Let's Grow an Unbiased Community : Guiding the Fairness of Graphs via New Links

Anonymous authors
Paper under double-blind review

Abstract

Graph Neural Networks (GNNs) have achieved remarkable success across diverse applications. However, due to the biases in the graph structures, graph neural networks face significant challenges in fairness. Although the original user graph structure is generally biased, it is promising to guide these existing structures toward unbiased ones by introducing new links. The fairness guidance via new links could foster unbiased communities, thereby enhancing fairness in downstream applications. To address this issue, we propose a novel framework named FairGuide. Specifically, to ensure fairness in downstream tasks trained on fairness-guided graphs, we introduce a differentiable community detection task as a pseudo downstream task. Our theoretical analysis further demonstrates that optimizing fairness within this pseudo task effectively enhances structural fairness, promoting fairness generalization across diverse downstream applications. Moreover, FairGuide employs an effective strategy which leverages meta-gradients derived from the fairness-guidance objective to identify new links that significantly enhance structural fairness. Extensive experimental results demonstrate the effectiveness and generalizability of our proposed method across a variety of graph-based fairness tasks. The code and datasets are available at `https://anonymous.4open.science/r/FairGuide-8907/README.md`.

1 INTRODUCTION

Graph data has become an essential part of many applications like social networks analysis Kumar et al. (2022), recommendation systems Wu et al. (2022), and fraud detection Cheng et al. (2020). In graph data, nodes typically represent entities or individuals, while edges capture the relationships between them. Graph Neural Networks (GNNs) have emerged as powerful tools for leveraging both node features and graph topology to perform tasks such as node classification Kipf & Welling (2016); Rong et al. (2020), graph embedding Ying et al. (2018); Zhu et al. (2020), and link prediction Zhang & Chen (2018), leading to significant improvements in task performance.

Despite the great performance of GNNs, linking biases in user graphs raise fairness concerns when deploying GNNs in critical applications. Specifically, as illustrated in Fig. 1, online social networks and residential communities often exhibit pronounced structural barriers, with tightly interconnected subgroups interacting primarily within themselves, resulting in inequitable access to resources, opportunities,and influence Saxena et al. (2024). Such biases originating from data can be further amplified by the message-passing mechanisms of GNNs. For example, social network recommendation systems have been found to systematically prevent female profiles from becoming among the most commented on or liked Bose & Hamilton (2019). Similarly, GNN-based book recommendation systems can be biased toward recommending books authored by males, thereby reinforcing existing disparities and limiting exposure to diverse content Buyl & De Bie (2020).

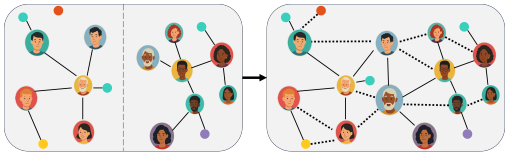


Figure 1: Guiding fairness via new links.

Biases inherent in user connections raise a critical yet underexplored question: *Can we guide existing real-world user graphs toward unbiased structures, thereby ensuring fairness in the deployment of GNN-based*

applications on these graphs? To address this question, a promising strategy is to introduce new links between users to guide the growth of graphs toward unbiased communities. As shown in Fig. 1, recommending connections between users from distinct groups can break structural barriers. Such fairness guidance via new links could foster unbiased communities in real-world scenarios. Consequently, GNN classifiers trained on the unbiased structures can yield fair outcomes for various downstream tasks. Therefore, it is crucial to investigate the problem of guiding graphs toward fairness by introducing new links.

Extensive research has been conducted to address fairness issues in graph neural networks. Specifically, in-processing Buyl & De Bie (2020); Wang et al. (2022); Zhao et al. (2022); Zhu et al. (2024a;b); Ling et al. (2023); Luo et al. (2024) and post-processing Kang et al. (2020); Masrour et al. (2020); Bose & Hamilton (2019) approaches primarily modify the training procedures or refine the output predictions of GNNs to mitigate algorithmic biases. Additionally, pre-processing Dong et al. (2022); Spinelli et al. (2021); Li et al. (2021) typically provides model-agnostic solutions by modifying the graph structures or node features to achieve fairness. Despite these significant advances, they are generally not applicable to the problem of guiding user graphs toward fairness via new links. *Firstly*, in-processing and post-processing approaches primarily focus on obtaining GNN classifiers that are fair with respect to the given task Dong et al. (2023); Zhu et al. (2024b); Wang et al. (2022). Updating the graph structures for fairness is not considered in these approaches. *Secondly*, though pre-processing approaches for fair GNNs Dong et al. (2022); Li et al. (2021) also involve modifying the input graphs, these approaches typically impose no explicit constraints on structural modifications. As a result, these methods often result in link removals and excessive link suggestions for individual users. This hinders the application of pre-processing to guiding the real-world growth of user graphs toward fairness, as users are generally reluctant to remove existing connections or accept numerous new recommendations. Furthermore, these methods typically optimize structures for specific downstream tasks, making them unsuitable for fairness guidance without predefined tasks.

Therefore, we aim to design a framework to guide the fairness of user graphs via new links. However, this goal is non-trivial, where two main challenges remain to be addressed. (i) The fairness guidance aims to reduce the inherent structural biases by selectively suggesting new links. Therefore, guidance on graph growth toward fairness should be task-agnostic. It remains challenging to effectively guide user graphs toward fairness without knowledge of specific downstream tasks. (ii) As discussed, users are unlikely to accept a large number of link suggestions. Thus, another crucial challenge is how to effectively guide graphs toward fairness by introducing only a limited number of new links. To address these challenges, we propose a novel framework named FairGuide. Specifically, to eliminate structural biases without specific downstream tasks, FairGuide leverages community detection as a pseudo downstream task. Since labels in many downstream tasks are closely correlated with community structures, ensuring structural fairness for community detection intuitively benefits various downstream applications. This intuition is further supported by our theoretical analysis. To effectively utilize the limited link-addition budget, FairGuide employs a module to identify optimal new links for structural fairness. Specifically, an efficient meta-gradient computation method is deployed to approximate the impact of potential link additions on structural fairness. This enables recommending optimal links to guide the graph toward fairness. In summary, our main contributions are:

- We focus on a novel problem of fairness guidance, which aims to guide the existing biased graph structures into fair community via introducing new links.
- We propose a novel fairness guidance framework named FairGuide, which incorporates a pseudo downstream task and link addition through meta gradients to identify optimal new links for structural fairness.
- We collect a new large-scale social network from GitHub to provide empirical validation of FairGuide in a real-world scenario.
- Both theoretical analysis and empirical results demonstrate that our FairGuide can effectively guide graphs toward fairness to facilitate the fairness of downstream tasks.

2 RELATED WORK

In this section, we provide a comprehensive review of the literature on fair graph learning and link prediction, which are closely related to our work.

2.1 Fair Graph Learning

Graph learning models have been widely adopted for analyzing topological data, demonstrating outstanding performance in various graph-based tasks (Kipf & Welling, 2016; Zhang & Chen, 2018; Tsitsulin et al., 2023). However, recent studies (Kang et al., 2020; Ma et al., 2022) reveal that fairness concerns emerge prominently in graph learning models. For instance, FairGNN (Dai & Wang, 2021) demonstrates that biases can implicitly propagate through graph structures, while EDITS (Dong et al., 2022) further demonstrates that biased graph topologies directly lead to discriminatory model outcomes. To address such challenges, numerous approaches have been proposed to improve fairness in graph learning. In-processing methods typically employ techniques like adversarial training (Ling et al., 2023; Chen et al., 2025), fairness-aware regularization (Bose & Hamilton, 2019; Jiang et al., 2024), invariant learning (Zhu et al., 2024a) to reduce bias of specific models. Some works also try to learn a fair graph representation by forcing distribution alignment (Li et al., 2024a) or preventing sensitive information leaking (Zhu et al., 2024b). Additionally, pre-processing strategies modify the graph data itself, such as edge rewiring (Spinelli et al., 2021; Li et al., 2021; Wang et al., 2025), feature masking (Ling et al., 2023) and graph reconstruction (Dong et al., 2022) to reduce the data distribution gap of different group and create fair input graphs for downstream tasks. Some data-oriented methods also achieve fairness at a lower cost through data rebalancing (Li et al., 2024b).

However, existing approaches face inherent limitations in solving the problem of guiding the fairness of graphs. In-processing methods are limited to debiasing specific graph neural networks and downstream tasks, making them inapplicable for addressing biases in the graph structures underlying graph data. Pre-processing methods attempt to achieve fairness by modifying graph data directly but often involve impractical link removal and lack explicit constraints on the amount of structural modifications, which is not suitable for the real-world community. By contrast, our method explores to obtain an unbiased graph network by guiding the original graph data towards a fairer state, which remains under-explored for prior works.

2.2 Link Prediction

Standing as one of the most fundamental tasks in graph representation learning, link prediction has been widely used in social network to help users discover and connect with individuals they have not yet interacted with or encountered (Su et al., 2020; Daud et al., 2020). Existing approaches to link prediction can be broadly categorized based on their underlying techniques, including similarity-based methods (Yu et al., 2017; Rossi et al., 2021), dimensionality reduction-based methods (Du et al., 2020) and other emerging paradigms such as graph neural networks and probabilistic models. Among these, similarity-based methods are the most prevalent and can be further divided into two paradigms: structure-based methods (Aziz et al., 2020; Luo et al., 2021) and attribute-aware methods (Xiao et al., 2021; Zhang et al., 2023). Structure-based methods rely on topological similarity metrics such as common neighbors to infer potential links while attribute-aware methods incorporate node features or semantic attributes to enhance prediction accuracy and capture more nuanced relationships. However, such similarity-based approaches face inherent fairness-related limitations. Due to the inherent similarity in the same group, traditional link prediction techniques are difficult to break the bubbles in the community (Masrour et al., 2020) and may even enlarge the bias. Some works have investigated fairness-aware link prediction or recommendation systems to mitigate algorithmic bias and improve diverse in recommended connections (Li et al., 2021; 2022). However, these works primarily focus on addressing specific link prediction or recommendation tasks, rather than providing systematic frameworks to help guide a fair and diverse graph network.

3 PRELIMINARY ANALYSIS

In this section, we present preliminaries of the fairness issues in the real-world user graph structures.

3.1 Notations

We use $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ to denote user graph where $\mathcal{V} = \{v_1, ..., v_N\}$ is the set of N nodes, representing the users in the network, $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{X} \in \mathbb{R}^{N \times M}$ is the node attribute matrix, and each node v_i is associated with a M-dimensional feature vector $\mathbf{x}_i$. $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix of $\mathcal{G}$, where $\mathbf{A}_{ij} = 1$ if nodes v_i and v_j are connected, otherwise $\mathbf{A}_{ij} = 0$. Y and $\hat{Y}$ represent the ground truth and

Table 1: Discrimination of GCN in node classification and CD tasks on GitHub.

Tasks	Metrics	Original	Link Pred.	Rand. Add
Node Classification	F1 (%) ↑	78.6 ± 0.2	78.8 ± 0.1	78.0 ± 0.1
	AUC (%) ↑	85.4 ± 0.5	85.4 ± 0.1	84.3 ± 0.1
	$\Delta_{SP}(\%)\downarrow$	12.5 ± 0.4	11.9 ± 0.2	11.0 ± 0.2
	$\Delta_{EO}(\%)\downarrow$	8.5 ± 0.5	8.3 ± 0.4	8.3 ± 0.3
CD	$\Delta_{SP}(\%)\downarrow$	41.9 ± 1.5	38.2 ± 4.0	35.7 ± 2.8

outcomes for the downstream task, respectively. In this work, we focus on binary sensitive attribute which is denoted as $s \in \{0, 1\}$.

3.2 Collection of GitHub Dataset

For the purpose of this study, we constructed a real-world dataset by crawling from the social platform **GitHub**, which is the most popular open source platform in the world. This GitHub dataset contains more than 30,000 developers' profiles and 270,000 links between users. The features of developers include gender, region , followers, development language and etc. The links represent the follower-followee relationship between two developers. Both features and links of the dataset are collected by GitHub REST API. We divide users into *developed* and *developing* according to the countries they belong to and treat it as sensitive attribute. Then we further categorize a user with more than 35 followers as popular developer and treat it as the predicting target for the classification task.

3.3 Preliminaries of Fairness

We focus on two widely used group fairness notions, i.e., statistical parity Dwork et al. (2012) and equal opportunity Hardt et al. (2016). And we consider a binary sensitive attribute, i.e., $s \in \{0, 1\}$.

Definition 1 (Statistical Parity)**.** *Statistical parity ensures that the predicted result $\hat{Y}$ is independent of the sensitive attribute s. Formally, for a binary classification task, i.e, $\hat{Y} \in \{0, 1\}$, the metric of statistical parity is computed by:*

$$\Delta_{SP} = |P(\hat{Y} = 1|s = 0) - P(\hat{Y} = 1|s = 1)|. \tag{1}$$

For a multi-category task, i.e., $\hat{Y} \in \{0, 1, ..., C\}$, Δ_{SP} extends to:

$$\Delta_{SP} = \frac{1}{2} \sum_{C_k \in \mathcal{C}} \left| P(\hat{Y} = C_k \mid s = 0) - P(\hat{Y} = C_k \mid s = 1) \right| \tag{2}$$

A lower value of Δ_{SP} indicates fairer predictions with respect to the sensitive attribute.

Definition 2 (Equal Opportunity)**.** *Equal opportunity requires that the predicted result $\hat{Y}$ is conditionally independent of the sensitive attribute s , given the true label Y. Considering a binary classification task, the metric of equal opportunity is formulated as:*

$$\Delta_{EO} = |P(\hat{Y}|s = 0, Y = 1) - P(\hat{Y}|s = 1, Y = 1)| \tag{3}$$

Similar to Δ_{SP}, lower Δ_{EO} implies fairer results of predictions.

3.4 Discrimination Analysis on GitHub Dataset

To analyze the biases of the user graph structure, we adopt GCN Kipf & Welling (2016) to perform node classification task and Louvain algorithm Blondel et al. (2008) to perform community detection task. From Tab. 1, we can observe that both Δ_{SP} and Δ_{EO} exhibit high values in the node classification and community detection tasks, indicating significant biases within the original graph structure.

Since we aim to guide the fairness of graphs via new links, we conduct preliminary analysis to investigate how simple link addition strategies will affect the performance and fairness on downstream tasks.Specifically, two link addition strategies are considered.(i) **Link Pred.**: It adds edges by a link predictor Schlichtkrull et al. (2018) ; (ii) **Rand. Add**: It randomly adds edges to the GitHub graph; Both strategies are constrained to add 4% of existing links. The results on node classification and community detection are presented in Tab. 1, where we can observe that:

- Adding edges via link prediction improves task performance but has limited effectiveness in mitigating bias. This is because connecting similar nodes primarily reinforces existing community structures, thus preserving structural biases;
- Adding some random links is slightly effective in mitigating bias. This is because these random edges can lead to inter-group connections. However, the biases are still significant and the task performance degrades largely.

These observations indicate that existing link prediction methods are insufficient to address the fairness guidance problem.

3.5 Problem Definition

The analysis presented in Sec. 3.4 demonstrates the necessity of developing methods that guide graph structures toward fairness by introducing new links. With the notations in Sec. 3.1 and fairness definitions in Sec. 3.3, the fairness guidance via link addition can be formulated as:

Problem 1 (Fairness Guidance via Link Addition)**.** *Given a user graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ with sensitive attributes $\mathcal{S}$, our objective is to strategically add n links to obtain a fair graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}', \mathbf{X})$. The number of added links, denoted as n, is subject to a constraint Δ. A GNN model $f : \mathcal{G}' \to \hat{y}$, trained on the modified graph $\mathcal{G}'$, is required to produce predictions $\hat{y}$ that satisfy the fairness criteria introduced in Sec. 3.3.*

4 METHODOLOGY

In this section, we present the details of FairGuide. FairGuide formulates fairness guidance via new links as an optimization problem, which selects new links that minimize the structural biases. Two major challenges remain to be solved: (i) how to measure the biases of structures without accessing to the downstream tasks; (ii) how to add links to effectively guide the fairness of the user graph structures within a constrained budget for link addition. As illustrated in Fig. 2, FairGuide leverages community detection as a pseudo task to reflect the structural biases for downstream tasks. This pseudo downstream task serves as a proxy for downstream tasks, allowing us to generalize fairness optimization without relying on task-specific adjustments. To fully utilize the limited budget for link addition, FairGuide employs meta-gradients derived from the fairness-guidance objective to identify new links that most effectively enhance structural fairness. Next, we introduce the bi-level optimization objective function of FairGuide followed by details of each component.

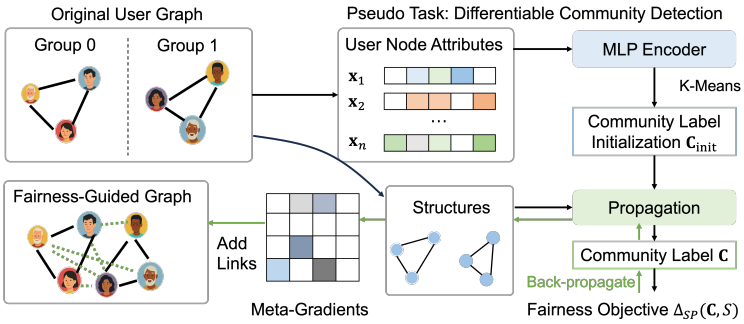


Figure 2: An illustration of FairGuide which adds new links to guide the growing of graph toward fairness.

4.1 Overall Goal of FairGuide

The goal of FairGuide is to add fairness-guiding links to $\mathcal{G}$ to ensure the GNN classifier trained on the updated graph structure $\mathcal{G}'$ produces fair predictions. Let $\mathcal{Y}_t$ and $\mathcal{L}_t$ denote the labels and the training loss for the target downstream task, respectively. Fairness guidance via the introduction of new links can be formulated as the following bi-level optimization on the structures:

$$\begin{aligned} \min_{\mathcal{G}'} \quad & \mathcal{M}\big(f_{\theta^*}(\mathcal{G}'), \mathcal{S}, \mathcal{Y}_t\big) \\ \text{s.t.} \quad & \theta^* = \arg\min_{\theta} \mathcal{L}_t\big(f_\theta(\mathcal{G}'), \mathcal{Y}_t\big), \quad \mathcal{E}' \supseteq \mathcal{E}, \quad \mathcal{E}' - \mathcal{E} \leq \Delta \end{aligned} \tag{4}$$

where $\mathcal{M}$ measures the fairness of f_{θ^*} on the updated graph structures $\mathcal{G}'$ that have been added links. $\mathcal{L}_t$ denotes the training loss of the downstream GNN classifier. The constraint $\mathcal{E}' - \mathcal{E} \leq \Delta$ limits the number of

added links to at most Δ. In the inner-level optimization, it simulates the training of the GNN classifier on the user graph for the target downstream task. In the outer-level optimization, the optimal set of new links are identified to promote fairness in the downstream GNN classifier.

4.2 Pseudo Downstream Task

With the Eq.(4), the fairness guidance via link addition is formatted to a bi-level optimization on the graph structures under constraints. However, the Eq.(4) requires to measure the fairness based on GNN predictions on downstream tasks, which are unavailable during the fairness guidance process. To address this problem, FairGuide deploys a pseudo task to reflect the fairness on the downstream task. Specifically, user communities naturally capture structural and node attribute information, and community labels often correlate strongly with labels from various downstream tasks. Thus, discrimination observed in an unsupervised community detection task can reflect structural biases relevant to downstream tasks. This intuition is further verified by the theoretical analysis. Therefore, community detection is deployed as the pseudo downstream task in FairGuide to estimate structural biases without the specific knowledge of downstream tasks. Next, we provide theoretical justification of adopting the pseudo downstream task for fairness on downstream tasks. Then, we present the updated objective function, followed by the design of community detection as the pseudo task.

Theoretical Justification. In the following, we present a theorem that justifies our motivation that alleviating the bias of downstream tasks can be achieved by reducing the correlation coefficient between the sensitive attribute $\mathcal{S}$ and the community label $\mathbf{C}$. Below, we first present the definition of the Pearson correlation coefficient followed by the theorem and proof.

Definition 3. *(Pearson Correlation Coefficient). Pearson correlation coefficient measures the linear correlation between two random variables X and Y as:*

$$\rho_{X,Y} = \frac{\mathbb{E}[(X - \mu_X) \cdot (Y - \mu_Y)]}{\sigma_X \cdot \sigma_Y} \tag{5}$$

Theorem 1. *Let C and S represent the community label and sensitive attribute, respectively. Let $\hat{Y}$ denote the output of a downstream prediction task. Assume that C is highly correlated with $\hat{Y}$, i.e., $\rho_{C,\hat{Y}}$ is larger than a positive constant $\cos\alpha$. If the model is trained to make $\rho_{S,C}$ close to zero, i.e., within*

$$\left[\cos\left(\frac{\pi}{2} + \delta\right), \cos\left(\frac{\pi}{2} - \delta\right)\right],$$

where δ is close to 0, then the correlation between the sensitive attribute S and the downstream prediction $\hat{Y}$ satisfies:

$$\rho_{S,\hat{Y}} \in \left[\cos\left(\frac{\pi}{2} + \delta + \alpha\right), \cos\left(\frac{\pi}{2} - \delta - \alpha\right)\right].$$

This theorem demonstrates that when bias mitigation in pseudo-community detection is achieved through link addition, the upper bound of bias in downstream tasks is consequently constrained. This theorem validates the deployment of community detection as the pseudo task to facilitate the structural fairness without the knowledge of downstream tasks.

Updated Objective Function with Pseudo Downstream Task. By adopting community detection as the pseudo task, the inner-level optimization is updated to the process of community detection. To measure the bias of the community labels $\mathbf{C}$, we adopt Δ_{SP} based on the predicted community assignments. Thus, the optimization problem in Eq.(4) can be reformulated as:

$$\begin{aligned} &\min_{\mathcal{G}'} \Delta_{SP}(\mathbf{C}, \mathcal{S}) \\ \text{s.t.} \quad &\mathbf{C} = \text{CommunityDetection}(\mathcal{G}'), \\ &\mathcal{E}' \supseteq \mathcal{E}, \quad \mathcal{E}' - \mathcal{E} \leq \Delta. \end{aligned} \tag{6}$$

Efficient Differentiable Community Detection. Despite the advantages of using community detection as a pseudo task, traditional community detection methods are typically non-differentiable with

respect to the graph structure. One may utilize GNN-based community detection $\mathbf{C} = f_{\theta'}(\mathcal{G}')$ with $\theta' = \arg\min_\theta \mathcal{L}_{CD}(f_\theta(\mathcal{G}'))$, where $\mathcal{L}_{CD}$ is the loss for community detection Sun et al. (2021). However, in this situation, computing the gradient with respect to the graph structure is computationally expensive, as it involves differentiating through the optimization process:

$$\frac{\partial \mathbf{C}}{\partial \mathcal{G}'} = \frac{\partial f_{\theta'}(\mathcal{G}')}{\partial \mathcal{G}'} + \frac{\partial f_{\theta'}(\mathcal{G}')}{\partial \theta'} \frac{\partial \theta'}{\partial \mathcal{G}'} \tag{7}$$

where the latter term $\frac{\partial \theta'}{\partial \mathcal{G}'}$ requires differentiating through the iterative training procedure of the GNN. To overcome this issue, we decouple the community detection into attribute-based clustering and structure-based aggregation. First, we utilize an MLP-based self-supervised auto-encoder to obtain latent feature representations for nodes. These representations are then clustered via K-means to generate initial community labels:

$$\mathbf{C}_{\text{init}} = \text{K-means}(\text{MLP}(\mathbf{X})). \tag{8}$$

The number of communities C is the predefined hyperparameter.

Inspired by label propagation Zarezadeh et al. (2022); Gasteiger et al. (2019), we propose to aggregate the initial community labels by graph structure, enabling the incorporation of structural information. Specifically, the aggregation process for final community labels is described as:

$$\mathbf{C} = \text{softmax}\left((1-\alpha)^K \hat{\mathbf{A}}^K \mathbf{C}_{\text{init}} + \alpha \sum_{i=0}^{K-1} (1-\alpha)^i \hat{\mathbf{A}}^i \mathbf{C}_{\text{init}} \right), \tag{9}$$

where $\hat{A}$ is the symmetric normalized adjacency matrix and α is the probability of restarting from the original node features in aggregation. After K-layer aggregation, we get the final community labels. Since label propagation with structures is decoupled with the MLP-based community label initialization, the computation of $\frac{\partial \mathbf{C}}{\mathbf{A}}$ could be computed without differentiating through the MLP parameters, resulting in much more efficient gradient computation compared to Eq.(7).

4.3 Fairness-Guided Link Addition via Meta Gradients and Gumbel-max Sampling

In this subsection, we present how to optimize the updated objective function Eq.(6) given the differentiable community detection. Following previous works in updating structures Zügner & Günnemann (2019), we solve the bi-level optimization problem in Eq.(6) using meta-gradients. Next, we first derive the computation of meta-gradients. Then, we present a strategy of link addition with meta-gradients that satisfies constraints on link discreteness and the number of new links.

Meta-Gradients of Candidate Links. The meta-gradient of a candidate link indicates how the addition of this link would influence the fairness of the pseudo community detection task. In the following, we formally illustrate the computation of the meta-gradient using the differentiable community detection method:

$$\nabla_{\mathbf{A}}^{\text{meta}} = \frac{\partial \Delta_{SP}(\mathbf{C}, \mathcal{S})}{\partial \mathbf{C}} \left(\frac{\partial \mathbf{C}}{\partial \mathbf{A}} + \frac{\partial \mathbf{C}}{\partial \mathbf{C}_{\text{init}}} \frac{\partial \mathbf{C}_{\text{init}}}{\partial \mathbf{A}} \right) \tag{10}$$

Here, the first term represents the direct influence of the perturbed graph structure $\hat{\mathbf{A}}$ on the final model output, while the second term accounts for the indirect effects propagated through initialized community labels. Note that initialized community labels are independent to the graph structure. Therefore, we can deduce that the indirect term $\frac{\partial \mathbf{C}_{\text{init}}}{\partial \mathbf{A}} = 0$. So finally the meta-gradient consequently be simplified as:

$$\nabla_{\mathbf{A}}^{\text{meta}} = \frac{\partial \Delta_{SP}(\mathbf{C}, \mathcal{S})}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{A}}. \tag{11}$$

With the Eq.(11), the meta-gradients of candidate links can be efficiently computed. Next, we'll introduce how we optimize the adjacent matrix $\mathbf{A}$ discretely to satisfy the constraints of our optimization problem .

Gumbel-max Sampling for Link Addition. With the Eq.(11), we get meta-gradients of graph structures, i.e., $\nabla_{\mathbf{A}}^{\text{meta}}$. And based on meta-gradient, a straightforward way to optimize the graph structure is:

$$\mathbf{A}' = \mathbf{A} - \alpha \nabla_{\mathbf{A}}^{\text{meta}}. \tag{12}$$

However, this approach cannot guarantee the constraints of link discreteness and the maximum number of link addition. To overcome these issues, we reformulate edges adding process as a stochastic sampling process based on reweighted meta-gradients. According to the rule for the derivative of a scalar with respect to a matrix, the final gradient matrix $\nabla_{\mathbf{A}}^{\text{meta}}$ has the same shape as the adjacency matrix $\mathbf{A}$, and $\nabla_{\mathbf{A_{i,j}}}^{\text{meta}}$ represents the possible effect of the edge i,j on fairness. Intuitively, to grow an unbiased community, more group-cross edges are needed to add, so we amplify gradients for edges connecting nodes with differing sensitive attributes while enabling probabilistic batch selection. Specifically, for each candidate edge $(i, j) \in \mathcal{E} = \{(u, v) \mid \mathbf{A}_{u,v} = 0\}$, we compute an adjusted gradient score:

$$\tilde{\nabla} i, j = -\nabla_{\mathbf{A_{i,j}}}^{\text{meta}} \cdot (1 + \beta \cdot \mathbb{I}(s_i \neq s_j)), \tag{13}$$

Here β controls inter-group connection incentives, and taking a negative value of the gradient means potential edges with smaller gradients have higher scores. We then sample edges via Gumbel-softmax reparameterization:

$$P(i, j) = \frac{\log(\tilde{\nabla} i, j + \epsilon) + g_{i,j}}{\tau}, \quad g_{i,j} \sim \text{Gumbel}(0, 1), \tag{14}$$

where $g_{i,j}$ is the gumbel noise, τ modulates exploration-exploitation trade-offs and ϵ prevents numerical underflow. And simultaneously the top-k edges with highest scores are added :

$$\mathbf{A}' = \mathbf{A} + \sum_{(i,j) \in \text{Top}k(P(i,j))} \delta_{i,j}. \tag{15}$$

These formulas describe the whole meta-gradient optimization process for dynamically updating the graph structure to optimize fairness. After one round of the process of sampling edges to update the graph structure, the whole optimization pipeline goes back to aggregation step. Repeating the cycle for several times, we get the final structure $\mathbf{A}'$ with new fair links. The detailed algorithm of FairGuide can be found in Appendix A.

4.4 Time Complexity Analysis

In this section, we give the time complexity for adding a link for a specific node v. Specifically, the computational complexity of FairGuide is determined by two components, i.e., differentiable community detection, and fairness-guided link addition via meta gradients. To recommend a link for node v, the differentiable community detection involve the existing links and candidate links in the differential aggregation phase. The time complexity would be $O((d+1)c|\mathcal{V}|)$, where d and c denotes the average degree of graph and predefined community number. According to Eq.(11), the cost of meta-gradient computation is the same as the forward computation $O((d + 1)c|\mathcal{V}|)$. Finally, the selection of optimal link would be $O(|\mathcal{V}| \log |\mathcal{V}|)$. Therefore, for a given node v, the total time complexity of suggesting a link to add would be $O(2(d + 1)c|\mathcal{V}| + |\mathcal{V}| \log |\mathcal{V}|)$. In comparison, a GNN-based link recommendation has the time complexity as $O(dh|\mathcal{V}| + |\mathcal{V}| \log |\mathcal{V}|)$, where h indicates the hidden dimension. Thus, the computational cost of FairGuide is similar to standard link recommendation methods. Additionally, the actual running time of adding a single link is reported in the Tab. 5 in the Appendix, which is less than 0.02 seconds.

5 Experiments

In this section, we conduct experiments on real-world user graphs to answer the following research questions:

- **RQ1**: Is FairGuide capable of adding new links that can effectively promote fairness in user graph structures?
- **RQ2**: How does the pseudo downstream task and dynamic link addition by meta-gradients contribute to FairGuide in guiding the fairness of graph structures ?
- **RQ3**: How does the number of new links introduced for fairness guidance influence the fairness of GNN-based applications?

5.1 Experimental Settings

5.1.1 Datasets

Three real-world include Pokec-n, Pokec-z and GitHub are adopted for our experiments. The GitHub dataset is introduced in Sec. 3.2. The Pokec-z/n datasets Dai & Wang (2021) are both sampled from Pokec, which is the most popular social network in Slovakia. Node features include gender, age, hobbies, regions, and other attributes. Edges in the dataset represent friendship relationships among users. The statistics of three datasets are in the table 2. We set the labels larger than 1 as 1 and select 25% of labeled nodes as the validation set and 25% labeled nodes as the test set. We also select 7000 ground-truth labels as training set for GitHub dataset and 500 labels for Pokec datasets following the prior works.

Table 2: The statistics of the three datasets.

Dataset	# Nodes	# Edges	Sensitive attribute	Label
GitHub	32,132	270,088	Country	Popularity
Pokec-z	67,797	882,765	Region	Job Field
Pokec-n	66,569	729,129	Region	Job Field

5.1.2 Baselines

Since our proposed framework guides graph fairness via link addition, we compare it with three pre-processing methods that could debias the graph structures via injecting new links. Additionally, we include two simple link-addition strategies, i.e., Rand. Add and Link Pred., for comparisons:

- **Rand. Add**: This baseline randomly add links to the graph.
- **Link Pred.** Schlichtkrull et al. (2018): It trains a GCN-based graph autoencoder. Then, pairwise cosine similarity between node embeddings are used for the link prediction.
- **EDITS** Dong et al. (2022): A preprocessing framework designed to mitigate bias in attributed networks by optimizing bias metrics across both attribute and structural modalities. We adopt its structure debiasing part and restrict it to only add edges according to the trained edge score.
- **Fairgen** Zheng et al. (2024): A generative model that promotes fairness in graph generation by incorporating label information and fairness constraints, thereby reducing representation disparity. We sample new edges in the generated graph structure and combine them with the original graph structure.
- **Graphair** Ling et al. (2023): A method for learning fair graph representations through the automatic discovery of fairness-aware augmentations.We choose its structure augmentation part and ensure the preservation of original edges.

5.1.3 Implementation Details

FairGuide is implemented using Pytorch and optimized via Adam optimizer Kingma & Ba (2014). Each iteration we fix the adding number as 100. For each method, we conduct experiments with seed {10,20,30,40,50} and compute the mean and standard deviations of F1, AUC, Δ_{SP} and Δ_{EO}. All methods consistently add the same number of new edges, corresponding to 3% of the original graph's total edge number for Pokec-n, 1.5% for Pokec-z and 4% for GitHub. For GNN backbone, we employ 2-layer GCN, 2-layer GraphSage and 10-layer APPNP. The hidden dimension is set as 128, the learning rate and training epochs are set as 1×10^{-3} and 1000 respectively.

Table 3: Results for the node classification on GCN model, with best results in bold and runner-up results in gray.

Dataset	Metrics (%)	Vanilla	Rand. Add	Link Pred.	Edits	Graphair	Fairgen	FairGuide
GitHub	F1 (↑)	78.6 ± 0.2	78.0 ± 0.1	78.8 ± 0.1	**78.6 ± 0.1**	77.5 ± 0.2	77.5 ± 0.2	77.8 ± 0.1
	AUC (↑)	85.4 ± 0.5	84.3 ± 0.1	85.4 ± 0.1	**84.8 ± 0.3**	84.2 ± 0.1	84.0 ± 0.1	84.3 ± 0.1
	Δ_{SP} (↓)	12.5 ± 0.4	11.0 ± 0.2	11.9 ± 0.2	11.5 ± 0.8	11.1 ± 0.3	10.8 ± 0.3	**8.6 ± 0.2**
	Δ_{EO} (↓)	8.5 ± 0.5	8.3 ± 0.3	8.3 ± 0.4	8.5 ± 0.8	8.4 ± 0.3	8.3 ± 0.1	**6.0 ± 0.2**
pokec-n	F1 (↑)	66.8 ± 0.7	67.1 ± 0.3	66.7 ± 0.5	65.4 ± 0.4	65.4 ± 0.3	65.2 ± 0.3	**66.2 ± 0.3**
	AUC (↑)	75.4 ± 0.1	74.8 ± 0.1	75.4 ± 0.2	72.8 ± 0.6	74.3 ± 0.1	74.2 ± 0.2	**74.9 ± 0.2**
	Δ_{SP} (↓)	8.5 ± 0.7	9.2 ± 1.0	7.9 ± 0.8	3.7 ± 1.3	9.7 ± 0.4	9.6 ± 0.7	**1.3 ± 0.8**
	Δ_{EO} (↓)	11.9 ± 1.4	12.3 ± 1.0	11.0 ± 1.6	6.3 ± 1.5	11.7 ± 1.6	11.7 ± 0.5	**3.1 ± 0.5**
pokec-z	F1 (↑)	70.6 ± 0.4	70.5 ± 0.2	70.3 ± 0.6	67.1 ± 1.1	69.0 ± 0.5	69.1 ± 0.9	**70.2 ± 0.4**
	AUC (↑)	77.2 ± 0.1	76.8 ± 0.1	77.2 ± 0.1	75.5 ± 0.7	76.0 ± 0.1	76.0 ± 0.3	**76.1 ± 0.1**
	Δ_{SP} (↓)	9.1 ± 0.9	8.8 ± 0.8	9.5 ± 0.8	5.0 ± 2.0	6.4 ± 1.5	6.9 ± 2.4	**3.1 ± 0.7**
	Δ_{EO} (↓)	8.2 ± 1.2	8.0 ± 0.7	8.1 ± 1.1	5.1 ± 1.8	5.5 ± 2.0	7.6 ± 2.6	**4.6 ± 0.4**

Table 4: Results on the downstream community detection task, with best results in bold and runner-up results in gray.

Dataset	Metrics (%)	Vanilla	Rand. Add	Link Pred.	Edits	Graphair	Fairgen	FairGuide
GitHub	Δ_{SP} (↓)	41.9 ± 1.5	35.7 ± 2.8	38.2 ± 4.0	40.2 ± 3.6	31.9 ± 2.1	36.3 ± 3.8	**29.0 ± 2.2**
pokec-n	Δ_{SP} (↓)	83.2 ± 3.0	76.1 ± 0.8	77.5 ± 2.6	79.6 ± 3.3	79.2 ± 2.9	80.0 ± 1.4	**74.1 ± 3.4**
pokec-z	Δ_{SP} (↓)	77.7 ± 0.9	79.3 ± 3.7	80.4 ± 3.0	82.3 ± 3.8	81.7 ± 1.7	79.6 ± 0.7	**70.1 ± 4.0**

5.2 Results of Fairness Guidance

To answer **RQ1**, we first demonstrate that the links suggested by FairGuide effectively improve the fairness of various GNN models across different downstream tasks. Then, we discuss the trade-off between fairness and utility in user graphs.

Fairness Improvements on Downstream Tasks. Two types of downstream tasks, i.e., node classification and community detection, are utilized to evaluate the effectiveness of FairGuide in guiding the user graph structures toward fairness. For node classification task, a GCN is trained on these updated graph structures as the downstream classifier. For community detection task, we adopt the Louvain method Blondel et al. (2008) based on modularity optimization. We utilize the Δ_{SP} and Δ_{EO} as metrics to evaluate the fairness of the GCN-based downstream classifiers. Fairer results in downstream tasks indicate a better performance in guiding the graph structure towards fairness. The results on two downstream tasks are presented in Tab. 3 and Tab. 4. From these tables, we observe that:

- For the node classification task, all baselines perform poor in fairness metric. In contrast, models trained on graphs with new links added by FairGuide achieves great performance in fairness and only sacrifices little task performance.
- For the community detection task, the Rand. Add baseline shows limited effectiveness in enhancing fairness. Furthermore, links added by some baseline methods even amplify biases between communities. By contrast, fairness-guided links added by FairGuide could bring significant improvements in community detection fairness compared with baselines.

Comparison with fair link predictor. Besides preprocessing baselines, We additionally compare against two fair link prediction methods: FairLp Li et al. (2022) and a within-group fairness regularization strategy WG-Reg Subramonian et al. (2024). We adopt these fair link predictors to add the same number of fair links as our FairGuide. As shown in Fig. 3(a), our FairGuide consistently achieves greater fairness improvements in downstream tasks compared to fair link prediction methods.

Trade-off between Utility and Fairness. We visualize the F1-Δ_{SP} of different methods by varying the strength of fairness guidance. Specifically, we vary the percentage of adding fairness-guided links from

0.5% to 3%. The results are given in Fig. 3(b–c). Scattered points in various colors represent results from different methods. We further fit a straight line for each method. From these two figures, we observe that great improvements in fairness are always accompanied with a decrease in task performance. And compared to other baseline models, FairGuide achieves a better utility-fairness balance. FairGuide either has better task performance at the same fairness level or reduce more bias at equivalent task performance.

Generalization to Various GNN Architectures. In addition, to further verify the structural fairness of graphs guided by FairGuide, we evaluate the impact of fairness-guided links on two different GNN backbones, i.e. , GraphSage Hamilton et al. (2017) and APPNP Gasteiger et al. (2019). Results in fairness are presented in Fig. 4(a–b). From figure, we observe that different GNN backbones consistently achieve fairness improvements when trained on graphs guided by FairGuide. This demonstrates that FairGuide can grow fair graph structures that facilitate different GNN models.

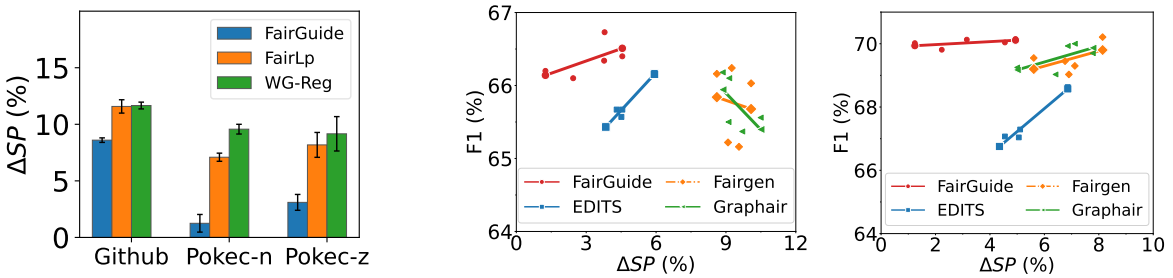
5.3 Ablation Study

FairGuide relies on two critical components to effectively guide the fairness of graph structures: the pseudo downstream task and link addition with Gumbel-max sampling. To assess the impact of these components to answer **RQ2**, we conduct ablation studies using the GCN-based node classification task.Specifically, to demonstrate the importance of employing community detection as the pseudo downstream task, we compare with a variant named FairGuide\C where the initial community labels are replaced with randomly generated labels. Additionally, to evaluate the effectiveness of Gumbel-max sampling in the fairness-guided link addition , we introduce another variant, FairGuide\S, in which links are determined through a single iteration without dynamic updates. The results are shown in Fig. 4(c–d), where we observe:

- FairGuide achieves smaller Δ_{SP} compared to the variant FairGuide\C, where the community detection labels are replaced by propagated random labels. This is because community labels correlate with labels in downstream tasks. Thus, incorporating community detection as a pseudo task effectively enhances fairness in downstream tasks, which empirically verifies our theoretical analysis.
- FairGuide\S consistently underperforms FairGuide in fairness evaluations across all datasets, thereby highlighting the importance of dynamic link addition strategies via Gumbel-max sampling.

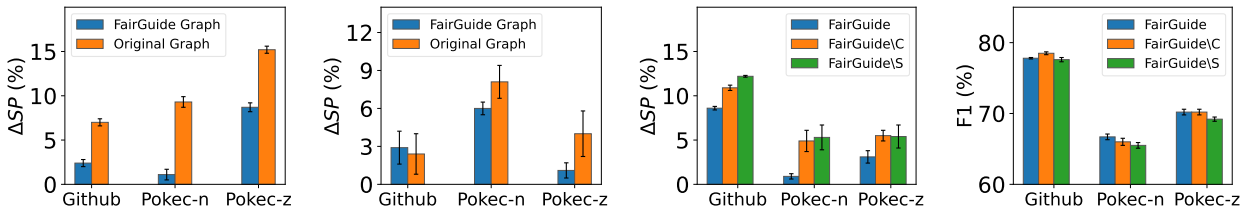
5.4 Impacts of the Number of New Links

To answer the **RQ3**, we vary edge adding percentage based on the existing edges from 0.5% to 3% to investigate the impact of link addition size. We conduct experiments for node classification task based on GCN. The impacts of link addition rate on pokec-n and pokec-z are shown in Fig. 5. We have similar observations on other datasets. From these results, we can observe that with FairGuide, the fairness of the downstream GNN classifier improves as the percentage of added edges increases. During this process, FairGuide consistently outperforms baseline methods in terms of fairness while preserving utility, which demonstrates the effectiveness of FairGuide in guiding graph structures toward fairness.



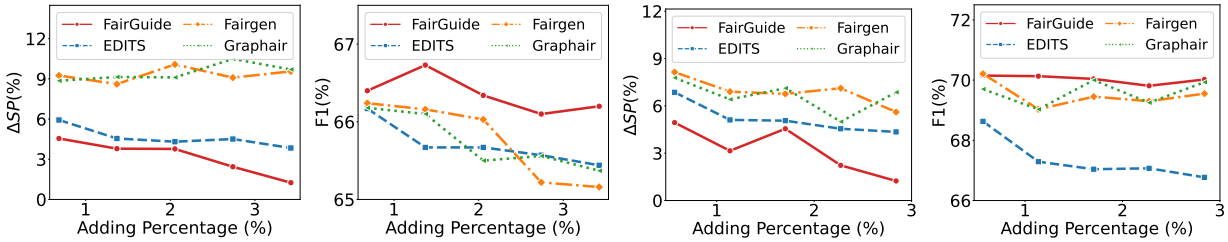
(a) Compared with fair link predictor. (b) Trade-off on Pokec-n. (c) Trade-off on Pokec-z.

Figure 3: (a) Comparison with fair link predictors in guiding the graph structures toward fairness. (b–c) Visualization of utility-fairness trade-off; methods in the upper-left indicate better utility and fairness.



(a) APPNP (b) GraphSage (c) ΔSP (%) (d) F1 (%)

Figure 4: (a–b) Fairness improvements across GNN backbones. (c–d) Ablation studies on three user graphs.

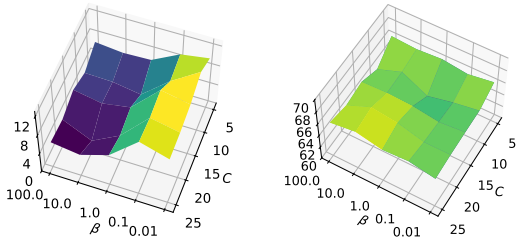


$\Delta_{SP}(\%)$ on Pokec-n F1(%) on Pokec-n $\Delta_{SP}(\%)$ on Pokec-z F1(%) on Pokec-z

Figure 5: Impacts of number of added links.

5.5 Hyperparameter Analysis

In FairGuide, the cross-group boost rate β and the number of community clusters C are two key hyperparameters.To investigate how the two hyperparameters affect fairness and task performance, We conduct GCN node classification task on the Pokec-n. Similar trends are also observed in other datasets. We vary β to $\{0.01,0.1,1.0,10.0,100.0\}$ and C to $\{5,10,15,20,25\}$, respectively. All other settings follow the description given in Sec. 5.1. As shown in Fig. 6, the task performance F1 score remains stable across the tested hyperparameter ranges, fluctuating by only around 1–2 percentage points.



(a) Δ_{SP} (%) (b) F1 score (%)

Figure 6: Hyperparameter sensitivity analysis.

For the fairness performance, FairGuide achieves a strong debiasing effect when $C \geq 15$ and $\beta \geq 1.0$. A larger number of clusters C allows the model to capture more fine-grained feature information, and a large cross-group boost rate β effectively establishes connections between different sensitive groups, which will benefit the graph structural fairness.

6 Conclusion and Future Work

In this work, we study a novel problem of fairness guidance via link addition which aims to guide existing graph structures toward fairness. To solve this problem, we form it as bi-level optimization problem and propose an algorithm called FairGuide which utilizes the differentiable pseudo community detection task to optimize the graph structure. Specifically, we theoretically demonstrate generalize fairness can be achieved by reducing the bias in pseudo task via new links. To efficiently identify beneficial new links, we compute meta-gradients with respect to the graph structure and adopt an edge-sampling technique to discretely update the structure. Experiment results on real-world datasets show the effectiveness of our proposed algorithm in solving the fair guidance problem. For the future work, there are several directions to investigate. First, in this paper links are added just according to the fairness without consideration of enhancing the structure utility. In future we will explore how to add links to simultaneously improve the task performance and fairness. Second, solving various sensitive attribute by adding links is also an interesting problem. We plan to extend the FairGuide to achieve more general fairness on diverse sensitive attributes.

References

Furqan Aziz, Haji Gul, Ishtiaq Muhammad, and Irfan Uddin. Link prediction using node information on local paths. *Physica A: Statistical Mechanics and its Applications*, 557:124980, 2020.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

Avishek Bose and William Hamilton. Compositional fairness constraints for graph embeddings. In *ICML*, pp. 715–724. PMLR, 2019.

Maarten Buyl and Tijl De Bie. Debayes: a bayesian method for debiasing network embeddings. In *ICML*, pp. 1220–1229. PMLR, 2020.

Wei Chen, Meng Yuan, Zhao Zhang, Ruobing Xie, Fuzhen Zhuang, Deqing Wang, and Rui Liu. Fairdgcl: Fairness-aware recommendation with dynamic graph contrastive learning. *TKDE*, 2025.

Dawei Cheng, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. Graph neural network for fraud detection via spatial-temporal attention. *TKDE*, 34(8):3800–3813, 2020.

Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *WSDM*, pp. 680–688, 2021.

Nur Nasuha Daud, Siti Hafizah Ab Hamid, Muntadher Saadoon, Firdaus Sahran, and Nor Badrul Anuar. Applications of link prediction in social networks: A review. *Journal of Network and Computer Applications*, 166:102716, 2020.

Yushun Dong, Ninghao Liu, Brian Jalaian, and Jundong Li. Edits: Modeling and mitigating data bias for graph neural networks. In *WWW*, pp. 1259–1269, 2022.

Yushun Dong, Jing Ma, Song Wang, Chen Chen, and Jundong Li. Fairness in graph mining: A survey. *TKDE*, 35(10):10583–10602, 2023.

Xingbo Du, Junchi Yan, Rui Zhang, and Hongyuan Zha. Cross-network skip-gram embedding for joint network alignment and link prediction. *TKDE*, 34(3):1080–1095, 2020.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *ITCS*, pp. 214–226, 2012.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2019.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *NeurIPS*, 30, 2017.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *NeurIPS*, 29, 2016.

Zhimeng Jiang, Xiaotian Han, Chao Fan, Zirui Liu, Na Zou, Ali Mostafavi, and Xia Hu. Chasing fairness in graphs: A gnn architecture perspective. In *AAAI*, volume 38, pp. 21214–21222, 2024.

Jian Kang, Jingrui He, Ross Maciejewski, and Hanghang Tong. Inform: Individual fairness on graph mining. In *SIGKDD*, pp. 379–389, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Sanjay Kumar, Abhishek Mallik, Anavi Khetarpal, and Bhawani Sankar Panda. Influence maximization in social networks using graph embedding and graph neural network. *Information Sciences*, 607:1617–1636, 2022.

Peizhao Li, Yifei Wang, Han Zhao, Pengyu Hong, and Hongfu Liu. On dyadic fairness: Exploring and mitigating bias in graph connections. In *ICLR*, 2021.

Yanying Li, Xiuling Wang, Yue Ning, and Hui Wang. Fairlp: Towards fair link prediction on social network graphs. In *ICWSM*, volume 16, pp. 628–639, 2022.

Yibo Li, Xiao Wang, Yujie Xing, Shaohua Fan, Ruijia Wang, Yaoqi Liu, and Chuan Shi. Graph fairness learning under distribution shifts. In *WWW*, pp. 676–684, 2024a.

Zhixun Li, Yushun Dong, Qiang Liu, and Jeffrey Xu Yu. Rethinking fair graph neural networks from re-balancing. In *SIGKDD*, pp. 1736–1745, 2024b.

Hongyi Ling, Zhimeng Jiang, Youzhi Luo, Shuiwang Ji, and Na Zou. Learning fair graph representations via automated data augmentations. In *ICLR*, 2023.

Hongsheng Luo, Longjie Li, Yakun Zhang, Shiyu Fang, and Xiaoyun Chen. Link prediction in multiplex networks using a novel multiple-attribute decision-making approach. *Knowledge-Based Systems*, 219: 106904, 2021.

Renqiang Luo, Huafei Huang, Shuo Yu, Zhuoyang Han, Estrid He, Xiuzhen Zhang, and Feng Xia. Fugnn: Harmonizing fairness and utility in graph neural networks. In *SIGKDD*, pp. 2072–2081, 2024.

Jing Ma, Ruocheng Guo, Mengting Wan, Longqi Yang, Aidong Zhang, and Jundong Li. Learning fair node representations with graph counterfactual fairness. In *WSDM*, pp. 695–703, 2022.

Farzan Masrour, Tyler Wilson, Heng Yan, Pang-Ning Tan, and Abdol Esfahanian. Bursting the filter bubble: Fairness-aware network link prediction. In *AAAI*, volume 34, pp. 841–848, 2020.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *ICLR*, 2020. URL `https://openreview.net/forum?id=Hkx1qkrKPr`.

Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. Knowledge graph embedding for link prediction: A comparative analysis. *TKDD*, 15(2):1–49, 2021.

Akrati Saxena, George Fletcher, and Mykola Pechenizkiy. Fairsna: Algorithmic fairness in social network analysis. *ACM Computing Surveys*, 56(8):1–45, 2024.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, pp. 593–607. Springer, 2018.

Indro Spinelli, Simone Scardapane, Amir Hussain, and Aurelio Uncini. Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning. *IEEE Transactions on Artificial Intelligence*, 3(3): 344–354, 2021.

Zhan Su, Xiliang Zheng, Jun Ai, Yuming Shen, and Xuanxiong Zhang. Link prediction in recommender systems based on vector similarity. *Physica A: Statistical Mechanics and its Applications*, 560:125154, 2020.

Arjun Subramonian, Levent Sagun, and Yizhou Sun. Networked inequality: preferential attachment bias in graph neural network link prediction. In *ICML*, 2024.

Jianyong Sun, Wei Zheng, Qingfu Zhang, and Zongben Xu. Graph neural network encoding for community detection in attribute networks. *IEEE Transactions on Cybernetics*, 52(8):7791–7804, 2021.

Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel Müller. Graph clustering with graph neural networks. *JMLR*, 24(127):1–21, 2023.

Nan Wang, Lu Lin, Jundong Li, and Hongning Wang. Unbiased graph embedding with biased graph observations. In *WWW*, pp. 1423–1433, 2022.

Zichong Wang, Zhibo Chu, Thang Viet Doan, Shaowei Wang, Yongkai Wu, Vasile Palade, and Wenbin Zhang. Fair graph u-net: A fair graph learning framework integrating group and individual awareness. In *AAAI*, volume 39, pp. 28485–28493, 2025.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

Yunpeng Xiao, Rui Li, Xingyu Lu, and Yanbing Liu. Link prediction based on feature representation and fusion. *Information Sciences*, 548:1–17, 2021.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *NeurIPS*, 31, 2018.

Chuanming Yu, Xiaoli Zhao, Lu An, and Xia Lin. Similarity-based link prediction in social networks: A path and node combined approach. *Journal of Information Science*, 43(5):683–695, 2017.

Mahdi Zarezadeh, Esmaeil Nourani, and Asgarali Bouyer. Dpnlp: distance based peripheral nodes label propagation algorithm for community detection in social networks. *WWW*, pp. 1–26, 2022.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *NeurIPS*, 31, 2018.

Peiliang Zhang, Jiatao Chen, Chao Che, Liang Zhang, Bo Jin, and Yongjun Zhu. Iea-gnn: Anchor-aware graph neural network fused with information entropy for node classification and link prediction. *Information Sciences*, 634:665–676, 2023.

Tianxiang Zhao, Enyan Dai, Kai Shu, and Suhang Wang. Towards fair classifiers without sensitive attributes: Exploring biases in related features. In *WSDM*, pp. 1433–1442, 2022.

Lecheng Zheng, Dawei Zhou, Hanghang Tong, Jiejun Xu, Yada Zhu, and Jingrui He. Fairgen: Towards fair graph generation. In *ICDE*, pp. 2285–2297. IEEE, 2024.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*, 2020. URL `http://arxiv.org/abs/2006.04131`.

Yuchang Zhu, Jintang Li, Yatao Bian, Zibin Zheng, and Liang Chen. One fits all: Learning fair graph neural networks for various sensitive attributes. In *SIGKDD*, pp. 4688–4699, 2024a.

Yuchang Zhu, Jintang Li, Liang Chen, and Zibin Zheng. The devil is in the data: Learning fair graph neural networks via partial knowledge distillation. In *WSDM*, pp. 1012–1021, 2024b.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *ICLR*, 2019.

Appendix A Algorithm of FairGuide

As shown in Algorithm 1, FairGuide first initializes community labels by K-means and achieve differential community detection. Then, FairGuide further computes the Δ_{SP} metric and meta-gradient to the graph structure. Finally, new links sampled based on meta-gradients of structure are selected.

Algorithm 1: Algorithm of FairGuide

Input: A: adjacency matrix; X: node feature; Δ: link addition constraint; K: number of aggregation steps; α: restart parameter; S: sensitive attribute vector

Output: Fairness-guided graph structure $\mathbf{A'}$ with new links

$\mathbf{A'} \leftarrow \mathbf{A}$;

$\mathbf{C_{init}} \leftarrow \text{K-Means}(f_{\text{MLP}}(X))$

while *number of added links do not reach* Δ **do**

Differential Community Detection:

Obtains the community labels $\mathbf{C^{(K)}}$ by Eq.(9)

Compute Fairness Loss:

$\Delta_{SP} \leftarrow \Delta_{SP}(\mathbf{C^{(K)}}, \mathcal{S})$

Compute Edge Score:

$\nabla^{\text{meta}}_{\mathbf{A'}} \leftarrow -\frac{\partial \Delta_{SP}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{A'}} \odot (1 + \beta \cdot \mathbb{I}(s_i \neq s_j))$;

Gumbel Edge Sampling:

$G_{i,j} \sim \text{Gumbel}(0, 1)$

$\mathbf{W} = \log(-\nabla^{\text{meta}}_{\mathbf{A'}} \odot (\mathbf{A'} == 0)$

$\mathbf{P} = \frac{\log(\mathbf{W}+\epsilon)+G_{i,j}}{\tau}$, $\delta \leftarrow \text{Top}_k(\mathbf{P})$

Batch Edge Addition:

$\mathbf{A'} \leftarrow \mathbf{A'} + \sum_{(i,j)\in\delta} \mathbf{E}_{i,j}$

return $\mathbf{A'}$;

Table 5: Running time of FairGuide for adding one link.

Dataset	Github	Pokec-n	Pokec-z
Run Time	4.6×10^{-3}s	1.6×10^{-2}s	1.2×10^{-2}s

Appendix B Proof of Theorem 1

Proof. To prove this theorem, we first introduce the following lemmas and theorem:

Lemma 1. *Given a unit sphere centered at origin O, let A, B, C be three points on the surface. Assume angles $AOB = \theta_1$ and $BOC = \theta_2$. Then, the cosine of angle AOC satisfies:*

$$\cos(AOC) \in [\cos(\theta_1 + \theta_2), \cos(\theta_1 - \theta_2)].$$

Proof. From the spherical law of cosines:

$$\cos\theta_3 = \cos\theta_1 \cos\theta_2 + \sin\theta_1 \sin\theta_2 \cos B',$$

where B' is the angle opposite to B in the spherical triangle ABC. Since angles are in $[0, \pi]$, we have:

$$\cos\theta_3 \geq \cos\theta_1 \cos\theta_2 - \sin\theta_1 \sin\theta_2 = \cos(\theta_1 + \theta_2),$$
$$\cos\theta_3 \leq \cos\theta_1 \cos\theta_2 + \sin\theta_1 \sin\theta_2 = \cos(\theta_1 - \theta_2).$$

Thus, the lemma is proven. □

Lemma 2. *Given two random variables X and Y, the Pearson correlation coefficient is equivalent to the cosine similarity between their z-score vectors x' and y', where:*

$$x'_i = \frac{X_i - \mu_X}{\sigma_X}, \quad y'_i = \frac{Y_i - \mu_Y}{\sigma_Y}.$$

Proof. By definition, the Pearson correlation coefficient is:

$$\rho_{X,Y} = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} = \lim_{n\to\infty} \sum_{i=1}^{n} x'_i y'_i = \cos(x', y').$$

Thus, the lemma is proven. □

Theorem 2. *Given three random variables X, Y, Z with correlation coefficients $\rho_{X,Y} = \cos\alpha$ and $\rho_{Y,Z} = \cos\beta$, then:*

$$\rho_{X,Z} \in [\cos(\alpha + \beta), \cos(\alpha - \beta)].$$

Proof. From Lemma 2, the correlation coefficients correspond to cosine similarities between z-score vectors. Applying Lemma 1 with these vectors yields:

$$\rho_{X,Z} = \cos(x', z') \in [\cos(\alpha + \beta), \cos(\alpha - \beta)].$$

Thus, the theorem is proven. □

Now, applying Theorem 2 directly, we have:

Given $\rho_{C,\hat{Y}} = \cos\alpha$, and $\rho_{S,C}$ within $[\frac{\pi}{2} + \delta, \frac{\pi}{2} - \delta]$, we obtain:

$$\rho_{S,\hat{Y}} \in \left[\cos\left(\frac{\pi}{2} + \delta + \alpha\right), \cos\left(\frac{\pi}{2} - \delta - \alpha\right)\right].$$

Thus, the theorem is proven. □

Broader Impact Statement

There is no ethical issue in the preparation of this work. All datasets used in this study containing sensitive information were collected from publicly available information and have undergone anonymization processing.