demographic group of node v_i.

With the distance-based bias, we furthermore show in Theorem 1 that the expected squared distance would shrink after message passing in any l-th hidden layer, which is equivalent to the amplification of the distance-based bias. Proof is deferred to Appendix C.

Theorem 1. ***(Distance shrinkage)*** *Suppose we have an L-layer linear GCN with row normalization and an input graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$. In the l-th hidden layer, let $\mathbf{b}_i^{(l)}$ and d_i be the biased embedding and the degree of any $v_i \in \mathcal{V}$, respectively. For any l-th hidden layer, we have*

$$\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l+1)} - \mu(v_i)\|_2^2] = \mathbb{E}_{v_i}[\frac{1}{d_i}]\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l)} - \mu(v_i)\|_2^2], \tag{7}$$

which means bias amplification after message passing $\left(bias^{(l)}(\mathcal{G}, \mathbf{B}^{(l+1)}, s) < bias^{(l)}(\mathcal{G}, \mathbf{B}^{(l)}, s)\right)$.

Theorem 1 demonstrates that message passing drives the convergence of bias residuals towards their centroids, thereby magnifying the differences in the distributions of bias residuals of different demographic groups. Moreover, as the expectation of node degrees increases, the distance shrinks at a faster rate. This distance shrinkage exacerbates distance-based biases, causing unfair predictions.

5 BeMap: A Fair Message Passing Schema

In this section, we first present how to avoid bias amplification in message passing and then propose a fair message passing method named BeMap.

5.1 Fair Message Passing

Motivated by the observation in Theorem 1 that bias amplification can lead to unfairness, we put forward the idea that a fair message passing schema should possess the capability to automatically diminish such differences of bias residuals across various demographic groups. Hence, a fair message passing schema entails two primary objectives: (1) **centroid consistency**. The centroid of bias residuals becomes the same across different demographic groups after fair message passing; (2) **distance shrinkage**. The distance between bias residuals and their centroids will shrink through message passing. An illustrative example is presented in Figure 3.

To achieve our first objective **centroid consistency**, as shown in Lemma 2, we have to make sure the proportion of neighbors from each demographic group is similar.

Lemma 2. *(**Sufficient condition for centroid consistency**) Suppose Assumption 1 and Assumption 2 hold, and we are given an input graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$ and an L-layer linear GCN with row normalization. For any node $v_i \in \mathcal{V}$, let $\widehat{N}_0(v_i)$ and $\widehat{N}_1(v_i)$ be the number of neighbors in $\widehat{\mathcal{N}}(v_i) = \mathcal{N}(v_i) \cup \{v_i\}$ that belong to the minority group and majority group, respectively. In the l-th hidden layer, the centroid of the bias residual $\mathbf{b}_i^{(l)}$ keeps the same for any $v_i \in \mathcal{V}$ when*

$$\frac{\widehat{N}_0(v_i)}{\widehat{N}_1(v_i)} = \frac{r_0}{r_1}, \quad \forall v_i \in \mathcal{V} \tag{8}$$

where $r_s, s \in \{0, 1\}$ is the ratio of neighbors from demographic group s, and $r_0 + r_1 = 1$.

Lemma 2 states that, as long as the self-augmented neighborhood of any node in the graph has the same ratio of neighbors from each demographic group, the bias residuals would be centered around the same centroid. Actually, the new fair centroid, which is denoted as $\bar{\mu}$, can be expressed as a linear combination of the original centroid of each demographic group, *i.e.*, $\bar{\mu} = r_0\mu_0 + r_1\mu_1$. Therefore, we can change the position of the fair centroid $\bar{\mu}$ by changing the ratio of each demographic group. However, if edge deletion is used to control the ratio for every node, then the ratio r_s are not supposed to be some extreme values, *i.e.*, 0 or 1. It is because that those extreme values will result in the removal of a large number of edges, thus leading to a severe degradation in the utility of the model. To ensure the balance between utility and fairness, we set the ratio of the minority group and majority group to be the same, *i.e.*, $r_0 = r_1 = \frac{1}{2}$.

To achieve our second objective **distance shrinkage**, as shown in Theorem 2, by increasing the balanced self-augmented neighborhood to sufficiently large, the expected squared distance between the bias residual and their fair centroids $\bar{\mu}$ will shrink after message passing via self-augmented neighborhood. Proof is deferred to Appendix D.

Theorem 2. *(**Fair message passing**) Suppose Lemma 2 holds. As long as $\widehat{N}_0(v_i) + \widehat{N}_1(v_i) \geq 4$, $\forall v_i \in \mathcal{V}$, the expected squared distance between the bias residual $\mathbf{b}_i^{(l)}$ and the fair centroid $\bar{\mu}$ will shrink after message passing. Mathematically, we have*

$$\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l+1)} - \bar{\mu}\|_2^2] < \mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l)} - \bar{\mu}\|_2^2], \quad l \in \{1, \ldots, L\} \tag{9}$$

5.2 BeMap Algorithm

To achieve **centroid consistency** and **distance shrinkage**, we propose a fair message passing algorithm named BeMap, whose key idea is to perform message passing on a sufficiently large and balanced self-augmented neighborhood (which we call fair neighborhood). To obtain the fair neighborhood, we consider sampling (i.e., edge deletion) over the original self-augmented neighborhood in BeMap. Note that other types of neighborhood augmentation techniques (e.g., edge addition, edge rewiring) could also be applied to obtain the fair neighborhood, which we leave for future work. Hereafter, BeMap is referred to as message passing over the sampled fair neighborhood.

In practice, there are two main challenges in obtaining the fair neighborhood in BeMap due to the long-tailed in many real-world networks. First, there could exist some node v_i such that $\widehat{N}_0(v_i) = \widehat{N}_1(v_i)$ and $\widehat{N}_0(v_i) + \widehat{N}_1(v_i) \geq 4$ cannot be satisfied simultaneously. In this case, we apply the following two empirical remedies.

- If all neighbors in $\widehat{\mathcal{N}}(v_i)$ of a node v_i belong to one demographic group, we sample a subset of k nodes where $k = \max\{4, \beta|\mathcal{N}(v_i)|\}$, $|\mathcal{N}(v_i)|$ is the cardinality of $\mathcal{N}(v_i)$ and β is a hyperparameter. The intuition is that when the local neighborhood is sufficiently large to satisfy Theorem 2 (*i.e.*, $k \geq 4$), decreasing the number of neighbors (i.e., the degree d_i of node v_i) helps reduce the difference between the expected squared distances before and after message passing as shown in Eq. (7). Thus, it helps decelerate the bias residuals moving towards the original centroid of the demographic group of v_i.
- Otherwise, we keep the sampled neighborhood to be balanced, i.e., $\widehat{N}_0(v_i) = \widehat{N}_1(v_i)$, by sampling over $\mathcal{N}_0(v_i)$ if $\widehat{N}_0(v_i) > \widehat{N}_1(v_i)$ or $\mathcal{N}_1(v_i)$ if $\widehat{N}_0(v_i) < \widehat{N}_1(v_i)$. By balancing the sampled neighborhood, it helps shrink the bias residual to converge into the fair centroid $\bar{\mu}$.

Second, there might exist some node v_i such that its neighbors within L hops could contain node(s) whose neighbors always belong to only one demographic group. Note that the receptive field of the l-th hidden layer in an L-layer GCN is the l-hop neighborhood of a node. Then, for such node v_i, it is hard to maintain a balanced l-hop neighborhood if we apply uniform sampling, so as to satisfy Lemma 2. To alleviate this issue, we propose *balance-aware sampling*. Its key idea is to adjust the sampling probability based on the difference between the numbers of neighbors in the majority group and in the minority group. Specifically, for any node $v_i \in \mathcal{V}$, we define the balance score as

$$\text{balance}_i = \frac{1}{|\widetilde{N}_0(v_i) - \widetilde{N}_1(v_i)| + \delta} \tag{10}$$

where δ is a hyperparameter to avoid division by zero and $\widetilde{N}_0(v_i)$ as well as $\widetilde{N}_1(v_i)$ are the numbers of all neighbors within L hops that belong to the minority group and majority group, respectively. Then in the balance-aware sampling, for any node $v_i \in \mathcal{V}$, the sampling probability of node $v_j \in \mathcal{N}(v_i)$ is

$$P(v_j|v_i) = \frac{\text{balance}_j}{\sum_{v_k \in \mathcal{N}(v_i)} \text{balance}_k} \tag{11}$$

In general, BeMap includes three key steps: pre-processing, balance-aware sampling, and fair message passing. First, during pre-processing, we precompute the sampling probability based on Eq. (11). Then, during the every epoch of training, we first generate the fair neighborhood using the balance-aware sampling, and then aggregate neighbors' information on the generated fair subgraph, which is called fair message passing. Finally, we update the model parameters with back-propagation. The detailed workflow and the extension to non-binary sensitive attribute of BeMap is presented in Appendix E and Appendix F, respectively.

6 Experiments

6.1 Experimental Settings

Datasets. We conduct experiments on 4 real-world datasets, including *Pokec-z* [40], *Pokec-n* [40], *NBA* [9], *Credit* [11], and *Recidivism* [28]. For each dataset, we use the same 50%/25%/25% splits for train/validation/test sets. Detailed descriptions of the datasets are provided in Appendix G.1.

Baseline Methods. We compare BeMap with graph neural networks with and without fairness considerations. Graph neural networks without fairness considerations include GCN [1] and GraphSAGE [2]. Fair graph neural networks for comparison include *FairGNN* [9], *EDITS* [11], *FairDrop* [13], *NIFTY* [28] and *FMP* [12]. Descriptions of baseline methods are in Appendix G.2. For BeMap, BeMap (sym) uses GCN with row normalization, BeMap (sym) uses GCN with symmetric normalization, and BeMap (gat) uses GAT.

Parameter Settings. Unless otherwise specified, we use default hyperparameter settings in the released code of corresponding publications. For BeMap, we set the learning rate as $1e-3$, weight decay as $1e-5$, β as $\frac{1}{4}$, and δ as 1. We set the backbone model of BeMap as a 2-layer GCN with 128 hidden dimensions and the optimizer as Adam. To be consistent with the number of hidden layers, for each node, all neighbors within 2 hops are used to calculate the balance score in Eq. (10).

Metrics. We consider the task of semi-supervised node classification. To measure the utility, we use the classification accuracy (ACC) and the area under the receiver operating characteristic curve (AUC). Regarding fairness, we use Δ_{SP} and Δ_{EO} mentioned in Section 3. For ACC and AUC, the higher the better; while for Δ_{SP} and Δ_{EO}, the lower the better.

6.2 Experimental Results

Main Results. The main evaluation results on the utility (ACC and AUC) and fairness (Δ_{EO} and Δ_{SP}) are presented in Table 1. Similar evaluation results on Pokec-n are presented in Table 6. Regarding

Table 1: Main results on semi-supervised node classification. Higher is better (↑) for ACC and AUC (white). Lower is better (↓) for Δ_{SP} and Δ_{EO} (gray). Bold font indicates the best performance for fair graph neural networks, and underlined number indicates the second best.

Methods	Pokec-z				NBA			
	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓
GCN	70.62 ± 0.22	76.41 ± 0.61	8.86 ± 2.32	7.81 ± 1.99	72.16 ± 0.46	78.45 ± 0.25	4.00 ± 2.15	13.07 ± 3.34
GraphSAGE	70.27 ± 0.32	75.73 ± 0.30	7.11 ± 2.69	6.97 ± 2.65	75.21 ± 0.86	77.83 ± 2.00	7.81 ± 3.45	8.87 ± 5.58
GAT	64.24 ± 0.57	69.48 ± 0.53	11.56 ± 0.94	12.40 ± 1.85	72.14 ± 2.34	76.94 ± 0.82	5.97 ± 2.89	12.07 ± 2.47
FairGNN	65.74 ± 2.49	72.54 ± 4.21	4.31 ± 0.80	4.34 ± 1.22	$\mathbf{72.39 \pm 0.46}$	77.74 ± 1.24	3.96 ± 1.81	4.94 ± 2.35
EDITS	67.25 ± 0.61	73.05 ± 0.57	10.36 ± 1.20	9.00 ± 1.34	66.19 ± 0.75	69.94 ± 0.72	18.15 ± 3.64	13.19 ± 3.48
FairDrop	67.45 ± 0.80	73.77 ± 0.50	9.46 ± 2.06	7.91 ± 1.59	70.81 ± 0.63	77.05 ± 0.46	5.42 ± 1.59	7.30 ± 1.34
NIFTY	67.55 ± 0.79	73.87 ± 0.23	8.83 ± 1.60	7.00 ± 1.70	60.84 ± 3.32	65.94 ± 1.30	10.03 ± 5.46	5.70 ± 3.21
FMP	$\mathbf{72.05 \pm 0.42}$	$\mathbf{80.50 \pm 0.11}$	5.03 ± 1.22	1.72 ± 0.64	67.57 ± 1.58	77.73 ± 0.35	31.41 ± 1.11	27.17 ± 3.19
BeMap (row)	68.88 ± 0.30	72.34 ± 0.44	$\mathbf{0.74 \pm 0.52}$	1.55 ± 0.25	67.84 ± 0.64	$\mathbf{78.87 \pm 0.17}$	3.91 ± 1.28	4.08 ± 2.15
BeMap (sym)	68.94 ± 0.46	73.01 ± 0.29	1.45 ± 0.40	$\mathbf{1.03 \pm 0.42}$	65.37 ± 1.77	78.76 ± 0.62	$\mathbf{3.54 \pm 0.97}$	$\mathbf{3.81 \pm 0.98}$
BeMap (gat)	66.89 ± 1.26	71.42 ± 0.96	0.97 ± 0.22	1.29 ± 0.78	71.99 ± 1.10	77.36 ± 0.01	4.01 ± 3.26	5.08 ± 2.8

Methods	Recidivism				Credit			
	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓
GCN	85.89 ± 0.18	88.74 ± 0.23	8.51 ± 0.48	5.75 ± 1.08	75.80 ± 0.92	73.83 ± 1.50	17.92 ± 0.50	15.41 ± 0.77
GraphSAGE	85.00 ± 0.52	89.41 ± 0.36	8.99 ± 0.37	6.14 ± 0.54	74.25 ± 0.25	74.35 ± 0.12	13.82 ± 0.84	11.62 ± 0.87
GAT	88.66 ± 1.08	92.31 ± 0.62	7.45 ± 0.48	4.99 ± 0.67	70.61 ± 2.30	73.78 ± 0.35	10.85 ± 0.62	8.84 ± 0.56
FairGNN	69.54 ± 5.70	80.79 ± 5.33	6.61 ± 2.29	4.75 ± 3.50	75.34 ± 1.18	70.79 ± 2.76	11.23 ± 4.69	8.95 ± 2.76
EDITS	83.88 ± 0.84	86.82 ± 0.40	7.63 ± 0.57	5.06 ± 0.44	73.49 ± 0.03	$\mathbf{73.52 \pm 0.10}$	13.33 ± 0.15	10.93 ± 0.06
FairDrop	$\mathbf{91.81 \pm 0.36}$	$\mathbf{92.17 \pm 0.86}$	6.80 ± 0.22	3.30 ± 0.13	68.41 ± 9.20	70.76 ± 4.11	14.23 ± 2.24	12.01 ± 2.14
NIFTY	83.43 ± 0.83	84.56 ± 0.33	4.75 ± 0.92	4.04 ± 1.46	73.48 ± 0.04	72.33 ± 0.01	11.80 ± 0.09	9.51 ± 0.08
FMP	56.80 ± 10.83	61.79 ± 4.87	22.43 ± 9.72	17.50 ± 8.87	74.37 ± 0.21	72.92 ± 0.14	14.07 ± 0.78	11.87 ± 0.72
BeMap (row)	77.61 ± 0.33	81.11 ± 1.44	2.87 ± 0.46	$\mathbf{2.09 \pm 0.38}$	71.46 ± 0.74	69.41 ± 0.71	10.51 ± 0.23	10.87 ± 1.55
BeMap (sym)	77.66 ± 0.35	80.77 ± 1.02	$\mathbf{1.76 \pm 0.14}$	2.68 ± 0.12	72.55 ± 0.28	72.98 ± 0.66	10.74 ± 0.46	8.36 ± 0.46
BeMap (gat)	71.83 ± 1.23	71.97 ± 1.28	3.67 ± 1.22	3.60 ± 1.78	$\mathbf{76.72 \pm 0.77}$	68.18 ± 0.50	$\mathbf{7.92 \pm 1.03}$	$\mathbf{6.22 \pm 0.57}$

Table 2: Ablation study of different sampling strategy. Higher is better (↑) for ACC and AUC (white). Lower is better (↓) for Δ_{SP} and Δ_{EO} (gray). Bold font indicates the best performance for fair graph neural networks, and underlined number indicates the second best.

Sampling Methods	Pokec-z				NBA			
	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓
Uniform	69.65 ± 0.23	73.50 ± 0.24	4.98 ± 0.78	2.87 ± 1.19	$\mathbf{72.67 \pm 1.58}$	78.48 ± 0.42	3.70 ± 1.18	12.02 ± 2.97
Degree	$\mathbf{70.00 \pm 0.36}$	$\mathbf{73.98 \pm 0.37}$	5.85 ± 1.46	3.40 ± 1.84	71.64 ± 1.41	78.37 ± 0.32	4.25 ± 1.60	14.88 ± 4.60
Balance-aware	68.94 ± 0.46	73.01 ± 0.29	$\mathbf{1.45 \pm 0.40}$	$\mathbf{1.03 \pm 0.42}$	65.37 ± 1.77	$\mathbf{78.76 \pm 0.62}$	$\mathbf{3.54 \pm 0.97}$	$\mathbf{3.81 \pm 0.98}$

Sampling Methods	Recidivism				Credit			
	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓	ACC(%) ↑	AUC(%) ↑	Δ_{SP}(%) ↓	Δ_{EO}(%) ↓
Uniform	81.80 ± 0.93	76.58 ± 0.31	4.40 ± 0.15	3.29 ± 0.45	75.04 ± 0.40	$\mathbf{73.66 \pm 0.14}$	14.34 ± 0.64	11.70 ± 0.54
Degree	$\mathbf{85.06 \pm 1.16}$	$\mathbf{87.12 \pm 0.52}$	6.13 ± 0.13	2.78 ± 0.90	$\mathbf{75.82 \pm 0.45}$	73.64 ± 0.06	15.45 ± 0.82	12.59 ± 0.84
Balance-aware	77.66 ± 0.35	80.77 ± 1.02	$\mathbf{1.76 \pm 0.14}$	$\mathbf{2.68 \pm 0.12}$	72.55 ± 0.28	72.98 ± 0.66	$\mathbf{10.74 \pm 0.46}$	$\mathbf{8.36 \pm 0.46}$

fairness, our proposed BeMap is the only method that can consistently mitigate bias (i.e., a smaller value of Δ_{EO} and Δ_{SP} than the vanilla GCN and GAT) for all datasets. Moreover, compared with the vanilla GCN and GAT, all the variants of BeMap could effectively reduce Δ_{EO} and Δ_{SP} to a low degree. For example, on the Pokec-z dataset, $\Delta_{\text{EO}}(\Delta_{\text{SP}})$ are reduced to 8.35% (19.84%), 24.75% (14.77%) and 8.39% (10.40%) of original values for BeMap (row), BeMap (sym), and BeMap (gat), respectively. More detailed statistics are exhibited in Table 5. At the same time, BeMap also achieves comparable performance in terms of the utility (ACC and AUC). This is because BeMap generates a new balanced graph for each epoch, analogous to data augmentation, which prevents the graph neural network from overfitting. For example, on NBA, the AUC scores of BeMap (sym) and BeMap (row) are 78.87% and 78.76%, respectively, both of which are higher than the AUC of vanilla GCN and GraphSAGE (78.45% and 77.83%, respectively). In short words, BeMap could achieve a good balance between mitigating the bias and maintaining the classification accuracy.

Ablation Study. To evaluate the effectiveness of the balance-aware sampling, we compare it with two other heuristic sampling strategies: (1) *uniform sampling* (Uniform), which assigns the same probability to all neighbors of a node, and (2) *degree-based sampling* (Degree), which sets the probability of node v_i in the neighborhood of node u as $P(v_i \mid u) \propto d_i^{0.75}, \forall v_i \in \mathcal{N}(u)$ [41]. From Table 2, we can see that the balance-aware sampling achieves the lowest Δ_{EO} and Δ_{SP} on all datasets and maintains a comparable classification accuracy, which demonstrates the superiority of the balance-aware sampling in balancing the utility and fairness.

7 Conclusion

In this paper, we study bias amplification in message passing and fair message passing. We empirically and theoretically prove that message passing amplifies the bias as long as the numbers of neighbors from different demographic groups for each node are unbalanced. Guided by our analyses, we propose BeMap, which relies on a balance-aware sampling strategy to generate a fair neighborhood among different demographic groups. Then, BeMap performs message passing over the generated fair neighborhood. Extensive evaluations on the real-world datasets demonstrate the effectiveness of our proposed method in mitigating bias while maintaining utility.

Acknowledgments

This work is supported by NSF (1947135, 2134079, 1939725, 2316233, and 2324770), DARPA (HR001121C0165), DHS (17STQAC00001-07-00), NIFA (2020-67021-32799) and ARO (W911NF2110088).

References

[1] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 1, 2, 3, 8, 23

[2] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 2, 8, 23

[3] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018. 2

[4] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019. 1, 5

[5] Latanya Sweeney. Discrimination in online ad delivery. *Communications of the ACM*, 56(5): 44–54, 2013. 1

[6] Anja Lambrecht and Catherine Tucker. Algorithmic bias? an empirical study of apparent gender-based discrimination in the display of stem career ads. *Management science*, 65(7): 2966–2981, 2019. 1

[7] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021. 1

[8] Avishek Bose and William Hamilton. Compositional fairness constraints for graph embeddings. In *International Conference on Machine Learning*, pages 715–724. PMLR, 2019. 1, 2, 3

[9] 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. 2, 3, 4, 8, 22, 23

[10] Peizhao Li, Yifei Wang, Han Zhao, Pengyu Hong, and Hongfu Liu. On dyadic fairness: Exploring and mitigating bias in graph connections. In *International Conference on Learning Representations*, 2021. 24

[11] Yushun Dong, Ninghao Liu, Brian Jalaian, and Jundong Li. Edits: Modeling and mitigating data bias for graph neural networks. In *Proceedings of the ACM Web Conference 2022*, pages 1259–1269, 2022. 2, 3, 8, 22, 23

[12] Zhimeng Jiang, Xiaotian Han, Chao Fan, Zirui Liu, Na Zou, Ali Mostafavi, and Xia Hu. Fmp: Toward fair graph message passing against topology bias. *arXiv preprint arXiv:2202.04187*, 2022. 1, 2, 5, 8, 23

[13] Indro Spinelli, Simone Scardapane, Amir Hussain, and Aurelio Uncini. Biased edge dropout for enhancing fairness in graph representation learning. *arXiv preprint arXiv:2104.14210*, 2021. 1, 2, 8, 23

[14] April Chen, Ryan Rossi, Nedim Lipka, Jane Hoffswell, Gromit Chan, Shunan Guo, Eunyee Koh, Sungchul Kim, and Nesreen K Ahmed. Graph learning with localized neighborhood fairness. *arXiv preprint arXiv:2212.12040*, 2022. 1, 2

[15] Huaisheng Zhu, Guoji Fu, Zhimeng Guo, Zhiwei Zhang, Teng Xiao, and Suhang Wang. Fairness-aware message passing for graph neural networks. *arXiv preprint arXiv:2306.11132*, 2023. 1, 24

[16] Kaize Ding, Qinghai Zhou, Hanghang Tong, and Huan Liu. Few-shot network anomaly detection via cross-network meta-learning. In *Proceedings of the Web Conference 2021*, pages 2448–2456, 2021. 2

[17] Harini Suresh and John V Guttag. A framework for understanding unintended consequences of machine learning. *arXiv preprint arXiv:1901.10002*, 2:8, 2019. 2

[18] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The world wide web conference*, pages 417–426, 2019. 2

[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 2

[20] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. Graph random neural networks for semi-supervised learning on graphs. *Advances in neural information processing systems*, 33:22092–22103, 2020. 2

[21] Pál András Papp, Karolis Martinkus, Lukas Faber, and Roger Wattenhofer. Dropgnn: random dropouts increase the expressiveness of graph neural networks. *Advances in Neural Information Processing Systems*, 34:21997–22009, 2021. 2

[22] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019. 2

[23] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019. 2

[24] Bingzhe Wu, Jintang Li, Chengbin Hou, Guoji Fu, Yatao Bian, Liang Chen, and Junzhou Huang. Recent advances in reliable deep graph learning: Adversarial attack, inherent noise, and distribution shift. *arXiv preprint arXiv:2202.07114*, 2022.

[25] Jiaqi Han, Yu Rong, Tingyang Xu, and Wenbing Huang. Geometrically equivariant graph neural networks: A survey. *arXiv preprint arXiv:2202.07230*, 2022.

[26] Jun Xia, Yanqiao Zhu, Yuanqi Du, and Stan Z Li. A survey of pretraining on graphs: Taxonomy, methods, and applications. *arXiv preprint arXiv:2202.07893*, 2022.

[27] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. Data augmentation for deep graph learning: A survey. *arXiv preprint arXiv:2202.08235*, 2022. 2

[28] 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. 2, 8, 22, 23

[29] 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. 2

[30] Jian Kang, Jingrui He, Ross Maciejewski, and Hanghang Tong. Inform: Individual fairness on graph mining. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 379–389, 2020. 2

[31] Yushun Dong, Jian Kang, Hanghang Tong, and Jundong Li. Individual fairness for graph neural networks: A ranking based approach. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 300–310, 2021. 2

[32] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016. 2

[33] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012. 3

[34] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016. 3

[35] Keyulu Xu, Mozhi Zhang, Stefanie Jegelka, and Kenji Kawaguchi. Optimization of graph neural networks: Implicit acceleration by skip connections and more depth. In *International Conference on Machine Learning*, pages 11592–11602. PMLR, 2021. 5

[36] Rongzhe Wei, Haoteng Yin, Junteng Jia, Austin R Benson, and Pan Li. Understanding non-linearity in graph neural networks from the bayesian-inference perspective. *arXiv preprint arXiv:2207.11311*, 2022. 5

[37] Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. Convergence and stability of graph convolutional networks on large random graphs. *Advances in Neural Information Processing Systems*, 33:21512–21523, 2020. 5

[38] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. Ood-gnn: Out-of-distribution generalized graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 35(7): 7328–7340, 2023. doi: 10.1109/TKDE.2022.3193725. 5

[39] Yang Zhou, Hong Cheng, and Jeffrey Xu Yu. Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment*, 2(1):718–729, 2009. 5

[40] Thin Nguyen, Hang Le, Thomas P Quinn, Tri Nguyen, Thuc Duy Le, and Svetha Venkatesh. Graphdta: Predicting drug–target binding affinity with graph neural networks. *Bioinformatics*, 37(8):1140–1147, 2021. 8, 22

[41] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015. 9

[42] Arjun Subramonian, Kai-Wei Chang, and Yizhou Sun. On the discrimination risk of mean aggregation feature imputation in graphs. *Advances in Neural Information Processing Systems*, 35:32957–32973, 2022. 24

[43] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 833–841, 2021. 24, 25

[44] Joonhyung Park, Jaeyun Song, and Eunho Yang. Graphens: Neighbor-aware ego network synthesis for class-imbalanced node classification. In *The Tenth International Conference on Learning Representations, ICLR 2022*. International Conference on Learning Representations (ICLR), 2022. 24

[45] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. Imgagn: Imbalanced network embedding via generative adversarial graph networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1390–1398, 2021. 25

[46] Deli Chen, Yankai Lin, Guangxiang Zhao, Xuancheng Ren, Peng Li, Jie Zhou, and Xu Sun. Topology-imbalance learning for semi-supervised node classification. *Advances in Neural Information Processing Systems*, 34:29885–29897, 2021. 24, 25

[47] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 25

[48] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002. 25

[49] Yihong Ma, Yijun Tian, Nuno Moniz, and Nitesh V Chawla. Class-imbalanced learning on graphs: A survey. *arXiv preprint arXiv:2304.04300*, 2023. 25

[50] Yao Ma, Suhang Wang, Tyler Derr, Lingfei Wu, and Jiliang Tang. Attacking graph convolutional networks via rewiring. *arXiv preprint arXiv:1906.03750*, 2019. 25

[51] Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. Make heterophily graphs better fit gnn: A graph rewiring approach. *arXiv preprint arXiv:2209.08264*, 2022. 25

[52] Adrián Arnaiz-Rodríguez, Ahmed Begga, Francisco Escolano, and Nuria Oliver. Diffwire: Inductive graph rewiring via the lov\'asz bound. *arXiv preprint arXiv:2206.07369*, 2022. 25

[53] Hau Chan and Leman Akoglu. Optimizing network robustness by edge rewiring: a general framework. *Data Mining and Knowledge Discovery*, 30:1395–1425, 2016. 25

[54] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019. 25

[55] Elli Voudigari, Nikos Salamanos, Theodore Papageorgiou, and Emmanuel J. Yannakoudakis. Rank degree: An efficient algorithm for graph sampling. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 120–129, 2016. doi: 10.1109/ASONAM.2016.7752223. 25

[56] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31.

Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/01eee509ee2f68dc6014898c309e86bf-Paper.pdf. 25

A Key Symbols of BeMap

Table 3: Table of key symbols in the paper.

Symbol	Definition
$\mathcal{V}$	The set of nodes
$\mathcal{V}_s$	The set of nodes with the sensitive attribute of s
$\mathcal{N}$	The set of 1-hop neighbors
$\widehat{\mathcal{N}}$	The set of 1-hop neighbors and the node itself
$\mathcal{N}^k$	The set of k-hop neighbors
$\mathcal{N}_s$	The set of 1-hop neighbors with the sensitive attribute of s
$\mathbf{A}$	The adjacency matrix
$\tilde{\mathbf{A}}$	The row-normalized adjacency matrix
$\mathbf{X}$	The node feature matrix
$\mathbf{H}^{(k+1)}$	The node representation matrix of the k-th hidden layer
$\mathbf{D}$	The degree matrix of nodes
$\mathbf{W}^{(k)}$	The weight matrix of the k-th hidden layer
$\mathbf{T}$	The fair embedding matrix
$\mathbf{B}$	The bias residual matrix
$\mathbf{x}_i$	The node feature of the node i
$\mathbf{h}_i^{(k)}$	The node representation of the node i in the k-th hidden layer
$\mathbf{t}_i^{(k)}$	The fair embedding of the node i in the k-th hidden layer
$\mathbf{b}_i^{(k)}$	The bias residual of the node i in the k-th hidden layer
$\mathbf{b}_{i,s}^{(k)}$	The bias residual of the node i with the sensitive attribute of s in the k-th hidden layer
μ	The centroid of all the nodes
μ_s	The centroid of the nodes with the sensitive attribute of s
d_i	The degree of the node i
r_s	The ratio of the neighbors with the sensitive attribute of s in BeMap

B Proof of Lemma 1

Given a L-layer linear GCN, the weight matrix in the l-th hidden layer is represented as $\mathbf{W}^{(l)}$, and the input node features and output node features for all nodes are denoted as $\mathbf{H}^{(l)}$ and $\mathbf{H}^{(l+1)}$. Then the output node features can be calculated as $\mathbf{H}^{(l+1)} = \widetilde{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}$, where $\widetilde{\mathbf{A}}$ is the normalized adjacency matrix. Therefore, the output node feature of the last hidden layer can be expressed as

$$\mathbf{Z} = \widetilde{\mathbf{A}}^L\mathbf{H}^{(1)}\mathbf{W} \tag{12}$$

where $\mathbf{W} = \mathbf{W}^{(1)}\mathbf{W}^{(2)}\dots\mathbf{W}^{(L)}$. Then given a particular output node feature $\mathbf{Z}$, if there exists a solution, then a feasible solution of the input node feature is

$$\mathbf{H}^{(1)} = (\widetilde{\mathbf{A}}^L)^\dagger\mathbf{Z}\mathbf{W}^\dagger \tag{13}$$

where $\mathbf{W}^\dagger$ and $(\widetilde{\mathbf{A}}^L)^\dagger$ are the Moore–Penrose inverse of $\mathbf{W}$ and $\widetilde{\mathbf{A}}^L$, respectively.

Based on Assumption 2, the input node feature can be further expressed as

$$\mathbf{H}^{(1)} = (\widetilde{\mathbf{A}}^L)^\dagger\mathbf{Z}_t\mathbf{W}^\dagger + (\widetilde{\mathbf{A}}^L)^\dagger\mathbf{Z}_b\mathbf{W}^\dagger = \mathbf{T}^{(1)} + \mathbf{B}^{(1)} \tag{14}$$

where $\mathbf{T}^{(1)} = (\widetilde{\mathbf{A}}^L)^\dagger\mathbf{Z}_t\mathbf{W}^\dagger$ is the fair part of input node features while $\mathbf{B}^{(1)} = (\widetilde{\mathbf{A}}^L)^\dagger\mathbf{Z}_b\mathbf{W}^\dagger$ is the bias residual which lead to discrimination towards different demographic groups in predictions.

Therefore, the input node feature could be linearly separated into two vectors: the fair embeddings $\mathbf{T}^{(1)}$ and the bias residual $\mathbf{B}^{(1)}$.

Then for any l-th hidden layer $\in \{1, \ldots, L\}$, the following expression holds.

$$\begin{aligned}
\mathbf{H}^{(l)} &= \widetilde{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l-1)} \\
&= \widetilde{\mathbf{A}}^{l-1}\mathbf{H}^{(1)}\mathbf{W}^{(1)} \ldots \mathbf{W}^{(l-1)} \\
&= \widetilde{\mathbf{A}}^{l-1}\left(\mathbf{T}^{(1)} + \mathbf{B}^{(1)}\right)\mathbf{W}^{(1)} \ldots \mathbf{W}^{(l-1)} \\
&= \left(\widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^{L})^{\dagger}\mathbf{Z}_t\mathbf{W}^{\dagger} + \widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^{L})^{\dagger}\mathbf{Z}_b\mathbf{W}^{\dagger}\right)\mathbf{W}^{(1)} \ldots \mathbf{W}^{(l-1)} \\
&= \left(\mathbf{T}^{(l)} + \mathbf{B}^{(l)}\right)\mathbf{W}^{(1)} \ldots \mathbf{W}^{(l-1)}
\end{aligned} \tag{15}$$

where $\mathbf{T}^{(l)} = \widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^{L})^{\dagger}\mathbf{Z}_t\mathbf{W}^{\dagger}$ and $\mathbf{B}^{(l)} = \widetilde{\mathbf{A}}^{l-1}(\widetilde{\mathbf{A}}^{L})^{\dagger}\mathbf{Z}_b\mathbf{W}^{\dagger}$. Written in vector form, we naturally have the following expression

$$\mathbf{h}_i^{(l)} = \left(\mathbf{t}_i^{(l)} + \mathbf{b}_i^{(l)}\right)\mathbf{W}^{(1)} \ldots \mathbf{W}^{(l-1)} \tag{16}$$

which completes the proof.

C Analysis on Linear GCN

In this section, we first prove that the centroid for each demographic group remains unchanged after message passing. Then we prove Theorem 1 (i.e., distance shrinkage).

To complete both proofs, we present the following two propositions. The first proposition focuses on two key properties of Gilbert random graphs (Assumption 1).

Proposition 1. *Given a Gilbert random graph $\mathcal{G}$, we have the following key properties.*

- ***Independence between nodes.*** *For any node v_i in $\mathcal{G}$, its node features and the node features of any of its 1-hop neighbor $v_j \in \mathcal{N}(v_i)$ are independently and identically distributed, where $\mathcal{N}(\cdot)$ represents the set of 1-hop neighbors.*
- ***Independence between node features and topology.*** *The probability distribution of node features are independent to topology structure of the graph $\mathcal{G}$, i.e., the input node features $\mathbf{X}$ is independent with the adjacency matrix $\mathbf{A}$.*

Proof. First, we first prove the **independence between nodes**. Note that in a Gilbert random graph, an edge is randomly added with a fixed probability. This process is equivalent to randomly selecting one node from the graph as the starting point and then randomly selecting another node as the endpoint. Consequently, two nodes from the same edge are obtained through simple random sampling from the given probability distribution. Therefore, node features for any node and its neighbors are independent and identically distributed.

Second, since the edge generation of a Gilbert random graph is independent to node features, the probability distribution of node features are naturally independent to the topology structure. □

For Gilbert random graphs, since node features are independent to topology, the bias residual matrix $\mathbf{B}$ is also independent to the adjacency matrix $\mathbf{A}$. Actually, a prerequisite for this independence is that, for any node v_i, the information from its neighbors does not affect the mean and variance of the biased vector distribution for the node v_i. Therefore, we propose proposition 2 and give the corresponding proof.

Proposition 2. *Given a graph $\mathcal{G}$, if bias residual matrix is independent to the adjacency matrix, then the following expressions hold true:*

$$\mathbb{E}_{v_i \in \mathcal{V}}\left[\mathbb{E}_{v_j \in \mathcal{N}(v_i)}\left[\mathbf{b}_j\right]\right] = \mathbb{E}_{v_j \in \mathcal{V}}[\mathbf{b}_j] \tag{17a}$$

$$\mathbb{E}_{v_i \in \mathcal{V}}\left[\mathbb{E}_{v_j \in \mathcal{N}(v_i)}\left[(\mathbf{b}_j - \mathbb{E}[\mathbf{b}_j])^2\right]\right] = \mathbb{E}_{v_j \in \mathcal{V}}[(\mathbf{b}_j - \mathbb{E}[\mathbf{b}_j])^2] \tag{17b}$$

where $\mathcal{N}(v_i)$ means the set of the neighbors of the node v_i.

Proof. We will only prove Eq. (17a) here, and the proof of Eq. (17b) is exactly the same.

Actually, proving proposition 2 is equivalent to proving its contrapositive. Therefore, we only need to prove that given a graph $\mathcal{G}$, if the following expression hold true:

$$\mathbb{E}_{v_i \in \mathcal{V}} \left[\mathbb{E}_{v_j \in \mathcal{N}(v_i)} \left[\mathbf{b}_j \right] \right] \neq \mathbb{E}_{v_j \in \mathcal{V}} \left[\mathbf{b}_j \right] \tag{18}$$

then the bias residual matrix is not independent to the adjacency matrix.

Imagine the situation that the bias residual matrix is independent to adjacency matrix. It means that we cannot obtain any information of the adjacency matrix even if we have the bias residual matrix. Then given the bias residual matrix, the probability of correctly predicting the adjacency matrix should be the same as random guessing. Considering a graph with N nodes, the adjacency matrix contains a total of N^2 elements. With each element having two possible values, 0 or 1, the probability of random guessing correctly is given by $Pr\left(\text{random}\right) = \frac{1}{2^{N^2}}$.

However, we can readily identify two specific types of graph topology structure that fail to meet the requirement of Eq. (18). Then we can prove that if Eq. (18) holds true, the possibility of predicting the adjacency matrix is larger than random guessing. Here are the two types of graph structures.

The first type is a graph with only self-loop edges. In such a graph, the adjacency matrix is an identity matrix. For this particular graph topology, regardless of the distribution of the bias matrix, Eq. (17a) always holds for this particular graph topology.

Another type of graph is a cyclic graph. Each node in the graph is assigned a unique sequential number from 1 to N, and the nodes are sorted accordingly. We denote the node with number k as v_k. Node v_k is exclusively connected to nodes v_{k-1} and v_{k+1}. The adjacency matrix for this type of graph topology is shown below. For this graph structure, we can mathematically prove that the left-hand side of Eq. (17a) represents the expectation obtained by summing the values twice for each node and taking the average subsequently, which is equal to the right-hand side of Eq. (17a).

$$\begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 1 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 0 & 1 & 0 & 1 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

Hence, if Eq. (18) holds, the adjacency matrices of the aforementioned two types of graphs do not satisfy the requirement. As a result, the number of possible adjacency matrices will be less than or equal to $\left(2^{N^2} - 2\right)$. Therefore, the probability of correctly predicting the adjacency matrix, $Pr\left(\text{correct}\right)$, satisfies $Pr\left(\text{correct}\right) \geq \frac{1}{2^{N^2}-2} > Pr\left(\text{random}\right) = \frac{1}{2^{N^2}}$. Thus, the contrapositive is valid because the adjacency matrix is not independent to the bias input matrix, which completes the proof of Proposition 2. □

With the above two propositions hold, we will further prove the stable centroid after message passing (Appendix C.1) and distance shrinkage (Theorem 1, Appendix C.2).

C.1 Stable Centroid

Here, we prove that, for GCNs with row normalization, the distribution centroid keeps unchanged after message passing. Mathematically, our goal is to prove the following equation.

$$\mathbb{E}_{v_i \in \mathcal{V}} \left[\mathbf{b}_i^{(l+1)} \right] = \mathbb{E}_{v_j \in \mathcal{V}} \left[\mathbf{b}_j^{(l)} \right] \tag{19}$$

In the l-th iteration of message passing, the mean of the bias residuals can be calculated as

$$\begin{aligned}\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbf{b}_i^{(l+1)}\right] &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}(v_i)}\alpha_i\mathbf{b}_j^{(l)}\right] = \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\sum_{v_j\in\mathcal{N}(v_i)}\mathbf{b}_j^{(l)}\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\sum_{v_j\in\mathcal{N}(v_i)}\mathbb{E}_{v_j\in\mathcal{N}(v_i)}\left[\mathbf{b}_j^{(l)}\right]\right] = \mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)}\left[\mathbf{b}_j^{(l)}\right]\right] \\ &= \mathbb{E}_{v_j\in\mathcal{V}}\left[\mathbf{b}_j^{(l)}\right]\end{aligned} \tag{20}$$

which completes the proof.

C.2 Proof of Theorem 1 (Distance Shrinkage)

Let $\boldsymbol{\mu} = \mu(v_i)$ and $\hat{\mathbf{b}}_i^{(l+1)} = \mathbf{b}_i^{(l+1)} - \boldsymbol{\mu}$. We have

$$\begin{aligned}&\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l+1)} - \mu(v_i)\|_2^2] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\mathbf{b}_i^{(l+1)} - \boldsymbol{\mu}\right)^T\left(\mathbf{b}_i^{(l+1)} - \boldsymbol{\mu}\right)\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\sum_{v_j\in\mathcal{N}(v_i)}\alpha_i\mathbf{b}_j^{(l)} - \boldsymbol{\mu}\right)^T\left(\sum_{v_j\in\mathcal{N}(v_i)}\alpha_i\mathbf{b}_j^{(l)} - \boldsymbol{\mu}\right)\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\sum_{v_j\in\mathcal{N}(v_i)}\alpha_i\hat{\mathbf{b}}_j^{(l)}\right)^T\left(\sum_{v_j\in\mathcal{N}(v_i)}\alpha_i\hat{\mathbf{b}}_i^{(l)}\right)\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right] \\ &= \underbrace{\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_j^{(l)}\right]}_{①} + \underbrace{\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{v_j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]}_{②}\end{aligned} \tag{21}$$

For ①, we have

$$\begin{aligned}&\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_j^{(l)}\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\mathbb{E}_{v_j\in\mathcal{N}(v_i)}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_j^{(l)}\right]\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\mathbb{E}_{v_j\in\mathcal{N}(v_i)}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_j^{(l)}\right]\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\right]\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_j^{(l)}\right]\right] \\ &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{1}{d_i}\right]\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l)}\right)^T\hat{\mathbf{b}}_i^{(l)}\right]\end{aligned} \tag{22}$$

where d_i is the degree of the i-th node. For ②, we have

$$\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\frac{1}{|\mathcal{V}|}\sum_{v_i\in\mathcal{V}}\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\frac{1}{|\mathcal{V}|}\sum_{v_j\in\mathcal{V}}\sum_{v_k\in\mathcal{N}^2(v_j)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}
\end{aligned}\tag{23}$$

where $\mathcal{N}^2(v_j)$ is the set of the 2-hop neighbors of the node v_j. Since the bias residual matrix $\mathbf{B}$ is independent to the adjacency matrix $\mathbf{A}$, it is also independent to $\widetilde{\mathbf{A}}=\phi(\mathbf{A}^2)$, where $\phi(\cdot)$ is the function that sets the diagonal elements of a given matrix to 0. Let $\widetilde{\mathcal{N}}(v)$ represent the neighborhood of node v in the new adjacency matrix $\widetilde{\mathbf{A}}$. Please note that the new neighborhood $\widetilde{\mathcal{N}}(\cdot)$ is the 2-hop neighborhood excluding the node itself, *i.e.*, $\widetilde{\mathcal{N}}(v_i)=\mathcal{N}^2(v_i)\setminus\{i\}$. Then we have

$$\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\frac{1}{|\mathcal{V}|}\sum_{v_j\in\mathcal{V}}\sum_{v_k\in\mathcal{N}^2(v_j)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbb{E}_{v_j\in\mathcal{V}}\left[\sum_{v_k\in\widetilde{\mathcal{N}}(v_j)}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbb{E}_{v_j\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\sum_{v_k\in\widetilde{\mathcal{N}}(v_j)}\hat{\mathbf{b}}_k^{(l)}\right]
\end{aligned}\tag{24}$$

Since there is no self-loop in $\widetilde{\mathbf{A}}$, the bias residual $\hat{\mathbf{b}}_j$ is different from the bias residual vector $\hat{\mathbf{b}}_k$, hence independent to $\hat{\mathbf{b}}_k$. Therefore, we have

$$\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}(v_i)}\sum_{v_k\in\mathcal{N}(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\hat{\mathbf{b}}_k^{(l)}\right]\quad ②\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbb{E}_{v_j\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\sum_{v_k\in\widetilde{\mathcal{N}}(v_j)}\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbb{E}_{v_j\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_j^{(l)}\right)^T\right]\mathbb{E}_{v_j\in\mathcal{V}}\left[\sum_{v_k\in\widetilde{\mathcal{N}}(v_j)}\hat{\mathbf{b}}_k^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\mathbf{0}^T\mathbb{E}_{v_j\in\mathcal{V}}\left[\sum_{v_k\in\widetilde{\mathcal{N}}(v_j)}\hat{\mathbf{b}}_k^{(l)}\right]\\
&=0
\end{aligned}\tag{25}$$

Combining everything together, we have the following equation holds.

$$\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]=\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{1}{d_i}\right]\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l)}\right)^T\hat{\mathbf{b}}_i^{(l)}\right]\tag{26}$$

Based on Eq. (26), for every demographic group, the corresponding distance will shrink at the rate proportional to the reciprocal of node degree, which completes the proof of Theorem 1.

D Analysis on BeMap

In this section, we prove **centroid consistency** (Appendix D.1) and **distance shrinkage** (Appendix D.2). We first present a variant of Proposition 2 in Proposition 3.

Proposition 3. *Given a graph $\mathcal{G}$ and a specific demographic group $\mathcal{S}$, if bias residuals from the demographic group is independent to the adjacency matrix, then the following equations holds*

$$\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)\cap\mathcal{S}}\left[\mathbf{b}_j\right]\right]=\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right] \tag{27}$$

$$\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)\cap\mathcal{S}}\left[\left(\mathbf{b}_j-\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right]\right)^2\right]\right]=\mathbb{E}_{v_j\in\mathcal{S}}\left[\left(\mathbf{b}_j-\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right]\right)^2\right] \tag{28}$$

where $\mathcal{N}(v_i)$ means the set of the neighbors of the node v_i.

Proof. Similar to the proof of Proposition 2, we prove the contrapositive related to Eq. (27). Specifically, we want to prove that given a graph and a specific demographic group $\mathcal{S}$, if the following equation holds

$$\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)\cap\mathcal{S}}\left[\mathbf{b}_j\right]\right]\neq\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right] \tag{29}$$

the bias residual matrix from the demographic group $\mathcal{S}$ is not independent to the adjacency matrix. To begin with, same as Appendix C, when the bias residuals are independent to the adjacency matrix, then we can easily have the probability of correctly predicting the adjacency matrix $Pr(random)=\frac{1}{2^{N^2}}$ with N representing the number of nodes.

Then we will prove that if Eq. (29) is true, the probability of correctly predicting the adjacency matrix will be lager than that of random guessing. First, we give every node a unique index. For the nodes from the demographic group $\mathcal{S}$, we assign them consecutive numbers from 1 to $|\mathcal{S}|$, and assign the nodes that do not belong to the demographic group $\mathcal{S}$ the consecutive numbers from $|\mathcal{S}|+1$ to N. Then we sort all the nodes, and use v_k to represent the node with the index k.

Consider the given adjacency matrix $\mathbf{A}^*$ listed below. Specifically, the matrix $\mathbf{M}_1$ is a $|\mathcal{S}|\times|\mathcal{S}|$ matrix which can be any type of adjacency matrices described in Proof C, i.e., the identity matrix or the adjacency matrix of a cyclic graph. The matrix $\mathbf{M}_2$ is a $(N-|\mathcal{S}|)\times|\mathcal{S}|$ matrix with all the elements being 1. The matrix $\mathbf{M}_3$ can be any matrix whose shape is $(N-|\mathcal{S}|)\times(N-|\mathcal{S}|)$. We can naturally prove that the adjacency matrix $\mathbf{A}^*$ does not meet the requirement of Eq. (29). For any node v_k from the demographic group $\mathcal{S}$, the neighbor relationship between the node v_i and its neighbors $\mathcal{N}(v_i)\cap\mathcal{S}$ from the demographic group $\mathcal{S}$ can be depicted by the matrix $\mathbf{M}_1$. Then due to Proof C, the expression of $\mathbb{E}_{v_i\in\mathcal{S}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)\cap\mathcal{S}}\left[\mathbf{b}_j\right]\right]=\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right]$ holds true for the nodes from the demographic group $\mathcal{S}$. For the nodes v_i which are not from the demographic group $\mathcal{S}$, their neighbors $\mathcal{N}(v_i)\cap\mathcal{S}$ from the demographic group $\mathcal{S}$ are the demographic group $\mathcal{S}$ since any element in the matrix $\mathbf{M}_2$ is 1. Then naturally, the expression of $\mathbb{E}_{v_i\in\mathcal{V}-\mathcal{S}}\left[\mathbb{E}_{v_j\in\mathcal{N}(v_i)\cap\mathcal{S}}\left[\mathbf{b}_j\right]\right]=\mathbb{E}_{v_j\in\mathcal{S}}\left[\mathbf{b}_j\right]$ holds true. Therefore, the given adjacency matrix $\mathbf{A}^*$ cannot satisfy Eq. (29).

$$\mathbf{A}^*=\begin{bmatrix}\mathbf{M}_1 & \mathbf{M}_2^T\\ \mathbf{M}_2 & \mathbf{M}_3\end{bmatrix}$$

Since the matrix $\mathbf{M}_1$ has at least two possible solutions listed in Proof C and the matrix $\mathbf{M}_3$ has $2^{(N-|\mathcal{S}|)\times(N-|\mathcal{S}|)}$ possible solutions, the matrix $\mathbf{A}^*$ has at least $2^{(N-|\mathcal{S}|)\times(N-|\mathcal{S}|)+1}$ possible solutions. Therefore, the probability of correcting predicting the adjacency matrix satisfies $Pr\,(\text{correct})\geq\frac{1}{2^{N^2}-2^{(N-|\mathcal{S}|)\times(N-|\mathcal{S}|)+1}}>\frac{1}{2^{N^2}}=Pr\,(\text{random})$. Thus, the contrapositive is valid, which completes the proof. □

D.1 Centroid Consistency

First, in the following paper, the last subscript separated by the comma on the lower right corner are used to refer to the demographic group to which bias vectors belongs. For example, $\mathbf{b}_{i,s}$ represents the bias vector of the node v_i with the sensitive attribute of s. Then the mean of the biased vector $\mathbf{b}_i^{(l+1)}$ can be calculated as

$$
\begin{aligned}
\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbf{b}_i^{(l+1)}\right] &= \mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}\right] \\
&= \mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}\right] \\
&= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\sum_{v_j\in\mathcal{N}_0(v_i)}\mathbf{b}_{j,0}^{(l)}\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i\sum_{v_j\in\mathcal{N}_1(v_i)}\mathbf{b}_{j,1}^{(l)}\right] \\
&= \mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i|\mathcal{N}_0\left(v_i\right)|\mathbb{E}_{v_j\in\mathcal{N}_0(v_i)}\left[\mathbf{b}_{j,0}^{(l)}\right]\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i|\mathcal{N}_1\left(v_i\right)|\mathbb{E}_{v_j\in\mathcal{N}_1(v_i)}\left[\mathbf{b}_{j,1}^{(l)}\right]\right] \\
&= \mathbb{E}_{v_i\in\mathcal{V}}\left[r_0\mathbb{E}_{v_j\in\mathcal{N}_0(v_i)}\left[\mathbf{b}_{j,0}^{(l)}\right]\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[r_1\mathbb{E}_{v_j\in\mathcal{N}_1(v_i)}\left[\mathbf{b}_{j,1}^{(l)}\right]\right] \\
&= r_0\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\mathbf{b}_{j,0}^{(l)}\right]+r_1\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\mathbf{b}_{j,1}^{(l)}\right]
\end{aligned}
\tag{30}
$$

where $\mathcal{N}_s\left(v_i\right)$ represent the neighbors of the node v_i with the sensitive attribute s.

For non-binary sensitive attribute $s\in\{0,\ldots,S-1\}$, we can easily rewrite the above results as

$$
\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbf{b}_i^{(l+1)}\right]=\sum_{s=0}^{S-1}r_s\mathbb{E}_{v_j\in\mathcal{V}_s}\left[\mathbf{b}_{j,s}^{(l)}\right] \tag{31}
$$

Actually, Eq. (30) and Eq. (31) hold for any node regardless of its sensitive attribute. Therefore, we successfully prove **centroid consistency**.

D.2 Distance Shrinkage

We will calculate the distance $\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l+1)}-\bar{\mu}\|_2^2]=\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]$ here. First, we simplify the expression of the bias residual $\hat{\mathbf{b}}_i^{(l+1)}$

$$
\begin{aligned}
\hat{\mathbf{b}}_i^{(l+1)} &= \mathbf{b}_i^{(l+1)}-\mathbb{E}_{v_i\in\mathcal{V}}\left[\mathbf{b}_i^{(l+1)}\right] \\
&= \sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}-\mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}\right]-\mathbb{E}_{v_i\in\mathcal{V}}\left[\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}\right] \\
&= \sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}-\mathbb{E}_{v_i\in\mathcal{V}}\left[r_0\mathbb{E}_{v_j\in\mathcal{N}_0(v_i)}\left[\mathbf{b}_{j,0}^{(l)}\right]\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[r_1\mathbb{E}_{v_j\in\mathcal{N}_1(v_i)}\left[\mathbf{b}_{j,1}^{(l)}\right]\right] \\
&= \sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\mathbf{b}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\mathbf{b}_{j,1}^{(l)}-r_0\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\mathbf{b}_{j,0}^{(l)}\right]-r_1\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\mathbf{b}_{j,1}^{(l)}\right] \\
&= \sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\left(\mathbf{b}_{j,0}^{(l)}-\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\mathbf{b}_{j,0}^{(l)}\right]\right)+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\left(\mathbf{b}_{j,1}^{(l)}-\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\mathbf{b}_{j,1}^{(l)}\right]\right) \\
&= \sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\hat{\mathbf{b}}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\hat{\mathbf{b}}_{j,1}^{(l)}
\end{aligned}
\tag{32}
$$

where $\mathcal{N}_s\left(v_i\right)$ represent the set of neighbors with the sensitive attribute of s for the node v_i, $\mathcal{V}_s$ represent the set of all the nodes with the sensitive attribute of s, and $\hat{\mathbf{b}}_{j,s}^{(l)}=\mathbf{b}_{j,s}^{(l)}-\mathbb{E}_{v_j\in\mathcal{V}_s}\left[\mathbf{b}_{j,s}^{(l)}\right], s\in\{0,1\}$. Obviously, the mean of the distribution of $\hat{\mathbf{b}}_{j,s}^{(l)}$ is $\mathbf{0}$. Then the expectation of distance will be expressed as:

$$
\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\hat{\mathbf{b}}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\left(\sum_{v_j\in\mathcal{N}_0(v_i)}\alpha_i\hat{\mathbf{b}}_{j,0}^{(l)}+\sum_{v_j\in\mathcal{N}_1(v_i)}\alpha_i\hat{\mathbf{b}}_{j,1}^{(l)}\right)\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\sum_{v_k\in\mathcal{N}_0(v_i)}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{k,0}^{(l)}\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_1(v_i)}\sum_{v_k\in\mathcal{N}_1(v_i)}\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{k,1}^{(l)}\right]\\
&\quad+2\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\sum_{v_k\in\mathcal{N}_1(v_i)}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{k,1}^{(l)}\right]\\
&\le\underbrace{2\,\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\sum_{v_k\in\mathcal{N}_0(v_i)}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{k,0}^{(l)}\right]}_{①}+\underbrace{2\,\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_1(v_i)}\sum_{v_k\in\mathcal{N}_1(v_i)}\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{k,1}^{(l)}\right]}_{②}
\end{aligned}
\tag{33}
$$

The first item ① can be simplified as

$$
\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\sum_{v_k\in\mathcal{N}_0(v_i)}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{k,0}^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_0(v_i)}\sum_{v_k\in\mathcal{N}_0(v_i)\setminus\{j\}}\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{k,0}^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2d_i^{(0)}\mathbb{E}_{v_j\in\mathcal{N}_0(v_i)}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\frac{|\mathcal{V}_0|}{|\mathcal{V}|}\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\sum_{v_k\in\widetilde{\mathcal{N}}_0(v_j)}\hat{\mathbf{b}}_{k,0}^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2d_i^{(0)}\right]\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\right]\frac{|\mathcal{V}_0|}{|\mathcal{V}|}\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\right]\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\sum_{v_k\in\widetilde{\mathcal{N}}_0(v_j)}\hat{\mathbf{b}}_{k,0}^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[r_0\alpha_i\right]\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+0\\
&=r_0\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{1}{d_i}\right]\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]
\end{aligned}
\tag{34}
$$

where $d_i^{(s)}$ the the number of the node v_i's neighbors with the sensitive attribute of s, and $\widetilde{\mathcal{N}}_s\left(v_i\right)$ is the neighbors from $\widetilde{\mathcal{N}}\left(v_i\right)$ with the sensitive attribute of $s, s\in\{0,1\}$.

Similarly, the second item ② can be simplified as:

$$
\mathbb{E}_{v_i\in\mathcal{V}}\left[\alpha_i^2\sum_{v_j\in\mathcal{N}_1(v_i)}\sum_{v_k\in\mathcal{N}_1(v_i)}\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{k,1}^{(l)}\right]=r_1\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{1}{d_i}\right]\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{j,1}^{(l)}\right]
\tag{35}
$$

Combining everything together, we have the following relationship between the distances to centroids before and after message passing.

$$
\begin{aligned}
&\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]\\
&=r_0\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{2}{d_i}\right]\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+r_1\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{2}{d_i}\right]\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{j,1}^{(l)}\right]\\
&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{2}{d_i}\right]\left(r_0\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+r_1\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{j,1}^{(l)}\right]\right)
\end{aligned}
\tag{36}
$$

For the non-binary sensitive attribute $s\in\{0,1,\ldots,S-1\}$, similarly, we have

$$
\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]=\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{S}{d_i}\right]\sum_{s=0}^{S-1}r_s\mathbb{E}_{v_j\in\mathcal{V}_s}\left[(\hat{\mathbf{b}}_{j,s}^{(l)})^T\hat{\mathbf{b}}_{j,s}^{(l)}\right]
\tag{37}
$$

Let us revisit the discussion concerning binary sensitive attribute. Since the local neighborhood is large enough mentioned in Theorem 2, i.e., $d_i>2$ for any node v_i, we have

$$
\begin{aligned}
\mathbb{E}_{v_i}[\|\mathbf{b}_i^{(l+1)}-\bar{\mu}\|_2^2]&=\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l+1)}\right)^T\hat{\mathbf{b}}_i^{(l+1)}\right]\\
&\leq\mathbb{E}_{v_i\in\mathcal{V}}\left[\frac{2}{d_i}\right]\left(r_0\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+r_1\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{j,1}^{(l)}\right]\right)\\
&<\left(r_0\mathbb{E}_{v_j\in\mathcal{V}_0}\left[\left(\hat{\mathbf{b}}_{j,0}^{(l)}\right)^T\hat{\mathbf{b}}_{j,0}^{(l)}\right]+r_1\mathbb{E}_{v_j\in\mathcal{V}_1}\left[\left(\hat{\mathbf{b}}_{j,1}^{(l)}\right)^T\hat{\mathbf{b}}_{j,1}^{(l)}\right]\right)\\
&<\mathbb{E}_{v_i\in\mathcal{V}}\left[\left(\hat{\mathbf{b}}_i^{(l)}\right)^T\hat{\mathbf{b}}_i^{(l)}\right]\\
&<\mathbb{E}_{v_i}\left[\|\mathbf{b}_i^{(l)}-\bar{\mu}\|_2^2\right]
\end{aligned}
\tag{38}
$$

which completes the proof.

E Pseudocode of BeMap

The pseudocode of BeMap is presented in Algorithm 1. Before training, we precompute the sampling probability in the balance-aware sampling (lines 3 – 5). During each epoch, we first generate the fair neighborhood using the balance-aware sampling (lines 7 – 11). Then for each hidden layer, the fair node representation of each node is learned on the fair neighborhood (lines 12 – 15). Finally, we update the model parameters with back-propagation (lines 16 – 17).

F Extension of BeMapto Non-binary Sensitive Attribute

We consider a non-binary sensitive attribute s which forms n_s demographic groups, i.e., $s\in\{1,\ldots,n_s\}$. The key idea of BeMap is to balance the number of neighbors across different demographics. We discuss two cases for balancing the neighborhood in the following.

- *When all neighbors belong to one demographic group:* We adopt the same strategy as the case of binary sensitive attribute by sampling a subset of k neighbors for any node v_i such that $k=\max\{\beta|\mathcal{N}(v_i)|,4\}$.
- *When the neighbors belong to different demographic groups:* In this case, for any node v_i, we first count the number of neighbors in each demographic group $\left\{\widehat{\mathcal{N}}_s(v_i)\,|\forall s=1,\ldots,n_s\right\}$. Then we set the number of neighbors to be sampled for any demographic group k as the smallest non-zero value in $\left\{\widehat{\mathcal{N}}_s(v_i)\,|\forall s=1,\ldots,n_s\right\}$. After that, for each demographic group in the neighborhood of v_i, we sample k neighbors to create a balanced neighborhood.

Regarding the balance-aware sampling probability in the above sampling steps, we modify Eq. (10) by replacing the absolute difference $|\widetilde{N}_0(v_i)-\widetilde{N}_1(v_i)|$ in the cardinalities of two demographic groups

Algorithm 1: Training GCN with BeMap.

Input : An input graph $\mathcal{G} = \{\mathcal{V}, \mathbf{A}, \mathbf{X}\}$, a set of training nodes $\mathcal{V}_{\text{train}}$, ground-truth labels $\mathcal{Y}_{\text{train}}$, an L-layer GCN with weight matrices $\mathbf{\Theta} = \left\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L)}\right\}$, a task-specific loss function J, maximum number of epochs $\text{epoch}_{\max}$, hyperaprameters β, m, δ;

Output : A well trained GCN $\mathbf{\Theta} = \left\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L)}\right\}$.

Initialize $\mathbf{H}^{(1)} = \mathbf{X}$;
Initialize the gradient-based optimizer *OPT*;
for *each node* $v_i \in \mathcal{V}$ **do**
  for *each node* $v_j \in \mathcal{N}(v_i)$ **do**
    Precompute $P(v_j|v_i)$ by Eqs. (10) and (11);
for $\text{epoch} = 1 \to \text{epoch}_{\max}$ **do**
  // Fair neighborhood generation
  for *each node* $v_i \in \mathcal{V}$ **do**
    if $\forall v_j \in \widehat{\mathcal{N}}(v_i)$ *belongs to the same demographic group* **then**
      Sample the fair neighborhood $\widehat{\mathcal{N}}^{\text{fair}}(v_i) = \mathcal{N}^{\text{fair}}(v_i) \cup \{v_i\}$ with probability $P(v_j|v_i), \forall v_j \in \mathcal{N}(v_i)$ such that $|\mathcal{N}^{\text{fair}}(v_i)| = \max\{\beta|\mathcal{N}(v_i)|, m\}$;
    else
      Sample the neighborhood $\mathcal{N}^{\text{fair}}(v_i)$ with probability $P(v_j|v_i), \forall v_j \in \mathcal{N}(v_i)$ and generate the fair neighborhood $\widehat{\mathcal{N}}^{\text{fair}}(v_i) = \mathcal{N}^{\text{fair}}(v_i) \cup \{v_i\}$ such that $|\widehat{\mathcal{N}}_0^{\text{fair}}(v_i)| = |\widehat{\mathcal{N}}_1^{\text{fair}}(v_i)|$
  // Forward propagation
  for *each hidden layer* $l \in \{1, \ldots, L\}$ **do**
    for *each node* $v_i \in \mathcal{V}$ **do**
      $\widehat{\mathbf{h}}_i^{(l)} = \sum_{v_j \in \widehat{\mathcal{N}}(v_i)} \alpha_{ij} \mathbf{h}_j^{(l)}$ with $\alpha_{ij} = \frac{1}{|\widehat{\mathcal{N}}^{\text{fair}}(v_i)|}$ if row normalization else $\alpha_{ij} = \frac{1}{\sqrt{|\widehat{\mathcal{N}}^{\text{fair}}(v_i)|}\sqrt{|\widehat{\mathcal{N}}^{\text{fair}}(v_j)|}}$;
      $\mathbf{h}_i^{(l+1)} = \sigma\left(\widehat{\mathbf{h}}_i^{(l)} \mathbf{W}^{(l)}\right)$;
  // Backward propagation
  Calculate the empirical loss $\text{loss} = J\left(\mathcal{V}_{\text{train}}, \mathcal{Y}_{\text{train}}, \left\{\mathbf{h}_i^{(L+1)}, \forall v_i \in \mathcal{V}\right\}\right)$;
  Update $\mathbf{\Theta}$ by $OPT(\nabla \text{loss})$;
return $\mathbf{\Theta} = \left\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(L)}\right\}$;

in binary case to the average squared difference between the cardinalities of any two demographic groups.

$$\text{balance}_i = \frac{1}{\sqrt{\frac{2}{n_s(n_s-1)} \sum_{s=1}^{n_s} \left(\widehat{\mathcal{N}}_k(v_i) - \widehat{\mathcal{N}}_j(v_i)\right)^2 + \delta}} \quad \forall k, j \in \{1, \ldots, n_s\}, k \neq j \tag{39}$$

It should be noted that when s is binary sensitive attribute, Eq. (39) is equivalent to Eq. (10). In this way, we could adopt similar training procedure in Algorithm 1 to train the fair graph neural network.

G Detailed Experimental Settings

G.1 Dataset Descriptions

Here, we provide detailed descriptions for *Pokec-z* [40], *NBA* [9], *Credit* [11], and *Recidivism* [28]. The *Pokec-z* dataset [40] is drawn from Pokec, a Facebook-like social network in Slovakia, based on regional information. In Pokec-z, we set the sensitive attribute as the region of a user and aim to predict the field of work of a user. The *NBA* dataset [9] includes the age, nationality (US vs. overseas), salary of a NBA player for the 2016 – 2017 season. Two players are connected if one

follows another on Twitter. In this dataset, the nationality is used as the sensitive attribute, and the goal is to predict whether the salary of a player is above the median. The *Credit* dataset [11] contains age, education and payment patterns of a credit card customer. The links among customers are determined by the pairwise similarity of their credit accounts. Here, age is set as the sensitive attribute and the label is whether a user will default on credit card payments. The *Recidivism* dataset [28] consists of defendants who were released on bail during 1990 – 2009. Two defendants are connected if they share similar demographic information and criminal records. The goal is to predict whether a defendant is likely to commit a crime when released (i.e., bail) or not (i.e., no bail) with race as the sensitive attribute. The detailed statistics of the four datasets are provided in Table 4.

Table 4: The statistics of datasets.

	Pokec-z	NBA	Recidivism	Credit
# Nodes	5,631	313	9,538	21,000
# Edges	17,855	14,543	167,930	160,198
# Attributes	59	39	18	13
# Classes	2	2	2	2
Avg. Degrees	6.4	92.9	35.2	15.3
Sensitive Attr.	Region	Country	Race	Age
Label	Working field	Salary	Bail decision	Future default

G.2 Descriptions of Baseline Methods

Regarding graph neural networks without fairness considerations,

- *GCN* [1] learns the representation of a node by iteratively aggregating the representations of its 1-hop neighbors;
- *GraphSAGE* [2] aggregates node representations from a subset of 1-hop neighbors.

For fair graph neural networks,

- *FairGNN* [9] leverages adversarial learning to debias the node representations;
- *EDITS* [11] removes bias in the input data by minimizing the Wasserstein distance;
- *FairDrop* [13] mitigates bias by randomly masking edges in the input graph;
- *NIFTY* [28] debias the graph neural networks with a contrastive learning framework;
- *FMP* [12] redesigns the message passing schema in Graph Convolutional Network for bias mitigation.

H Additional Experimental Results

Table 5: Relative reduction with respect to Δ_{SP} and Δ_{EO} for BeMap on Pokec-z, NBA, Recidivism, and Credit. For BeMap (row) and BeMap (sym), the relative reduction is computed by comparing to the vanilla GCN. For BeMap (gat), the relative reduction is computed by comparing to the vanilla GAT.

Methods	Pokec-z		NBA	
	$\text{Reduction}_{\text{SP}}$ (%) ↑	$\text{Reduction}_{\text{EO}}$ (%) ↑	$\text{Reduction}_{\text{SP}}$ (%) ↑	$\text{Reduction}_{\text{EO}}$ (%) ↑
BeMap (row)	91.7	80.0	2.25	68.8
BeMap (sym)	83.6	86.8	0.12	70.8
BeMap (gat)	91.6	89.5	32.8	57.9
Methods	**Recidivism**		**Credit**	
	$\text{Reduction}_{\text{SP}}$ (%) ↑	$\text{Reduction}_{\text{EO}}$ (%) ↑	$\text{Reduction}_{\text{SP}}$ (%) ↑	$\text{Reduction}_{\text{EO}}$ (%) ↑
BeMap (row)	66.3	63.7	41.4	29.5
BeMap (sym)	79.3	53.4	40.1	45.7
BeMap (gat)	50.4	27.8	27.2	29.6

Relative Reductions with respect to Δ_{SP} and Δ_{EO}. We present the relative reduction of Δ_{SP} and Δ_{EO} for BeMap in Table 5. The relative reductions with respect to Δ_{SP} and Δ_{EO} are defined as

follows.

$$\begin{aligned} \text{Reduction}_{\text{SP}} &= \left(1 - \frac{\Delta_{\text{SP}}^{\text{BeMap}}}{\Delta_{\text{SP}}^{\text{vanilla}}}\right) \times 100\% \\ \text{Reduction}_{\text{EO}} &= \left(1 - \frac{\Delta_{\text{EO}}^{\text{BeMap}}}{\Delta_{\text{EO}}^{\text{vanilla}}}\right) \times 100\% \end{aligned} \tag{40}$$

where $\Delta_{\text{SP}}^{\text{BeMap}}$ and $\Delta_{\text{EO}}^{\text{BeMap}}$ are Δ_{SP} and Δ_{EO} for the proposed BeMap, respectively, and $\Delta_{\text{SP}}^{\text{vanilla}}$ and $\Delta_{\text{EO}}^{\text{vanilla}}$ are Δ_{SP} and Δ_{EO} for the corresponding vanilla graph neural network, i.e., vanilla GCN for BeMap (row) and BeMap (sym) and vanilla GAT for BeMap (gat), respectively. From Table 5, we observe that BeMap consistently reduce more than 25% of Δ_{SP} and Δ_{EO} on all datasets, except for BeMap (row) and BeMap (sym) on NBA. On Pokec-z, BeMap could even reduce more than 80% of Δ_{SP} and Δ_{EO} compared to vanilla graph neural networks without fairness consideration.

Table 6: Semi-supervised node classification on the Pokec-n. Higher is better (↑) for ACC and AUC (white). Lower is better (↓) for Δ_{SP} and Δ_{EO} (gray). Bold font indicates the best performance for fair graph neural networks, and underlined number indicates the second best.

Methods	Pokec-n			
	ACC (%) ↑	AUC (%) ↑	Δ_{SP} (%) ↓	Δ_{EO} (%) ↓
GCN	68.41 ± 0.30	73.4 ± 0.09	6.83 ± 1.09	9.59 ± 1.16
GraphSAGE	65.41 ± 0.74	69.35 ± 0.76	6.81 ± 0.47	11.20 ± 0.84
GAT	65.81 ± 0.43	69.30 ± 0.86	6.63 ± 0.68	10.34 ± 1.37
FairGNN	68.20 ± 0.87	<u>72.92 ± 0.18</u>	8.73 ± 1.37	11.02 ± 1.32
EDITS	66.82 ± 0.98	70.39 ± 0.02	**4.32 ± 0.43**	6.20 ± 0.74
NIFTY	66.91 ± 1.38	71.27 ± 0.32	7.72 ± 1.06	**5.83 ± 0.43**
FMP	67.62±1.27	**77.40±0.23**	32.78±1.89	29.67±2.51
BeMap (row)	<u>68.39 ± 0.52</u>	71.58 ± 0.61	5.63 ± 2.65	<u>6.11 ± 1.85</u>
BeMap (sym)	**69.12 ± 0.76**	71.59 ± 1.12	<u>5.45 ± 0.62</u>	7.97 ± 0.61
BeMap (gat)	65.47 ± 0.97	67.95 ± 0.39	5.67 ± 0.56	8.73 ± 0.50

Additional Results on Pokec-n. We conduct experiments on another social network named Pokec-n. Similar to Pokec-z, it is also drawn from the Slovakian social network Pokec, but focuses on different province in the country compared to Pokec-z. Additionally, the sensitive attribute is selected as the region of a user and aim to predict the field of work of a user, which is consistent with the settings in Pokec-z. Experimental results on Pokec-n is shown in Table 6. From the table, BeMap can still consistently mitigate bias, despite being the second bias method in terms of bias mitigation (as shown by Δ_{SP} and Δ_{EO}). In the meanwhile, BeMap achieves the best classification results (as shown by ACC and AUC) while mitigating bias, which demonstrate that BeMap achieves the best trade-off between fairness and utility.

I More Related Works

Here, we discuss more related works in designing fair message passing, and a few related work on class-imbalanced graph learning, graph rewiring and graph sampling.

Fair message passing ensures fairness on graphs by either preprocessing the input graph or redesigning message passing in graph neural networks [10, 15, 42]. [15] views message passing as the solution to an optimization and introduces Maximum Mean Discrepancy (MMD) as a regularization in the optimization problem to redesign the fairness-aware message passing. [10] learns a fair adjacency matrix for link prediction by aligning the edge weights between intra-group edges and inter-group edges. [42] reduces the discrimination risk, in order to ensure fairness on mean aggregation feature imputation. Our work bears subtle differences existing works. Compared to [15], we do not change how neighborhood aggregation procedure in message passing, but change the neighborhood itself to ensure fairness. Compared to [10], we focus on fairness in node classification rather than dyadic fairness in link prediction. Compared to [42], we only perform sampling on the adjacency matrix and do not consider changing node features.

Class-imbalanced graph learning refers to graph learning with uneven label distribution, i.e., one class has a significantly higher number of data than other classes [43–46]. For example, [44] mixes

up the nodes from the minority class and selected target nodes. [45] generates synthetic nodes from the minority class via generative adversarial networks (GAN) [47]. [46] reweighs the influence of labeled node adaptively based on their distances to class boundaries. [43] generalizes SMOTE [48] to graphs and synthesizes new samples in the embedding space. It should be noted that, different from this line of work that focuses on the imbalance with respect to class label, our work focuses on the imbalance with respect to a sensitive attribute. For more related work, please refer to [49].

Graph rewiring changes the graph topology by rewiring edges in the graph (i.e., delete an edge from a source node to a target node and add an edge between the source node to a different target node) to find a better graph topology for the learning task [50–53]. [53] proposes degree-preserving edge rewiring to maximally improve its robustness under a certain budget. Different from [53], our method focuses on the balance of neighborhoods with respect to a sensitive attribute for fairness rather than node degrees. [50] rewires the input graph via reinforcement learning for adversarial attacks on graphs, whereas our work focuses on learning fair graph neural network. [51] reduces heterophily by rewiring graphs in consideration of pairwise similarity of label/feature-distribution between a node and its neighbors. Compared to [51], we focuses on generating balanced neighborhood with respect to a sensitive attribute instead of reducing heterophily with respect to class labels. [52] incorporates the Lovász bound into graph rewiring to overcome over-smoothing, over-squashing and under-reaching in graph neural networks, while our work focuses on fairness in node classification.

Graph sampling aims to reduce computational complexity by sampling a subset of edges in the graph. [54] preserves neighbors with higher influence for node representation learning with better utility, wheres our method applies sampling to generate balanced neighborhoods for fair representation learning. [55] considers node degrees and selects edges based on the ranks of node degrees in an deterministic way. Different from it, our work does not consider node degree but the sensitive attribute of the neighbors and samples the neighborhood in a non-deterministic way. [56] develops an adaptive layer-wise sampling method by sharing the same neighborhood in low layers with different parent nodes in high layers. Different from it, we do not associate the same neighborhood with different nodes across layers, and the balanced neighborhood keeps the same across all layers.