FairWire: Fair Graph Generation

O. Deniz Kose
Department of Electrical Engineering and Computer Science
University of California Irvine
Irvine, CA, USA
okose@uci.edu

Yanning Shen[*]
Department of Electrical Engineering and Computer Science
University of California Irvine
Irvine, CA, USA
yannings@uci.edu

Abstract

Machine learning over graphs has recently attracted growing attention due to its ability to analyze and learn complex relations within critical interconnected systems. However, the disparate impact that is amplified by the use of biased graph structures in these algorithms has raised significant concerns for their deployment in real-world decision systems. In addition, while synthetic graph generation has become pivotal for privacy and scalability considerations, the impact of generative learning algorithms on structural bias has not yet been investigated. Motivated by this, this work focuses on the analysis and mitigation of structural bias for both real and synthetic graphs. Specifically, we first theoretically analyze the sources of structural bias that result in disparity for the predictions of dyadic relations. To alleviate the identified bias factors, we design a novel fairness regularizer that offers a versatile use. Faced with the bias amplification in graph generation models brought to light in this work, we further propose a fair graph generation framework, FairWire, by leveraging our fair regularizer design in a generative model. Experimental results on real-world networks validate that the proposed tools herein deliver effective structural bias mitigation for both real and synthetic graphs.

1 Introduction

The volume of graph-structured data has been explosively growing due to the advancement in interconnected systems. In this context, machine learning (ML) over graphs attracts increasing attention (1), where specifically graph neural networks (GNNs) (2; 3; 4) have been proven to handle complex learning tasks over graphs, such as social recommendation (5), traffic flow forecasting (6).

Despite the increasing research focus on graph ML, the deployment of these algorithms in real-world decision systems requires guarantees preventing disparate impacts. Here, algorithmic disparity refers to the performance gap incurred by ML algorithms with respect to certain sensitive attributes protected by anti-discrimination laws or social norms (e.g., ethnicity, religion). While algorithmic bias is a concern over tabular data (7; 8), such bias becomes more critical for learning over graphs, as the use of graph structure in the algorithm design has been demonstrated to amplify the already existing bias (9). Motivated by this, in this work, we specifically focus on structural bias and consequently the disparity in the predictions of dyadic relationships among nodes. Note that since the link predictions

[*]corresponding author

are informed by the proximity principle (nodes connect to other nodes that are similar to themselves), the bias in graph topology is directly reflected in link prediction. For example, in a social network, the denser connectivity within people from the same ethnic group leads to higher recommendation rates within these groups and may cause segregation in social relations (10). Hence, the development of a fair link prediction algorithm is of crucial importance to prevent potential segregation.

Fairness-aware algorithms typically require the knowledge of the sensitive attributes, the sharing of which can potentially create privacy concerns (11). From a scalability perspective, sharing real graphs is also accompanied by difficulties due to the ever-increasing size of graphs. All these factors contribute to the value of synthetic graph generation for a number of applications, such as recommendation systems (12), anomaly detection (13). For graph generation, data-driven models are shown to achieve state-of-the-art results (14; 15; 16), however, the fairness aspect of these models is under-explored. Recent works demonstrate that, in general, generative models tend to amplify the already existing bias in real data (17; 18), which is a potential issue for graph generation as well.

Faced with the aforementioned structural bias issues in graphs, in this work, we first carry out a theoretical analysis investigating the sources of such structural bias. Specifically, we deduce the factors that affect a commonly used bias metric, namely statistical parity (19), for link prediction. Guided by the theoretical findings, a novel fairness regularizer, $\mathcal{L}_{\text{FairWire}}$ is designed, which can be utilized for various graph-related problems, including link prediction and graph generation. In addition, an empirical analysis for a graph generation model is carried out, which reveals that the use of generative algorithms amplifies the already existing structural bias in real graph data. To resolve this issue, we design a new diffusion-based fair graph generation framework, FairWire, which leverages the proposed regularizer $\mathcal{L}_{\text{FairWire}}$. The training of diffusion model in FairWire is specifically designed to capture the correlations between the synthetic sensitive attributes and the graph connectivity, which enables fair model training with the existing techniques without revealing the real sensitive information. Overall, the contributions of this work can be summarized as follows:
c1) A theoretical analysis that reveals the causes of disparity in the predictions of dyadic relations between nodes is derived. Differing from the existing analyses regarding the statistical parity in link prediction, our analysis considers a more general setting where sensitive attributes can be non-binary.
c2) Based on the theoretical findings, we design a novel fairness regularizer, $\mathcal{L}_{\text{FairWire}}$, which can be directly utilized for link prediction, as well as for graph generation models to alleviate the structural bias in a task-agnostic way.
c3) We conduct an empirical analysis for the effect of graph generation models on the structural bias, which reveals the possible bias amplification related to these models.
c4) FairWire, a novel fair graph generation framework, is developed by leveraging $\mathcal{L}_{\text{FairWire}}$ within a diffusion model. The diffusion model is trained to capture the relations between the sensitive attributes and the graph topology, facilitating fair model training without private information leakage.
c5) Comprehensive experimental results over real-world networks show that the proposed framework can effectively mitigate structural bias and create fair synthetic graphs.

2 Related Work

Fairness-aware learning over graphs. Fairness-aware graph ML has attracted increasing attention in recent years (20; 21; 22). Existing works mainly focus on: 1) Group fairness (9; 23; 24; 25), 2) Individual fairness (26; 27), and 3) Counterfactual fairness (28; 29; 30). To mitigate bias in graph ML, different strategies are leveraged, including but not limited to adversarial regularization (9; 24; 31; 32), Bayesian debiasing (33), and graph editing (28; 34; 35; 36; 37). With a specific focus on link prediction, (19; 38) propose fairness-aware strategies to alter the adjacency matrix, while (39) designs a fairness-aware regularizer. Differing from the majority of existing strategies, our proposed design herein is guided and supported by theoretical results. Specifically, we rigorously analyze the factors in graph topology leading to disparity for link prediction considering non-binary sensitive attributes. Furthermore, the developed bias mitigation tool can be employed in a versatile manner for training the link prediction models, as well as for training the generative models to create synthetic, fair graphs.

Synthetic Graph Generation. Generating synthetic graphs that simulate the existing ones has been a topic of interest for a long time (40; 41), for which the success of deep neural networks has been demonstrated (12; 42; 43). Recently, the use of diffusion-based graph generative models has been increasing, due to their success in reflecting several important statistics of real graphs in the synthetic ones (44; 45; 46; 16; 47; 48).To the best of our knowledge, the only existing work that considers fair graph generation is (49) which only outputs a graph structure without nodal features, sensitive attributes and node labels, and also requires class labels as input. Furthermore, it focuses on the disparities in generation quality for different sensitive groups as the fairness metric, which may not be predictive for the fairness performance in downstream tasks. In contrast, our scheme herein does not require any class labels or training of a particular downstream task. In addition, differing from the existing diffusion models, our generation of graph topology and nodal features is guided by the sensitive attributes, which enables us to capture the correlations between the synthetic sensitive attributes and synthetic graph structure/nodal features. To the best of our knowledge, this work provides the first fairness-aware diffusion-based graph generation framework.

3 Preliminaries

Given an input graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, the focus of this study is investigating and mitigating the structural bias that may lead to unfair results for learning algorithms. Here, $\mathcal{V} := \{v_1, v_2, \cdots, v_N\}$ denotes the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ stands for the set of edges. Nodal features and the adjacency matrix of the input graph $\mathcal{G}$ are represented by $\mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{A} \in \{0,1\}^{N \times N}$, respectively, where $\mathbf{A}_{ij} = 1$ if and only if $(v_i, v_j) \in \mathcal{E}$. This work considers a single, potentially non-binary sensitive attribute for each node denoted by $\mathbf{s} \in \{1, \cdots, K\}^N$. In addition, $\mathbf{S} \in \{0,1\}^{N \times K}$ represents the one-hot encoding of the sensitive attributes. For the graphs with class information, $\mathbf{y} \in \mathbb{R}^N$ denotes the class labels. Node representations output by the lth GNN layer are $\mathbf{H}^{l+1}$, with $\mathbf{h}_i^{l+1} \in \mathbb{R}^{F^{l+1}}$ denoting the learned hidden representations for node v_i. $\mathbf{x}_i \in \mathbb{R}^F$, and s_i represent the feature vector, and the sensitive attribute of node v_i, respectively. Furthermore, $\mathcal{S}_k$ denotes the set of nodes whose sensitive attributes are equal to k. We define inter-edge set $\mathcal{E}^{\chi} := \{e_{ij} | v_i \in \mathcal{S}_a, v_j \in \mathcal{S}_b, a \neq b\}$, and intra-edge set $\mathcal{E}^{\omega} := \{e_{ij} | v_i \in \mathcal{S}_a, v_j \in \mathcal{S}_b, a = b\}$. Similarly, $d_i^{\chi} := \sum_{v_j \in \mathcal{V} - \mathcal{S}_a} A_{ij}, \forall v_i \in \mathcal{S}_a$ and $d_i^{\omega} := \sum_{v_j \in \mathcal{S}_a} A_{ij}, \forall v_i \in \mathcal{S}_a$ are the inter- and intra-degrees of node v_i, respectively. Finally, $U_{\mathcal{A}}$ represents the discrete uniform distribution over the elements of set $\mathcal{A}$.

4 Inspection and Mitigation of Structural Bias

This section first derives the conditions for a graph topology that leads to optimal statistical parity for link prediction. Guided by the obtained conditions, a fairness regularizer will then be presented. Statistical parity for link prediction is defined as $\Delta_{\text{SP}} := |\mathbb{E}_{(v_i,v_j) \sim U_{\mathcal{V}} \times U_{\mathcal{V}}}[g(v_i, v_j) \mid s_i = s_j] - \mathbb{E}_{(v_i,v_j) \sim U_{\mathcal{V}} \times U_{\mathcal{V}}}[g(v_i, v_j) \mid s_i \neq s_j]|$ (19), where $g(v_i, v_j)$ denotes the predicted probability for an edge between the nodes i and j. To the best of our knowledge, our analysis is the first theoretical investigation for the relation between Δ_{SP} and the graph topology considering multi-valued sensitive attributes, thus it generalizes previous findings with binary sensitive attributes (19).

4.1 Bias Analysis

This subsection derives the conditions for a fair graph topology that achieves optimal statistical parity in the ensuing link prediction task. First, we will introduce the GNN model considered in this work.

GNN model: Throughout the analysis, a stochastic graph view, $\tilde{\mathbf{A}}$, is adopted, i.e., $\tilde{A}_{ij}$ denotes the probability of an edge between the nodes v_i and v_j, and $\tilde{A}_{ij} = \tilde{A}_{ji}$. Let $\mathbf{Z}^{l+1}$ represent the aggregated representations by the lth GNN layer with ith row $\mathbb{E}_{\tilde{\mathbf{A}}}[\mathbf{z}_i^{l+1}] := \sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j^{l+1}$, where $\mathbf{c}_i^{l+1} := \mathbf{W}^l \mathbf{h}_i^l$. Then, the hidden representation output by lth GNN layer for node v_i can be written as $\mathbb{E}_{\tilde{\mathbf{A}}}[\mathbf{h}_i^{l+1}] = \sigma(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{W}^l \mathbf{h}_j^l) = \sigma(\mathbb{E}_{\tilde{\mathbf{A}}}[\mathbf{z}_i^{l+1}])$, where $\mathbf{W}^l$ is the weight matrix and $\sigma(\cdot)$ is the non-linear activation employed in the lth GNN layer.

The following assumptions are made for Theorem 1 that will be presented in this subsection:
A1: $\|\mathbf{c}_i\|_\infty \leq \delta, \forall v_i \in \mathcal{V}$.
A2: $\frac{\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\omega}]}{|\mathcal{S}_k|} \geq \frac{\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}]}{N - |\mathcal{S}_k|}, \forall v_i \in \mathcal{S}_k, \forall k \in \{1, \cdots, K\}$.
A3: $\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} \gg \mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}], \forall v_i \in \mathcal{V}$.

These assumptions are naturally satisfied by most of the real-world graphs. Assumption **A1** implies that the representations, $\mathbf{c}_i$'s, are finite. For **A2**, note that most of the real-world social networks have considerably more intra-edges than inter-edges (50), i.e., $\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\omega}] \geq \mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}]$. Thus, unless $|\mathcal{S}_i\| \gg \|\mathcal{S}_j|, i \neq j$ (extremely unbalanced sensitive group sizes), **A2** holds. Finally, **A3** holds with high probability as $\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}] = \sum_{v_i \in \mathcal{S}_{s_i}, v_j \in \mathcal{V} - \mathcal{S}_{s_i}} \tilde{A}_{ij}$. We also demonstrate that these assumptions are valid for the real-world networks we are using in Appendix A in order to further justify them.

Building upon these assumptions, Theorem 1 reveals the factors leading to the disparity between the representations of different sensitive groups obtained at any GNN layer. Specifically, it upper bounds the term $\delta_k^{(l+1)} := \|\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i^{l+1} \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{(\mathcal{V} - \mathcal{S}_k)}}[\mathbf{h}_i^{l+1} \mid s_i \neq k]\|_2$. The proof of the theorem is presented in Appendix B.

Theorem 1. *The disparity between the representations of nodes in a sensitive group $\mathcal{S}_k$ and the representations of the remaining nodes output by the lth GNN layer, $\delta_k^{(l+1)}$, can be upper bounded by:*

$$\delta_k^{(l+1)} \leq L\Big(\delta\sqrt{F^{(l+1)}}\Big(\left|\frac{p_k^{\omega}}{|\mathcal{S}_k|} - \frac{p_k^{\chi}}{N - |\mathcal{S}_k|}\right| + \left|\frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} - p_k^{\omega} - 2p_k^{\chi}}{N - |\mathcal{S}_k|} - \frac{p_k^{\chi}}{|\mathcal{S}_k|}\right|\Big) + 2\sqrt{N}\Delta_z\Big), \tag{1}$$

where L is the Lipschitz constant of the activation function $\sigma(\cdot)$, $\left\|\mathbf{z}_i^{l+1} - \operatorname{mean}(\mathbf{z}_j^{l+1} \mid v_j \in \mathcal{V})\right\|_{\infty} \leq \Delta_z$, $\forall v_i \in \mathcal{V}$, and $p_k^{\chi} := \sum_{v_i \in \mathcal{S}_k, v_j \notin \mathcal{S}_k} \tilde{A}_{i,j}$, $p_k^{\omega} := \sum_{v_i \in \mathcal{S}_k, v_j \in \mathcal{S}_k} \tilde{A}_{i,j}$.

Representation disparity resulting from the aforementioned GNN-based aggregation is examined and explained by Theorem 1. The commonly used fairness measures, such as statistical parity (19), are naturally a function of the representation disparity. Herein, we further investigate the said relation between the representation disparity and Δ_{SP} mathematically. Specifically, for a link prediction model described by a function $g(v_i, v_j) := \mathbf{h}_i^{\top}\Sigma\mathbf{h}_j$, Proposition 1 directly upper bounds Δ_{SP}. Here, $\mathbf{h}_i$ denotes the representation for node v_i that is employed for the link prediction task, i.e., the hidden representations in the final layer. The proof of Proposition 1 is presented in Appendix C.

Proposition 1. *For a link prediction model described by $g(v_i, v_j) := \mathbf{h}_i^{\top}\Sigma\mathbf{h}_j$, Δ_{SP} can be upper bounded by:*

$$\Delta_{\mathrm{SP}} \leq \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} q\|\Sigma\|_2 \delta^{max}, \tag{2}$$

where $\|\mathbf{h}_i\|_2 \leq q, \forall v_i$, $\delta^{max} := \max_k(\|\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_j \sim U_{(\mathcal{V} - \mathcal{S}_k)}}[\mathbf{h}_j \mid s_j \neq k]\|_2)$.

Combining the findings of Theorem 1 and Proposition 1, Corollary 1 further demonstrates the factors (including the topological ones) that affect the resulting statistical parity in the link prediction task.

Corollary 1. *For a link prediction model $g(v_i, v_j) := (\mathbf{h}_i^{L+1})^{\top}\Sigma\mathbf{h}_j^{L+1}$, where $\mathbf{h}_j^{L+1}$ is the representation created by Lth (final) GNN layer, Δ_{SP} can be upper bounded by:*

$$\Delta_{\mathrm{SP}} \leq \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} q\|\Sigma\|_2 L\big(\delta\sqrt{F^{(L+1)}}(\alpha_1 + \alpha_2) + 2\sqrt{N}\Delta_z\big),$$

where $\alpha_1 := \left|\frac{p_k^{\omega}}{|\mathcal{S}_k|} - \frac{p_k^{\chi}}{N - |\mathcal{S}_k|}\right|$ and $\alpha_2 := \left|\frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} - p_k^{\omega} - 2p_k^{\chi}}{N - |\mathcal{S}_k|} - \frac{p_k^{\chi}}{|\mathcal{S}_k|}\right|$.

4.2 A Regularizer for Fair Connections

The bias analysis in Subsection 4.1 brings to light the factors resulting in topological bias for a probabilistic graph connectivity. Corollary 1 shows that the topological bias can be minimized if $\alpha_1 = 0$ and $\alpha_2 = 0$. One can obtain $\alpha_1 = 0$ by ensuring $\frac{p_k^{\omega}}{p_k^{\chi}} = \frac{|\mathcal{S}_k|}{N - |\mathcal{S}_k|}, \forall k$. Meanwhile, $\alpha_2 = 0$ if $p_k^{\omega} = \sum_{v_i, v_j \in \mathcal{V}}(\tilde{\mathbf{A}}) - c|\mathcal{S}_k| - cN$ and $p_k^{\chi} = c|\mathcal{S}_k|$ for any constant $c \in \mathbb{R}$. Overall, the optimal values of p_k^{ω} and p_k^{χ} that minimize both α_1 and α_2 follow as $(p_k^{\omega})^* = \frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij}|\mathcal{S}_k|^2}{N^2}$ and $(p_k^{\chi})^* = \frac{(\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij})(N|\mathcal{S}_k| - |\mathcal{S}_k|^2)}{N^2}$. Therefore, in order to mitigate structural bias, we can design a regularizer that pushes the expected number of inter-edges and intra-edges towards $(p_k^{\chi})^*$ and $(p_k^{\omega})^*$: $\mathcal{L} := \sum_{k=1}^{K} |\sum_{v_i, v_j \in \mathcal{V}}(\tilde{\mathbf{A}} \odot (\mathbf{S}\mathbf{e}_k)(\mathbf{S}\mathbf{e}_k)^{\top})_{i,j} - (p_k^{\omega})^*| + |\sum_{v_i, v_j \in \mathcal{V}}(\tilde{\mathbf{A}} \odot (\mathbf{S}\mathbf{e}_k)(\mathbf{1} - (\mathbf{S}\mathbf{e}_k))^{\top})_{i,j} -$

$(p_k^\chi)^*|$. Here, $\mathbf{e}_k \in \mathbb{R}^K$ is the basis vector with only non-zero entry 1, at the kth element, and $\odot$ denotes the Hadamard product. Note that such a regularizer is compatible with any learning algorithm that outputs probabilities of all possible edges in the graph, e.g., topology inference algorithms.

Although $\mathcal{L}$ can be applied to several graph ML algorithms and its theory-guided design can promise effective topological bias mitigation, its design requires a single-batch learning setting due to the definitions of p_k^χ and p_k^ω (resulting in a complexity growing exponentially with N). Specifically, p_k^χ and p_k^ω are calculated based on all edge probabilities related to the all nodes in $\mathcal{S}_k$. Therefore, regularizing the values of p_k^χ and p_k^ω will lead to scalability issues for large graphs. To tackle this challenge, we only focus on the optimal ratio between the expected number of intra- and inter-edges, i.e., $\frac{p_k^\omega}{p_k^\chi} = \frac{|\mathcal{S}_k|}{N-|\mathcal{S}_k|}, \forall k \in \{1, \cdots, K\}$, which is governed by α_1. The idea is to manipulate the ratio between the expected number of intra- and inter-edges in each mini-batch of nodes for a better scalability. We call the corresponding batch-wise fairness regularizer $\mathcal{L}_{\text{FairWire}}$, which follows as

$$\mathcal{L}_{\text{FairWire}}(\tilde{\mathbf{A}}, \mathcal{B}) := \sum_{k=0}^{K} \left| \frac{\sum_{v_i, v_j \in \mathcal{B}} (\tilde{\mathbf{A}} \odot (\mathbf{S}\mathbf{e}_k)(\mathbf{S}\mathbf{e}_k)^\top)_{ij}}{|\mathcal{S}_k|} - \frac{\sum_{v_i, v_j \in \mathcal{B}} (\tilde{\mathbf{A}} \odot (\mathbf{S}\mathbf{e}_k)(\mathbf{1} - (\mathbf{S}\mathbf{e}_k))^\top)_{ij}}{N - |\mathcal{S}_k|} \right|, \tag{3}$$

where $\mathcal{B}$ denotes the set of nodes within the utilized minibatch. Note that the aforementioned versatile use of $\mathcal{L}$ also applies to $\mathcal{L}_{\text{FairWire}}$, which can directly be used in topology inference tasks. Specifically, for link prediction, the following loss function can be employed in training to combat bias:

$$\mathcal{L}_{lp} = \sum\nolimits_{v_i, v_j \in \mathcal{B}} \mathcal{L}_{CE}\big(\tilde{\mathbf{A}}_{ij}, \mathbf{A}_{ij}\big) + \lambda \mathcal{L}_{\text{FairWire}}(\tilde{\mathbf{A}}, \mathcal{B}), \tag{4}$$

where $\tilde{\mathbf{A}}_{ij}$ denotes the predicted probability by the algorithm for an edge between v_i and v_j, and $\mathcal{L}_{CE}$ is cross-entropy loss. The hyperparameter λ is used to adjust the weight of fairness in training.

5 Fair Graph Generation

Generating synthetic graphs that capture the structural characteristics in real data attracts increasing attention as a promising remedy for scalability (ever-increasing size of real-world graphs) and privacy issues. Especially, sharing real sensitive attributes for fair model training exacerbates the privacy concerns due to the sensitive attribute leakage problem (11). Thus, creating synthetic graphs with generative models becomes instrumental in applications over interconnected systems. In this work, we focus on diffusion models whose success in capturing the original data distribution has been shown for various types of networks (45; 46; 16; 47). Despite the growing interest in these models, their effects on fairness have not yet been investigated, which limits their use in critical real-world decision systems. Motivated by this, in Subsection 5.1, we first empirically analyze the impact of diffusion models on the algorithmic bias by comparing the original and synthetic graphs in terms of different fairness metrics for link prediction. This empirical investigation reveals that the algorithmic bias is amplified while using generative models for graph creation. To resolve this critical issue, we develop FairWire in Subsection 5.2, a fair graph generation framework, which leverages our proposed regularizer $\mathcal{L}_{\text{FairWire}}$ during the training of a diffusion model.

5.1 Diffusion Models and Structural Bias

To evaluate the effect of synthetic graph generation on bias, we first sample 10 different synthetic graphs for each of the 4 real-world networks (see Table 7 in Appendix E and Subsection 6.1 for more details on the datasets). Synthetic graphs are sampled using a diffusion model that is trained following the setup in (47), which is a state-of-the-art algorithm for diffusion-based graph generation. Upon creating graphs, we evaluate them for the link prediction task on the same test set (generated from the real data) and report the corresponding utility (AUC) and fairness performance. Fairness performance is measured via two widely used bias metrics, statistical parity (Δ_{SP}) and equal opportunity (Δ_{EO}) (19) for which lower values indicate better fairness (see Subsection 6.1 for more details on the link prediction model and evaluation metrics). The obtained results are presented in Table 1.

In Table 1, $\mathcal{G}$ denotes the original graphs, and the synthetic graphs are represented by $\tilde{\mathcal{G}}$. Overall, Table 1 shows that graph generation via diffusion models indeed amplifies the already existing bias in the original graphs consistently for all the considered datasets. This brings the potential bias-related issues in synthetic graph creation to light and calls for robust bias mitigation solutions.

Table 1: Comparative results

	Cora			Citeseer		
	Accuracy (%)	Δ_{SP} (%)	Δ_{EO} (%)	Accuracy (%)	Δ_{SP} (%)	Δ_{EO}(%)
$\mathcal{G}$	94.92	27.71	11.53	95.76	29.05	9.53
$\tilde{\mathcal{G}}$	87.29 ± 1.09	35.72 ± 1.74	13.27 ± 0.81	92.19 ± 1.06	37.56 ± 1.29	13.52 ± 0.92
	Amazon Photo			Amazon Computer		
	Accuracy (%)	Δ_{SP} (%)	Δ_{EO} (%)	Accuracy (%)	Δ_{SP} (%)	Δ_{EO}(%)
$\mathcal{G}$	96.91	32.58	8.24	96.14	22.90	4.63
$\tilde{\mathcal{G}}$	94.45 ± 0.21	33.49 ± 0.28	10.01 ± 0.56	94.04 ± 0.26	23.56 ± 0.55	6.23 ± 0.49

5.2 FairWire: A Fair Graph Generation Framework

The proposed fairness regularizer in Subsection 4.2, $\mathcal{L}_{\text{FairWire}}$, can be utilized in two different settings: i) during model training for link prediction, ii) for training a graph generation model in a task-agnostic way. Note that for both cases, a model is trained to predict a probabilistic graph adjacency matrix, $\tilde{\mathbf{A}}$, upon which $\mathcal{L}_{\text{FairWire}}$ can be employed. Both use cases can facilitate several fairness-aware graph-based applications. That said, the bias amplification issue in generative models (also observed from Table 1) makes creating fair graphs via graph generation models of particular interest.

The proposed fair graph generation framework, FairWire, is built upon structured denoising diffusion models for discrete data (51). In the forward diffusion, FairWire employs a Markov process to create noisy graph data samples by independently adding or deleting edges. For denoising, a message-passing neural network (MPNN) is trained to predict the clean graph based on noisy samples by using the guidance of sensitive attributes (and node labels if available). Finally, we sample synthetic graphs with the guidance of synthetic sensitive attributes that are initialized based on their distribution in the original data. If input graph has also node labels, during graph generation, these node labels are sampled based on their distribution conditioned on the sensitive attributes in the original graph. In the sequel, as our main novelty lies in the denoising process, we discuss the training process of FairWire (reverse diffusion process) in more detail, while the forward diffusion and sampling processes are explained in Appendix D. Note that the diffusion process is presented for attributed graphs, where synthetic nodal features $\tilde{\mathbf{X}}$ are also generated. However, the proposed approach can be readily adapted to graphs without nodal features.

Reverse diffusion process: For denoising, we train an MPNN, ϕ_θ parametrized by θ, which is shown to be a scalable solution for the generation of large, attributed graphs (47). Specifically, ϕ_θ inputs a noisy version of the input graph and the original sensitive attributes described by $\mathbf{X}^t, \mathbf{A}^t, \mathbf{S}$ and aims to recover the original nodal features $\mathbf{X}^0$ and graph topology $\mathbf{A}^0$. Here, $\mathbf{A}^0 \in \mathbb{R}^{N\times N\times 2}$ denotes the one-hot representations for the edge labels. Note that the sensitive attributes are used to guide the MPNN to capture the relations between them and graph topology. Therefore, the sensitive attributes are initialized and kept the same during both training and sampling (the original distribution of sensitive attributes is used to initialize them during sampling). For a node v, the message passing at the lth layer can be described as:

$$\mathbf{h}_v^{(t,l+1)} = \sigma\left(\mathbf{W}_{T\to H}^{(l)}\mathbf{h}_t + \mathbf{b}_H^{(l)} + \sum\nolimits_{u\in\mathcal{N}^{(t)}(v)} \frac{1}{\left|\mathcal{N}^{(t)}(v)\right|}\left[\mathbf{h}_u^{(t,l)} \| \mathbf{S}_u^{(l)}\right]\mathbf{W}_{[H,S]\to H}^{(l)}\right), \quad (5)$$

$$\mathbf{S}_v^{(l+1)} = \sigma\left(\mathbf{b}_S^{(l)} + \sum\nolimits_{u\in\mathcal{N}^{(t)}(v)} \frac{1}{\left|\mathcal{N}^{(t)}(v)\right|}\mathbf{S}_u^{(l)}\mathbf{W}_{S\to S}^{(l)}\right), \quad (6)$$

where $\|$ stands for the concatenation operator. In this aggregation, $\mathbf{W}_{T\to H}^{(l)}, \mathbf{W}_{[H,S]\to H}^{(l)}, \mathbf{W}_{S\to S}^{(l)}, \mathbf{b}_H^{(l)}$ and $\mathbf{b}_S^{(l)}$ are all learnable parameters, while $\sigma(\cdot)$ consists of ReLU (52) and LayerNorm (53) layers. In addition, $\mathbf{H}^{(t,0)}$ and $\mathbf{h}_t$ are initialized as hidden representations created for $\mathbf{X}^t$ and time step t via multi-layer perceptrons (MLP), respectively, and $\mathbf{S}^{(0)} = \mathbf{S}$. After creating hidden representations for nodes and their sensitive attributes, final representation for a node v is generated via $\mathbf{h}_v = \mathbf{h}_v^{(t,0)} \| \mathbf{h}_v^{(t,1)} \| \cdots \| \mathbf{S}_v^{(0)} \| \mathbf{S}_v^{(1)} \| \cdots \| \mathbf{h}_t$. Note that when node labels are available, their one-hot representations $\mathbf{Y}$, are also employed in this MPNN in the same way as $\mathbf{S}$ are utilized. Based on these final representations, node attributes, and edge labels are predicted. To create fair graph connections in the synthetic graphs, we regularize the predicted edge probabilities, $\tilde{\mathbf{A}}$, via the designed fairness regularizer $\mathcal{L}_{\text{FairWire}}$. Overall, the training loss of the MPNN follows as:

$$\sum_{v_i \in \mathcal{B}} \mathcal{L}_{CE}\big(\tilde{\mathbf{X}}_{i:}, \mathbf{X}^0_{i:}\big) + \sum_{v_i, v_j \in \mathcal{B}} \mathcal{L}_{CE}\big(\tilde{\mathbf{A}}_{ij:}, \mathbf{A}^0_{ij:}\big) + \lambda \mathcal{L}_{\text{FairWire}}(\tilde{\mathbf{A}}, \mathcal{B}), \tag{7}$$

where λ adjusts the focus on the fairness regularizer.

Remark (Applicability to general generative models): Although the designed regularizer in Subsection 4.2 is embodied in a diffusion-based graph generation framework in Subsection 5.2, $\mathcal{L}_{\text{FairWire}}$ can be utilized in any generative model outputting synthetic graph topologies as a fairness regularizer on the connections, including but not limited to graph autoencoder-based or random walk-based graph generation models.

Remark (Creation of synthetic sensitive attributes): We design a generative framework in Subsection 5.2 that outputs synthetic sensitive attributes whose effect on the connections is reflected by inputting them in the training of MPNN. We emphasize that the creation of these synthetic sensitive attributes also enables the use of existing fairness-aware schemes on the created graphs without leaking the real sensitive attributes.

6 Experiments

6.1 Datasets and Experimental Setup

Datasets. In the experiments, four attributed networks are employed, namely Cora, Citeseer, Amazon Photo and Amazon Computer for link prediction. Cora and Citeseer are widely utilized citation networks, where the articles are nodes and the network topology depicts the citation relationships between these articles (54). Amazon Photo and Amazon Computer are product co-purchase networks, where the nodes are the products and the links are created if two products are often bought together (55). In addition to link prediction, we also evaluate the synthetic graphs on node classification, where the German credit (56) and Pokec-n (9) graphs are employed. For more details on the datasets and their statistics, please see Appendix E.

Experimental Setup. In this section, we first report the performance of $\mathcal{L}_{\text{FairWire}}$ for link prediction. For this task, the area under the curve (AUC) is employed as the utility metric. As fairness metrics, statistical parity and equal opportunity definitions in (19; 37) are used, where $\Delta_{\text{SP}} := |\mathbb{E}_{(v_i,v_j)\sim U_\mathcal{V}\times U_\mathcal{V}}[\tilde{A}_{ij} = 1 \mid s_i = s_j] - \mathbb{E}_{(v_i,v_j)\sim U_\mathcal{V}\times U_\mathcal{V}}[\tilde{A}_{ij} = 1 \mid s_i \neq s_j]|$ and $\Delta_{\text{EO}} := |\mathbb{E}_{(v_i,v_j)\sim U_\mathcal{V}\times U_\mathcal{V}}[\tilde{A}_{ij} = 1 \mid A_{ij} = 1, s_i = s_j] - \mathbb{E}_{(v_i,v_j)\sim U_\mathcal{V}\times U_\mathcal{V}}[\tilde{A}_{ij} = 1 \mid A_{ij} = 1 s_i, \neq s_j]|$. Lower values for Δ_{SP} and Δ_{EO} indicate better fairness performance.

To evaluate the generated synthetic graphs, we use both the link prediction and node classification tasks. Herein, we sample 10 synthetic graphs for each dataset with the trained diffusion models. Afterward, we train link prediction/node classification models (for more details on these models, please see Appendix G) on the sampled graphs, and test these models on the real graphs $\mathcal{G}$ (the test set is the same for all baselines and FairWire). Here, we consider the scenario where there is no access to the real graphs due to privacy concerns, and the models are trained on the synthetic graphs for downstream tasks. To evaluate these synthetic graphs on link prediction, the same utility and fairness metrics as in the link prediction task are used. For node classification, accuracy is employed as the utility measure with $\Delta_{SP} := |P(\hat{y}_j = 1 \mid s_j = 0) - P(\hat{y}_j = 1 \mid s_j = 1)|$, and $\Delta_{EO} := |P(\hat{y}_j = 1 \mid y_j = 1, s_j = 0) - P(\hat{y}_j = 1 \mid y_j = 1, s_j = 1)|$ being the fairness metrics.

For more details on the training of link prediction, node classification, diffusion models, and the hyperparameter selection for FairWire and baselines, see Appendix G. A sensitivity analysis is also provided in Appendix H for the effect of hyperparameter λ in (4) and in (7). Note that the performance of the generative algorithms is generally reported in terms of the distances between the statistics of real data and the synthetic ones, instead of the fairness performance. For completeness, we report the distance metrics for node degree distribution and clustering coefficient distribution in Appendix F.

Baselines. For link prediction, fairness-aware baselines include adversarial regularization (9), FairDrop (37), and FairAdj (19). For graph generation, FairGen (49), is the only existing fairness-aware baseline designed for node classification. For a comprehensive evaluation, we also employ adversarial regularization (9) and FairAdj (19) as in-processing and post-processing fairness-aware strategies within the generative model. For more details on the baselines, please see Appendix G.

Table 2: Comparative link prediction results.

	Cora			Citeseer		
	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
GNN	$\mathbf{94.43} \pm 0.74$	27.01 ± 1.38	9.11 ± 1.43	$\mathbf{96.16} \pm 0.28$	27.40 ± 1.24	7.37 ± 1.33
Adversarial	87.77 ± 1.64	24.33 ± 6.36	2.96 ± 2.24	93.53 ± 0.51	13.97 ± 10.36	6.52 ± 4.68
FairDrop	94.10 ± 0.81	7.86 ± 4.30	4.05 ± 0.32	95.92 ± 0.42	13.77 ± 6.15	5.60 ± 1.85
FairAdj	82.01 ± 1.56	14.76 ± 0.89	7.35 ± 1.24	84.76 ± 1.08	16.00 ± 11.93	5.91 ± 3.42
$\mathcal{L}_{\text{FairWire}}$	92.18 ± 1.03	$\mathbf{4.76} \pm 0.24$	$\mathbf{2.05} \pm 0.37$	96.00 ± 0.23	$\mathbf{8.62} \pm 0.80$	$\mathbf{1.29} \pm 0.68$
	Amazon Photo			Amazon Computer		
	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
GNN	97.01 ± 0.26	32.65 ± 0.95	8.01 ± 0.52	$\mathbf{96.13} \pm 0.06$	23.70 ± 0.79	5.51 ± 0.79
Adversarial	96.17 ± 0.09	29.57 ± 0.91	8.03 ± 1.05	95.64 ± 0.12	22.72 ± 1.11	4.71 ± 0.99
FairDrop	95.36 ± 0.33	28.63 ± 1.39	8.49 ± 1.54	95.61 ± 0.13	21.30 ± 0.59	4.30 ± 0.77
$\mathcal{L}_{\text{FairWire}}(\lambda = c)$	$\mathbf{97.25} \pm 0.11$	$\mathbf{27.75} \pm 0.52$	$\mathbf{7.11} \pm 0.41$	96.05 ± 0.05	$\mathbf{20.44} \pm 0.44$	$\mathbf{4.24} \pm 0.51$
$\mathcal{L}_{\text{FairWire}}(\lambda = 5c)$	94.85 ± 0.32	$\mathbf{24.61} \pm 0.96$	$\mathbf{6.24} \pm 1.22$	93.86 ± 0.23	$\mathbf{15.36} \pm 1.02$	$\mathbf{0.17} \pm 0.21$

Table 3: Comparative results for graph generation on Link Prediction.

	Cora			Citeseer		
	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
$\mathcal{G}$	94.92	27.71	11.53	95.76	29.05	9.53
$\tilde{\mathcal{G}}$	$\mathbf{87.29} \pm 1.09$	35.72 ± 1.74	13.27 ± 0.81	$\mathbf{92.19} \pm 1.06$	37.56 ± 1.29	13.52 ± 0.92
FairAdj	82.13 ± 1.07	15.47 ± 2.39	6.26 ± 2.05	82.67 ± 2.78	$\mathbf{15.45} \pm 2.68$	7.98 ± 1.47
Adversarial	83.66 ± 5.64	16.35 ± 9.80	7.82 ± 5.84	89.59 ± 2.70	24.20 ± 5.82	10.34 ± 1.66
FairWire	86.49 ± 2.79	$\mathbf{12.91} \pm 6.35$	$\mathbf{4.31} \pm 3.59$	91.27 ± 2.78	18.35 ± 6.91	$\mathbf{7.80} \pm 2.76$

6.2 Link Prediction Results

Comparative results for the link prediction task are presented in Table 2, where we consider the setting $\mathcal{L}_{\text{FairWire}}$ is employed as a fairness regularizer while training a GNN model for link prediction. The natural baseline here is to employ the same GNN model without any fairness interventions, which is denoted by GNN in Table 2. Note that in Table 2, c equals to 0.01 for Amazon Photo and the results are presented for $c = 0.1$ for Amazon Computer.

The results in Table 2 demonstrate that employing $\mathcal{L}_{\text{FairWire}}$ as a fairness regularizer leads to better fairness measures compared to the naive baseline, while also providing similar utility. Specifically, the proposed regularizer is observed to improve both fairness metrics, Δ_{SP} and Δ_{EO}, with improvements ranging from 20% to 80% for every evaluated dataset compared to the natural baseline, GNN. Furthermore, the obtained results show that $\mathcal{L}_{\text{FairWire}}$ also outperforms the fairness-aware baselines Adversarial (9), FairDrop (37), and FairAdj (19) in both fairness metrics. For certain datasets (e.g., Amazon Photo, Amazon Computer), we report the results of FairWire for different values of λ to illustrate the trade-off between fairness and utility and to show that FairWire leads to a better trade-off compared to the other fairness-aware baselines. Note that we could not include the results of FairAdj over the product networks (i.e., Amazon Photo, Amazon Computer) due to its substantial memory use during its optimization process, which led to out-of-memory errors for the infrastructure we use. In addition to the improved fairness performance, it can be observed that the employment of $\mathcal{L}_{\text{FairWire}}$ generally results in the lowest standard deviation values for fairness metrics, which demonstrates the stability of the proposed strategy for bias mitigation. Overall, the results corroborate the effectiveness of $\mathcal{L}_{\text{FairWire}}$ in enhancing fairness while also providing similar utility compared to the state-of-the-art fairness-aware baselines.

6.3 Results for Graph Generation

Comparative results for graph generation are presented in Tables 3 and 4, where the link prediction and node classification tasks are used to evaluate the synthetic graphs, respectively. In these tables, $\mathcal{G}$ represents the original data, and $\tilde{\mathcal{G}}$ stands for the synthetic graphs generated by the fairness-agnostic GraphMaker (47). Overall, Tables 3 and 4 show that FairWire improves fairness metrics compared to $\tilde{\mathcal{G}}$, fairness-agnostic synthetic graphs created via diffusion, without a significant utility loss for

Table 4: Comparative results for graph generation on Node Classification.

	German			Pokec-n		
	Acc (%)	Δ_{SP} (%)	Δ_{EO} (%)	Acc (%)	Δ_{SP} (%)	Δ_{EO}(%)
$\mathcal{G}$	70.00	2.13	1.78	68.73	8.58	9.68
FairGen	**73.60**	28.71	15.34	51.73	0.00	0.00
$\tilde{\mathcal{G}}$	68.92 ± 2.37	2.61 ± 5.83	2.29 ± 5.06	66.19 ± 2.05	3.63 ± 2.58	2.66 ± 2.50
FairAdj	70.08 ± 1.08	2.17 ± 4.49	1.11 ± 2.24	-	-	-
Adversarial	70.00 ± 0.62	1.57 ± 2.70	1.34 ± 2.86	$\mathbf{69.36} \pm 0.70$	2.16 ± 1.73	2.73 ± 2.01
FairWire	69.76 ± 0.51	$\mathbf{0.63} \pm 1.53$	$\mathbf{0.30} \pm 0.61$	68.23 ± 0.45	$\mathbf{1.91} \pm 0.92$	$\mathbf{1.35} \pm 0.92$

both link prediction and node classification. Specifically, FairWire can achieve improvements in both Δ_{SP} and Δ_{EO} ranging from 25% to 90% for all datasets compared to $\tilde{\mathcal{G}}$ with similar utility.

Note that, for link prediction, the fairness improvement reported for FairAdj in Table 3 is accompanied by a significant utility drop. Specifically, for a larger λ value (i.e., $\lambda = 1$), FairWire can provide better fairness measures ($\Delta_{SP} = 7.01 \pm 6.25$ and $\Delta_{EO} = 3.06 \pm 2.95$) on Citeseer with a similar accuracy (83.47 ± 7.79) to FairAdj. Thus, the results in Table 3 demonstrate that FairWire provides a better utility/fairness trade-off compared to fairness-aware baselines on all evaluated datasets.

In Table 4, similar to the link prediction experiments (Table 2), the results of FairAdj for the Pokec-n network could not be obtained due to computational limitations. For FairGen (49), we directly input the synthetic graph output by the algorithm to the node classification model we train, thus the results are obtained for a single synthetic graph. A possible explanation for the better accuracy of FairGen on the German dataset is that the algorithm is observed to output a denser synthetic network, which might be useful for the utility. It is observed that the synthetic graph output by FairGen for Pokec-n was not informative enough for the node classification task (we provide the codes for the FairGen algorithm in our supplementary material for the reproducibility of these results.) All in all, the results in Table 4 signify that the superior performance of FairWire in terms of fairness/utility trade-off can also be observed for node classification, which validates the efficacy of FairWire in creating fair synthetic graphs that also capture the real data distribution.

6.4 Visualization of Synthetic Graphs

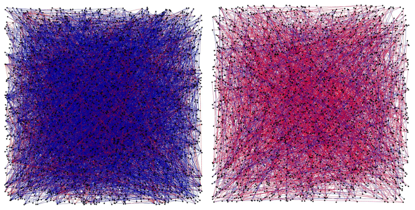
Figure 1: Distribution of the intra-edges (blue) and inter-edges (red) in the synthetic graphs created for Cora dataset by GraphMaker (47) (left) and FairWire (right).

Our analysis in Subsection 4.1 reveals that the ratio of intra- (edges connecting the same sensitive group) and inter-edges (edges between different sensitive groups) is a factor contributing to the structural bias. Specifically, the bias factor α_1 is minimized when $\frac{p_k^{\omega}}{p_k^{\chi}} = \frac{|\mathcal{S}_k|}{N-|\mathcal{S}_k|}, \forall k$, where p_k^{ω} and p_k^{χ} are the expected number of intra-and inter-edges for the nodes in $\mathcal{S}_k$. This finding suggests that for a graph with multiple (> 2) sensitive groups, given the sizes of sensitive groups are not catastrophically unbalanced, the number of inter-edges (related to p_k^{χ}) should be larger than the number of intra-edges (related to p_k^{ω}) to alleviate structural bias (i.e., $|\mathcal{S}_k| \leq N - |\mathcal{S}_k|$). However, for graphs encountered in several domains, the number of intra-edges is significantly larger than the number of inter-edges, due to the homophily principle (10). Motivated by this, in Figure 1, we visualize the distributions of intra- and inter-edges in synthetic graphs created by i) a fairness-agnostic strategy, GraphMaker (47), and ii) FairWire, for Cora. In Figure 1, intra- and inter-edges are colored with blue and red, respectively. Figure 1 reveals that the graph created by GraphMaker (47) predominantly consists of intra-edges, leading to the structural bias reflected in Table 3. In contrast, FairWire exhibits a remarkable balancing effect, which provides a potential explanation for the improvement in fairness.

7 Conclusion

This study focuses on the investigation and mitigation of structural bias for both real and synthetic graphs, where a novel fairness regularizer, $\mathcal{L}_{\text{FairWire}}$, is designed to alleviate the effects of bias factors identified in a developed theoretical bias analysis. Furthermore, the proposed fairness regularizer is leveraged in a fair graph generation framework, FairWire, which alleviates the bias amplification observed in graph generative models. Experimental results corroborate the effectiveness of the proposed tools in bias mitigation for both real and synthetic graphs.

Limitations: This paper considers the setting where sensitive attributes are available during model training, which might limit its use for certain real-world applications. Thus, one future direction of this work would be to consider the partial availability of these sensitive attributes in the input graph data. Furthermore, although we showed that real-world graphs typically satisfy the assumptions in Subsection 4.1, another possible future work we consider is deriving a theoretical bias analysis without the dependency on these assumptions.

Acknowledgement

Work in this paper is supported by NSF ECCS 2412484.

References

[1] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, and K. Murphy, “Machine learning on graphs: A model and comprehensive taxonomy,” *The Journal of Machine Learning Research*, vol. 23, no. 1, pp. 3840–3903, 2022.

[2] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. International Conference on Learning Representations (ICLR)*, Apr. 2017.

[3] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *International Conference on Learning Representations*, 2018.

[4] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, “How powerful are graph neural networks?” in *International Conference on Learning Representations*, 2018.

[5] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang, “Self-supervised multi-channel hypergraph convolutional network for social recommendation,” in *Proceedings of the web conference 2021*, 2021, pp. 413–424.

[6] M. Li and Z. Zhu, “Spatial-temporal fusion graph neural networks for traffic flow forecasting,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4189–4196.

[7] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, “A survey on bias and fairness in machine learning,” *ACM computing surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[8] D. Pessach and E. Shmueli, “A review on fairness in machine learning,” *ACM Computing Surveys (CSUR)*, vol. 55, no. 3, pp. 1–44, 2022.

[9] E. Dai and S. 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*, 2021, pp. 680–688.

[10] B. Hofstra, R. Corten, F. Van Tubergen, and N. B. Ellison, “Sources of segregation in social networks: A novel approach using facebook,” *American Sociological Review*, vol. 82, no. 3, pp. 625–656, 2017.

[11] E. Dai and S. Wang, “Learning fair graph neural networks with limited and private sensitive attribute information,” *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[12] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, “Netgan: Generating graphs via random walks,” in *International conference on machine learning*. PMLR, 2018, pp. 610–619.

[13] L. Akoglu, M. McGlohon, and C. Faloutsos, "Rtm: Laws and a recursive generator for weighted time-evolving graphs," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 701–706.

[14] N. Goyal, H. V. Jain, and S. Ranu, "Graphgen: A scalable approach to domain-agnostic labeled graph generation," in *Proceedings of The Web Conference 2020*, 2020, pp. 1253–1263.

[15] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. W. Battaglia, "Learning deep generative models of graphs. corr abs/1803.03324 (2018)," 1803.

[16] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "Digress: Discrete denoising diffusion for graph generation," in *The Eleventh International Conference on Learning Representations*, 2022.

[17] K. Schwarz, Y. Liao, and A. Geiger, "On the frequency bias of generative models," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 126–18 136, 2021.

[18] S. Zhao, H. Ren, A. Yuan, J. Song, N. Goodman, and S. Ermon, "Bias and generalization in deep generative models: An empirical study," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[19] P. Li, Y. Wang, H. Zhao, P. Hong, and H. Liu, "On dyadic fairness: Exploring and mitigating bias in graph connections," in *International Conference on Learning Representations*, 2021.

[20] Y. Dong, J. Ma, S. Wang, C. Chen, and J. Li, "Fairness in graph mining: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[21] M. Choudhary, C. Laclau, and C. Largeron, "A survey on fairness for machine learning on graphs," *arXiv preprint arXiv:2205.05396*, 2022.

[22] Y. Dong, O. D. Kose, Y. Shen, and J. Li, "Fairness in graph machine learning: Recent advances and future prospectives," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 5794–5795.

[23] T. Rahman, B. Surma, M. Backes, and Y. Zhang, "Fairwalk: towards fair graph embedding," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 3289–3295.

[24] A. Bose and W. Hamilton, "Compositional fairness constraints for graph embeddings," in *International Conference on Machine Learning*. PMLR, 2019, pp. 715–724.

[25] J. Palowitch and B. Perozzi, "Debiasing graph representations via metadata-orthogonal training," in *IEEE International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2020, pp. 435–442.

[26] Y. Dong, J. Kang, H. Tong, and J. Li, "Individual fairness for graph neural networks: A ranking based approach," in *Proc ACM Conference on Knowledge Discovery & Data Mining (SIGKDD)*, 2021, pp. 300–310.

[27] W. Song, Y. Dong, N. Liu, and J. Li, "Guide: Group equality informed individual fairness in graph neural networks," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 1625–1634.

[28] C. Agarwal, H. Lakkaraju, and M. Zitnik, "Towards a unified framework for fair and stable graph representation learning," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 2114–2124.

[29] J. Ma, R. Guo, M. Wan, L. Yang, A. Zhang, and J. Li, "Learning fair node representations with graph counterfactual fairness," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 695–703.

[30] Z. Guo, J. Li, T. Xiao, Y. Ma, and S. Wang, "Towards fair graph neural networks via graph counterfactual," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 669–678.

[31] J. Fisher, A. Mittal, D. Palfrey, and C. Christodoulopoulos, “Debiasing knowledge graph embeddings,” in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 7332–7345.

[32] D. Guo, C. Wang, B. Wang, and H. Zha, “Learning fair representations via distance correlation minimization,” *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2022.

[33] M. Buyl and T. De Bie, “Debayes: a bayesian method for debiasing network embeddings,” in *International Conference on Machine Learning (ICML)*. PMLR, 2020, pp. 1220–1229.

[34] Y. Dong, N. Liu, B. Jalaian, and J. Li, “Edits: Modeling and mitigating data bias for graph neural networks,” in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1259–1269.

[35] O. D. Kose and Y. Shen, “Fair contrastive learning on graphs,” *IEEE Transactions on Signal and Processing over Networks*, vol. 8, pp. 475–488, 2022.

[36] ——, “Demystifying and mitigating bias for node representation learning,” *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[37] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini, “Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 3, pp. 344–354, 2021.

[38] C. Laclau, I. Redko, M. Choudhary, and C. Largeron, “All of the fairness for edge prediction with optimal transport,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 1774–1782.

[39] M. Buyl and T. D. Bie, “The KL-divergence between a graph model and its fair I-projection as a fairness regularizer,” in *Joint European Conf. on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 351–366.

[40] X. Ying and X. Wu, “Graph generation with prescribed feature constraints,” in *Proceedings of the 2009 SIAM International Conference on Data Mining*. SIAM, 2009, pp. 966–977.

[41] D. Chakrabarti and C. Faloutsos, “Graph mining: Laws, generators, and algorithms,” *ACM computing surveys (CSUR)*, vol. 38, no. 1, pp. 2–es, 2006.

[42] L. Rendsburg, H. Heidrich, and U. Von Luxburg, “Netgan without gan: From random walks to low-rank approximations,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 8073–8082.

[43] M. Simonovsky and N. Komodakis, “Graphvae: Towards generation of small graphs using variational autoencoders,” in *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*. Springer, 2018, pp. 412–422.

[44] J. Liu, A. Kumar, J. Ba, J. Kiros, and K. Swersky, “Graph normalizing flows,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[45] C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon, “Permutation invariant graph generation via score-based generative modeling,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4474–4484.

[46] J. Jo, S. Lee, and S. J. Hwang, “Score-based generative modeling of graphs via the system of stochastic differential equations,” in *International Conference on Machine Learning*. PMLR, 2022, pp. 10 362–10 383.

[47] M. Li, E. Kreačić, V. K. Potluru, and P. Li, “Graphmaker: Can diffusion models generate large attributed graphs?” *arXiv preprint arXiv:2310.13833*, 2023.

[48] X. Chen, J. He, X. Han, and L.-P. Liu, “Efficient and degree-guided graph generation via discrete diffusion modeling,” *arXiv preprint arXiv:2305.04111*, 2023.

[49] L. Zheng, D. Zhou, H. Tong, J. Xu, Y. Zhu, and J. He, “Fairgen: Towards fair graph generation,” *arXiv preprint arXiv:2303.17743*, 2023.

[50] B. Hofstra, R. Corten, F. Van Tubergen, and N. B. Ellison, "Sources of segregation in social networks: A novel approach using facebook," *American Sociological Review*, vol. 82, no. 3, pp. 625–656, May 2017.

[51] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. Van Den Berg, "Structured denoising diffusion models in discrete state-spaces," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 981–17 993, 2021.

[52] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 2146–2153.

[53] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[54] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.

[55] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," *arXiv preprint arXiv:1811.05868*, 2018.

[56] D. Dua, C. Graff *et al.*, "Uci machine learning repository," 2017.

[57] O. D. Kose and Y. Shen, "Fairgat: Fairness-aware graph attention networks," *arXiv preprint arXiv:2303.14591*, 2023.

[58] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans, "Cascaded diffusion models for high fidelity image generation," *The Journal of Machine Learning Research*, vol. 23, no. 1, pp. 2249–2281, 2022.

[59] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "Digress: Discrete denoising diffusion for graph generation," in *Proceedings of the 11th International Conference on Learning Representations*, 2023.

[60] P. Erdos and A. Renyi, "On random graphs i," *Publ. math. debrecen*, vol. 6, no. 290-297, p. 18, 1959.

[61] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social networks*, vol. 5, no. 2, pp. 109–137, 1983.

[62] T. Chen, W. Zhang, Q. Lu, K. Chen, Z. Zheng, and Y. Yu, "Svdfeature: a toolkit for feature-based collaborative filtering," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 3619–3622, 2012.

[63] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[64] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, May 2010, pp. 249–256.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[66] J. Gasteiger, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *International Conference on Learning Representations*, 2018.

[67] M. Y. Wang, "Deep graph library: Towards efficient and scalable deep learning on graphs," in *ICLR workshop on representation learning on graphs and manifolds*, 2019.

[68] L. Takac and M. Zabovsky, "Data analysis in public social networks," in *International Scientific Conference and International Workshop. 'Present Day Trends of Innovations'*, vol. 1, no. 6, May 2012.

A Assumptions and Real-World Graphs

In order to show that the assumptions made in Subsection 4.1 are typically valid for real-world graphs, here we present the exact values of the terms within Assumptions 2 and 3 for the datasets we use. Specifically, we make the following assumptions:

A2: $\frac{\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\omega}]}{|\mathcal{S}_k|} \geq \frac{\mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}]}{N-|\mathcal{S}_k|}, \forall v_i \in \mathcal{S}_k, \forall k \in \{1, \cdots, K\}$,
A3: $\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} \gg \mathbb{E}_{\tilde{\mathbf{A}}}[d_i^{\chi}], \forall v_i \in \mathcal{V}$.

First, for Assumption 2, we obtained the real values of the terms $l_k := \frac{\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[d_i^{\omega}]}{|\mathcal{S}_k|}$ and $r_k := \frac{\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[d_i^{\chi}]}{N-|\mathcal{S}_k|}$, where we want $l_k \geq r_k \forall k \in \{1, \cdots, K\}$. For all different sensitive groups, these values are presented in Table 5 for all the used datasets.

Table 5: Validity of Assumption 2 for real-world graphs.

Cora	l_0	l_1	l_2	l_3	l_4	l_5	l_6			
	0.0087	0.0174	0.0095	0.0035	0.0073	0.0094	0.0156			
	r_0	r_1	r_2	r_3	r_4	r_5	r_6			
	0.0024	0.0023	0.0021	0.0021	0.0019	0.0019	0.018			
Citeseer	l_0	l_1	l_2	l_3	l_4	l_5				
	0.0028	0.0026	0.0047	0.0026	0.0039	0.0034				
	r_0	r_1	r_2	r_3	r_4	r_5				
	0.0011	0.0012	0.0018	0.0011	0.0013	0.0009				
Amazon Photo	l_0	l_1	l_2	l_3	l_4	l_5	l_6	l_7		
	0.0874	0.0094	0.0428	0.0214	0.0260	0.0262	0.0202	0.0484		
	r_0	r_1	r_2	r_3	r_4	r_5	r_6	r_7		
	0.0055	0.0037	0.0045	0.0057	0.0054	0.0032	0.0088	0.0088		
Amazon Computer	l_0	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
	0.0381	0.0078	0.0178	0.0281	0.0080	0.0891	0.0376	0.0219	0.0096	0.0696
	r_0	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9
	0.0025	0.0033	0.0026	0.0034	0.0064	0.0024	0.0040	0.0017	0.0036	0.0024

For Assumption 3, we present the real values of the terms $\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij}$ and $\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{V}}}[d_i^{\chi}]$, where we want $\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} \gg \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{V}}}[d_i^{\chi}]$. For all real-world datasets we use, these values are reported in Table 6.

Table 6: Validity of Assumption 3 for real-world graphs.

	Cora	Citeseer	Amazon Photo	Amazon Computer
$\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij}$	10556	9104	238162	491722
$\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{V}}}[d_i^{\chi}]$	1.48	1.45	10.76	15.93

Overall, the results presented in both Tables 5 and 6 demonstrate that the assumptions made for the theoretical bias analysis in Section 4.1 are valid for the real-world graphs we are using. This supports that our analysis is applicable to most of the real-world data and settings.

B Proof of Theorem 1

Here, without loss of generality, we will focus on the lth GNN layer, where the input representations are represented by $\mathbf{H}^l$ and output representations are denoted $\mathbf{H}^{l+1}$. The considered disparity measure follows as:

$$\delta_k^{(l+1)} := \left\| \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i^{l+1} \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{(\mathcal{V}-\mathcal{S}_k)}}[\mathbf{h}_i^{l+1} \mid s_i \neq k] \right\|_2 . \tag{8}$$

Let's re-write the disparity measure $\delta_k^{(l+1)}$ by using definitions $\mathbf{c}_i^{l+1} := \mathbf{W}^l \mathbf{h}_i^l$, and $\mathbb{E}_{\tilde{\mathbf{A}}}[\mathbf{h}_i^{l+1}] = \sigma(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j^{l+1})$.

$$\begin{aligned}\delta_k^{(l+1)} &:= \left\| \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i^{l+1} \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{(\mathcal{V}-\mathcal{S}_k)}}[\mathbf{h}_i^{l+1} \mid s_i \neq k] \right\|_2, \\ &= \left\| \frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sigma\left(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right) - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sigma\left(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right) \right\|_2. \end{aligned} \tag{9}$$

Using Lemma A.1. in (57), it can be shown that $\delta_k^{(l+1)}$ can be upper-bounded as:

$$\begin{aligned}\delta_k^{(l+1)} &= \left\| \frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sigma\left(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right) - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sigma\left(\sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right) \right\|_2, \\ &\leq L \left(\left\| \frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right\|_2 + 2\sqrt{N} \Delta_z \right). \end{aligned} \tag{10}$$

Then, we can divide the sums over all nodes into two: nodes in $\mathcal{S}_k$ and the remaining ones.

$$\begin{aligned}\delta_k^{(l+1)} &\leq L \left(\left\| \frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sum_{v_j \in \mathcal{V}} \tilde{A}_{ij} \mathbf{c}_j \right\|_2 + 2\sqrt{N} \Delta_z \right), \\ &= L \Bigg(\Bigg\| \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sum_{v_j \in \mathcal{S}_k} \tilde{A}_{ij} \mathbf{c}_j + \frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \sum_{v_j \notin \mathcal{S}_k} \tilde{A}_{ij} \mathbf{c}_j \right) \\ &\quad - \left(\frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sum_{v_j \in \mathcal{S}_k} \tilde{A}_{ij} \mathbf{c}_j + \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \sum_{v_j \notin \mathcal{S}_k} \tilde{A}_{ij} \mathbf{c}_j \right) \Bigg\|_2 + 2\sqrt{N} \Delta_z \Bigg) \\ &= L \Bigg(\Bigg\| \sum_{v_j \in \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \mathbf{c}_j \\ &\quad + \sum_{v_j \notin \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \mathbf{c}_j \Bigg\|_2 + 2\sqrt{N} \Delta_z \Bigg) \end{aligned} \tag{11}$$

Then, by triangle inequality, it follows that:

$$\begin{aligned}\delta_k^{(l+1)} &\leq L \Bigg(\left\| \sum_{v_j \in \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \mathbf{c}_j \right\|_2 \\ &\quad + \left\| \sum_{v_j \notin \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \mathbf{c}_j \right\|_2 + 2\sqrt{N} \Delta_z \Bigg). \end{aligned} \tag{12}$$

Assumption 2 in Subsection 4.1 ensures that $\left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N-|\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \geq 0, \forall v_j \in \mathcal{S}_k$. Furthermore, third assumption presented in Subsection 4.1 guarantees that $\left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N-|\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \leq 0, \forall v_j \notin \mathcal{S}_k$. Utilizing Assumption 1 in Subsection 4.1, $\|\mathbf{c}_i\|_\infty \leq \delta, \forall v_i \in \mathcal{V}$, the following upper bound can be derived.

$$\begin{aligned}\delta_k^{l+1} &\leq L \Bigg(\left\| \sum_{v_j \in \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 \\ &\quad + \left\| \sum_{v_j \notin \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 + 2\sqrt{N} \Delta_z \Bigg), \end{aligned} \tag{13}$$

where $\boldsymbol{\delta}_{F^{(l+1)}}$ stands for an $F^{(l+1)}$ dimensional vector with all elements being equal to δ. Then, by utilizing the definitions $p_k^{\chi} := \sum_{v_i \in \mathcal{S}_k, v_j \notin \mathcal{S}_k} \tilde{A}_{i,j}, p_k^{\omega} := \sum_{v_i \in \mathcal{S}_k, v_j \in \mathcal{S}_k} \tilde{A}_{i,j}$, the upper bound in (13) can be rewritten as:

$$\begin{aligned}
\delta_k^{(l+1)} \leq L\Bigg(& \left\| \sum_{v_j \in \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 \\
& + \left\| \sum_{v_j \notin \mathcal{S}_k} \left(\frac{1}{|\mathcal{S}_k|} \sum_{v_i \in \mathcal{S}_k} \tilde{A}_{ij} - \frac{1}{N - |\mathcal{S}_k|} \sum_{v_i \notin \mathcal{S}_k} \tilde{A}_{ij} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 + 2\sqrt{N}\Delta_z \Bigg), \\
& = L \left(\left\| \left(\frac{p_k^{\omega}}{|\mathcal{S}_k|} - \frac{p_k^{\chi}}{N - |\mathcal{S}_k|} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 + \left\| \left(\frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} - p_k^{\omega} - 2p_k^{\chi}}{N - |\mathcal{S}_k|} - \frac{p_k^{\chi}}{|\mathcal{S}_k|} \right) \boldsymbol{\delta}_{F^{(l+1)}} \right\|_2 + 2\sqrt{N}\Delta_z \right) \\
& = L \left(\left| \frac{p_k^{\omega}}{|\mathcal{S}_k|} - \frac{p_k^{\chi}}{N - |\mathcal{S}_k|} \right| \|\boldsymbol{\delta}_{F^{(l+1)}}\|_2 + \left| \frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} - p_k^{\omega} - 2p_k^{\chi}}{N - |\mathcal{S}_k|} - \frac{p_k^{\chi}}{|\mathcal{S}_k|} \right| \|\boldsymbol{\delta}_{F^{(l+1)}}\|_2 + 2\sqrt{N}\Delta_z \right).
\end{aligned} \tag{14}$$

The final result follows from the inequality, $\|\boldsymbol{\delta}_{F^{(l+1)}}\|_2 \leq \delta\sqrt{F^{(l+1)}}$:

$$\begin{aligned}
\delta_k^{(l+1)} & := \left\| \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i^{l+1} \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{(\mathcal{V} - \mathcal{S}_k)}}[\mathbf{h}_i^{l+1} \mid s_i \neq k] \right\|_2, \\
& \leq L \left(\delta\sqrt{F^{(l+1)}} \left(\left| \frac{p_k^{\omega}}{|\mathcal{S}_k|} - \frac{p_k^{\chi}}{N - |\mathcal{S}_k|} \right| + \left| \frac{\sum_{v_i, v_j \in \mathcal{V}} \tilde{A}_{ij} - p_k^{\omega} - 2p_k^{\chi}}{N - |\mathcal{S}_k|} - \frac{p_k^{\chi}}{|\mathcal{S}_k|} \right| \right) + 2\sqrt{N}\Delta_z \right),
\end{aligned} \tag{15}$$

which concludes the proof.

C Proof of Proposition 1

Statistical parity for link prediction is defined as (19):

$$\Delta_{\mathrm{SP}} := |\mathbb{E}_{(v_i, v_j) \sim U_{\mathcal{V}} \times U_{\mathcal{V}}}[g(v_i, v_j) \mid s_i = s_j] - \mathbb{E}_{(v_i, v_j) \sim U_{\mathcal{V}} \times U_{\mathcal{V}}}[g(v_i, v_j) \mid s_i \neq s_j]|. \tag{16}$$

By considering each sensitive group explicitly, statistical parity can also be written as:

$$\Delta_{\mathrm{SP}} := \left| \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} \big(\mathbb{E}_{\tilde{\mathbf{A}}}[g(v_i, v_j) \mid s_i = s_j, s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}}[g(v_i, v_j) \mid s_i \neq s_j, s_i = k] \big) \right|. \tag{17}$$

Define $\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i \mid s_i = k] := \mathbf{p}_k$ and $\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{(\mathcal{V} - \mathcal{S}_k)}}[\mathbf{h}_i \mid s_i \neq k] := \mathbf{q}_k$. We further assume that $\|\mathbf{h}_i\|_2 \leq q, \forall v_i \in \mathcal{V}$ and it holds that $\|\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_j \sim U_{(\mathcal{V} - \mathcal{S}_k)}}[\mathbf{h}_j \mid s_j \neq k]\|_2 \leq \delta_k^{max}$. Using the definitions for $\mathbf{p}_k$ and $\mathbf{q}_k$ and link prediction model $g(v_i, v_j) := \mathbf{h}_i^{\top} \Sigma \mathbf{h}_j$, Δ_{SP} can be reformulated as:

$$\begin{aligned}
\Delta_{\mathrm{SP}} & := \left| \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} \big(\mathbf{p}_k^{\top} \Sigma \mathbf{p}_k - \mathbf{p}_k^{\top} \Sigma \mathbf{q}_k \big) \right|, \\
& = \left| \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} \big(\mathbf{p}_k^{\top} \Sigma (\mathbf{p}_k - \mathbf{q}_k) \big) \right|.
\end{aligned} \tag{18}$$

By triangle inequality, it follows that

$$\Delta_{\mathrm{SP}} \leq \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} \left| \big(\mathbf{p}_k^{\top} \Sigma (\mathbf{p}_k - \mathbf{q}_k) \big) \right|. \tag{19}$$

Finally, by using Cauchy-Schwarz inequality and the assumption $\|\mathbf{h}_i\|_2 \leq q, \forall v_i \in \mathcal{V}$, we can conclude that

$$\begin{aligned} \Delta_{\text{SP}} &\leq \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} q \|\Sigma\|_2 \left\| (\mathbf{p}_k - \mathbf{q}_k) \right\|_2, \\ \Delta_{\text{SP}} &\leq \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{N} q \|\Sigma\|_2 \delta_k^{max}, \end{aligned} \tag{20}$$

where the final inequality follows from the assumption that $\|\mathbb{E}_{\tilde{\mathbf{A}}, v_i \sim U_{\mathcal{S}_k}}[\mathbf{h}_i \mid s_i = k] - \mathbb{E}_{\tilde{\mathbf{A}}, v_j \sim U_{(\mathcal{V}-\mathcal{S}_k)}}[\mathbf{h}_j \mid s_j \neq k]\|_2 \leq \delta_k^{max}$.

D Diffusion Model

Diffusion Models for Graph Generation. Briefly, diffusion models are composed of two main elements: a noise model q, and a denoising neural network ϕ_θ. The noise model q progressively corrupts data to create a sequence of increasingly noisy data points. Inspired by the success of Gaussian noise for diffusion-based image generation (58), the earlier diffusion-based graph generation models employ Gaussian noise to create noisy graph data (45; 46). However, such a noise model cannot properly capture the structural properties of discrete graph connections. Motivated by this, a discrete noise model is introduced in (51). The discrete noise model for graph structure is typically applied in the form of edge deletion and additions (16; 47). After creating noisy data, a denoising network ϕ_θ is trained to invert this process by predicting the original graph structure $\mathbf{A}$ from the noisy samples. While different neural network structures are examined as candidates for the denoising network, message-passing neural networks are shown to be a scalable solution for the creation of medium- to large-scale graphs (47).

Forward diffusion process: introduced in (51), we employ a Markov process herein to add noise to the input graph structure in the form of edge additions or deletions. These edge modifications can be executed by modeling the existence/non-existence of an edge as the edge class labels, where we have 2 classes, and applying a transition matrix that switches the labels with a certain probability. Then, given $\mathbf{A}^0 \in \mathbb{R}^{N \times N \times 2}$ denotes the one-hot representations for the edge labels, the noise model can be described by the transition matrices $\mathbf{Q}_A^t \in \mathbb{R}^{2\times 2}$ for $t = 1, \cdots, T$, where $q(\mathbf{A}^t \mid \mathbf{A}^{t-1}) = \mathbf{A}^{t-1}\mathbf{Q}_A^t$, $q(\mathbf{A}^t \mid \mathbf{A}^0) = \mathbf{A}^0 \bar{\mathbf{Q}}_A^t$, and $\bar{\mathbf{Q}}_A^t = \mathbf{Q}_A^1 \mathbf{Q}_A^2 \cdots \mathbf{Q}_A^t$. For a uniform transition model, (59) proves that the empirical data distribution (the probability for the existence of an edge) is the optimal prior distribution. Following this finding, we specifically employ the transition matrix:

$$\mathbf{Q}_A^t = \alpha^t \mathbf{I} + (1 - \alpha^t) \mathbf{1} \mathbf{m}_E^\top, \tag{21}$$

where $\mathbf{I} \in \mathbb{R}^{2\times 2}$ is the identity matrix, $\mathbf{1} \in \mathbb{R}^2$ is a vector of ones, and $\mathbf{m}_E \in \mathbb{R}^2$ describes the distribution of edge labels in the original graph. For the assignment of α^t, cosine schedule is utilized, where $\bar{\alpha}^t := \cos(0.5\pi(t/T + s)/(1+s))^2$ with a small s value for $\bar{\alpha}^t = \Pi_{\tau=1}^t \alpha^\tau$. Note that for categorical nodal features, the forward diffusion process follows the same procedure.

Sampling: Using the trained MPNN, synthetic graphs can be sampled iteratively. During sampling, we first sample the sensitive attributes of the nodes, $\tilde{\mathbf{S}}$, based on its original distribution in the real graph. As the next step, we need to estimate the reverse diffusion iterations $p_\theta\left(\mathcal{G}^{t-1} = (\mathbf{A}^{t-1}, \mathbf{X}^{t-1}) \mid \mathcal{G}^t = (\mathbf{A}^t, \mathbf{X}^t)\right)$, which is modeled as a product over nodes and edges (59):

$$\begin{aligned} p_\theta\left(\mathcal{G}^{t-1} \mid \mathcal{G}^t, \tilde{\mathbf{S}}, t\right) = &\prod_{i=1}^{N} \prod_{f=1}^{F} p_\theta\left(\mathbf{X}_{if}^{t-1} \mid \mathcal{G}^t, \tilde{\mathbf{S}}, t\right) \\ &\prod_{1 \leq i < j \leq N} p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathcal{G}^t, \tilde{\mathbf{S}}, t\right). \end{aligned} \tag{22}$$

To compute each independent term, we marginalize over the predictions of the MPNN:

$$\begin{aligned} p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathcal{G}^t, \tilde{\mathbf{S}}, t\right) &= \sum_{e \in \mathcal{E}^c} p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij} = e, \mathcal{G}^t\right) p_\theta\left(\mathbf{A}_{ij} = e \mid \mathcal{G}^t, \tilde{\mathbf{S}}, t\right) \\ &\approx \sum_{e \in \mathcal{E}^c} p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij} = e, \mathcal{G}^t\right) \tilde{\mathbf{A}}_{ije}, \end{aligned} \tag{23}$$

where $\mathcal{E}^c$ denotes all possible labels for edges, which are $\{0,1\}$ for an unweighted graph. Then, we can use our Markovian noise to model $p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij}=e, \mathcal{G}^t\right)$:

$$p_\theta\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij}=e, \mathcal{G}^t\right)=\begin{cases} q\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij}=e, \mathbf{A}_{ij}^t\right) & \text{if } q\left(\mathbf{A}_{ij}^t \mid \mathbf{A}_{ij}=e\right)>0 \\ 0 & \text{otherwise.}\end{cases}$$

Note that posterior $q\left(\mathbf{A}_{ij}^{t-1} \mid \mathbf{A}_{ij}=e, \mathbf{A}_{ij}^t\right)$ can be computed in closed-form using Bayes rule. Leveraging this model, G^{t-1} can be sampled, which becomes the input of the MPNN at the next time step.

E Additional Details on Datasets and Datasets Statistics

For citation networks (Cora and Citeseer), the one-hot encoding representations of the words in the article descriptions constitute the binary nodal attributes. In these networks, similar to the setups in (19; 37), the category of the articles is employed as the sensitive attribute. Furthermore, for product co-purchase networks, nodal attributes are again the one-hot encodings of the words in the product reviews and the product categories are utilized as the sensitive attributes. For the evaluation of node classification, we employ German (56) and Pokec-n (9) networks. Specifically, the German credit graph has 1,000 nodes representing the clients in a German bank, where the links are created based on the similarity of credit accounts. For this graph, the node labels classify clients into good vs. bad credit risks, where the clients' gender are employed as the sensitive attribute (28). In addition, Pokec-n is sampled from an anonymized version of the Pokec network of 2012 (a social network from Slovakia), where nodes correspond to users who live in two major regions, and the region information is utilized as the sensitive attribute (9). The working field of the users is binarized and utilized as the labels to be predicted in node classification.

Table 7: Dataset statistics.

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	F	K
Cora	2708	10556	1433	7
Citeseer	3327	9228	3703	6
Amazon Photo	7650	238163	745	8
Amazon Computer	13752	491722	767	10
Credit	1000	22242	27	2
Pokec-n	6185	21844	59	2

Statistical information for the utilized datasets are presented in Table 7, where F is the total number of nodal features and K represents the number of sensitive groups.

F Evaluation with Statistics

We evaluate the created synthetic graphs by FairWire via link prediction in Subsection 6.2. In order to provide a more traditional evaluation scheme, here we also report the 1-Wasserstein distance between the node degree distribution and clustering coefficient distribution of original graph and the synthetic ones. Table 8 presents the corresponding distance measures, where lower values for all metrics signify better performance. In Table 8, ER and SBM stand for traditional baselines Erdos–Rényi model (60) anf SBM stochastic block model (61), respectively. Furthermore, as deep learning based baselines, Feature-based MF represents feature-based matrix factorization (62), GAE and VGAE stand for graph autoencoder and variational graph autoencoder (63), respectively. Finally, GraphMaker in Table 8 corresponds to a diffusion-based graph generation baseline (47). Note that all these baselines are fairness-agnostic. Overall, results in Table 8 signify that FairWire can create synthetic graphs that follow a similar distribution to the original data while also improving the fairness metrics.

G Implementation Details

Link Prediction Model. For link prediction, we train a one-layer graph convolutional network (GCN), where the inner product between the output node representations signifies the corresponding edge probability. For training, 80% of the edges are used, where the remaining edges are split equally into two for the validation and test sets. For link prediction experiments, results are obtained for five

	Cora		Citeseer		Amazon Photo	
	Degree $\downarrow$	Cluster$\downarrow$	Degree$\downarrow$	Cluster$\downarrow$	Degree$\downarrow$	Cluster$\downarrow$
ER	1.0	2.4e^1	8.5e^{-1}	1.4e^1	1.9e^1	4.0e^1
SBM	9.6e^{-1}	2.3e^1	8.0e^{-1}	1.4e^1	1.5e^1	3.8e^1
Feature-based MF	1.3e^3	2.4e^1	1.7e^3	1.4e^1	3.8e^3	4.0e^1
GAE	1.3e^3	2.4e^1	1.7e^3	1.4e^1	3.8e^3	4.0e^1
VGAE	1.4e^3	2.4e^1	1.7e^3	1.4e^1	3.8e^3	4.0e^1
GraphMaker	2.8	2.3e^1	5.6e^{-1}	1.4e^1	1.1e^2	1.4
FairWire	1.9	2.4e^1	8.7e^{-1}	1.4e^1	8.2e^1	1.5

Table 8: Distances of statistical measures between the real graph and synthetic ones.

random data splits, and their average along with the standard deviations are reported. The weights of the GNN model for link prediction are initialized utilizing Glorot initialization (64), where it is trained for 1000 epochs by employing Adam optimizer (65). The learning rate, the dimension of hidden representations, and the dropout rate are selected via grid search for the proposed scheme and all baselines, where the value leading to the best validation set performance is selected. For learning rate the, the dimension of hidden representations, and the dropout rate, the corresponding hyperparameter spaces are $\{1e-1, 1e-2, 3e-3, 1e-3\}$, $\{32, 128, 512\}$, and $\{0.0, 0.1, 0.2\}$, respectively.

Diffusion Model. Diffusion models are trained for 10000 epochs by employing Adam optimizer (65), where the number of diffusion steps is 3. In the MPNN described in Subsection 5.2, hidden representation size for time step t is 32 for Cora and Citeseer and 16 for the Amazon Photo, German credit, and Pokec-n networks. In addition, hidden representation sizes for the nodal attributes and sensitive attributes are 512 and 64, respectively, for all datasets. The MPNN consists of two layers.

Node Classification Model. For node classification, we employ one-layer and two-layers APPNP (66) networks for German credit and Pokec-n graphs, respectively. For training, 50% of the nodes are used, where the remaining nodes are split equally into two for the validation and test sets. The weights of the GNN model for node classification are initialized utilizing Glorot initialization (64), where it is trained for 1000 epochs by employing Adam optimizer (65). The learning rate, the dimension of hidden representations, and the dropout rate are selected via grid search for the proposed scheme and all baselines, where the value leading to the best validation set performance is selected. For learning rate the, the dimension of hidden representations, and the dropout rate, the corresponding hyperparameter spaces are $\{3e-2, 1e-2, 3e-3\}$, $\{32, 128, 512\}$, and $\{0.0, 0.1\}$, respectively.

Hyperparameter Selection. For the link prediction task, we select the multiplier of $\mathcal{L}_{\text{FairWire}}$ among the values $\{0.01, 0.05, 0.1, 0.5\}$ via grid search (the multiplier of the cross-entropy loss is 1). The results for $\mathcal{L}_{\text{FairWire}}$ in Table 2 are obtained for the λ values $0.05, 0.1, 0.01/0.05, 0.1$ on Cora, Citeseer, Amazon Photo, and Amazon Computer, respectively. For adversarial regularization (9), the multiplier of the regularizer is selected via a grid search among the values $\{0.1, 1, 10\}$ (the multiplier of link prediction loss is again 1). The multiplers of the adversarial regularization for the results in Table 2 are $\{10, 1, 1, 1\}$ on Cora, Citeseer, Amazon Photo, and Amazon Computer, respectively. Furthermore, the hyperparameter δ in FairDrop algorithm is tuned among the values $\{0.16, 0.25\}$ (0.16 is the default value in their codes), where 0.16 led to the best fairness/utility trade-off on each dataset. For FairAdj (19), we use the hyperparameter values suggested by (19) directly for the citation networks.

For the generative models, we select the multiplier of $\mathcal{L}_{\text{FairWire}}$ (λ in (7)) among the values $\{0.05, 0.1, 1.0, 10.0\}$ via grid search. The results for FairWire in Table 3 are reported for the λ values $10.0, 0.1$ on Cora, and Citeseer, respectively ($\lambda = 0.05$ for Amazon photo in Table 11). For the results in Table 4, λ equals to 10 and 1 for German credit and Pokec-n, respectively. For adversarial regularization (9), the multiplier of the regularizer (again in the training loss of the MPNN) is selected via a grid search among the values $\{0.001, 0.01, 0.1\}$. The multiplers of the adversarial regularization for the results in Table 3 are $\{0.01, 0.01, 0.01\}$ on Cora, Citeseer, and Amazon Photo, respectively. Furthermore, the hyperparameter η in FairAdj (19) algorithm is tuned among the values $\{0.001, 0.005, 0.01\}$, where 0.001 led to the best fairness/utility trade-off on each dataset.

Baselines. Fairness-aware baselines in the experiments include adversarial regularization (9), FairDrop (37), FairAdj (19), and FairGen (49). Adversarial regularization refers to the bias mitigation technique where an adversary is trained to predict the sensitive attributes. In link prediction, the adversary is trained to predict the sensitive attributes of the nodes that are incident to the edges whose labels are of interest. Furthermore, FairDrop (37) is an edge dropout strategy where the dropout probabilities vary for intra- or inter-edges so as to create a more balanced graph topology. FairAdj (19) optimizes a fair adjacency with certain structural constraints for link prediction in an iterative manner considering both fairness and utility. Finally, FairGen (35) focuses on the disparities in generation quality (distances between different graph statistics) for different sensitive groups and employs fairness-aware regularizers during graph generation via a transformer-based model.

For graph generation experiments, GraphMaker (47) is utilized to create synthetic graphs in a fairness-agnostic way. While an existing work that considers fair link prediction for synthetic graphs is not available to the best of our knowledge, we employ adversarial regularization (9) as an in-processing bias mitigation strategy during the training of the MPNN described in Subsection 5.2. Furthermore, we use FairAdj (19) as a post-processing bias mitigation strategy on the synthetic graphs generated via GraphMaker, and the processed synthetic graphs are then evaluated for the link prediction and node classification tasks.

H Sensitivity Analysis

In order to examine the impact of hyperparameter selection on fairness improvements, the sensitivity analyses for the proposed tools are executed with respect to the hyperparameter λ.The results are obtained for changing λ values for both the link prediction (see (4)) and graph generation (see (7)) experiments and reported in Tables 9, 10. Overall, the results signify that both $\mathcal{L}_{\text{FairWire}}$ in the link prediction and FairWire, lead to better fairness measures compared to the natural baselines within a wide range of hyperparameter selection.

Table 9: Sensitivity Analyses for Different Tasks

	Cora			Citeseer		
Link Prediction	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
GNN	$\mathbf{94.43} \pm 0.74$	27.01 ± 1.38	9.11 ± 1.43	$\mathbf{96.16} \pm 0.28$	27.40 ± 1.24	7.37 ± 1.33
$\lambda = 0.01$	93.85 ± 1.06	16.17 ± 4.50	5.58 ± 1.18	95.55 ± 0.66	17.38 ± 7.80	6.55 ± 2.33
$\lambda = 0.05$	92.18 ± 1.03	4.76 ± 0.24	2.05 ± 0.37	$\mathbf{96.18} \pm 0.46$	12.43 ± 0.57	5.44 ± 0.86
$\lambda = 0.1$	91.98 ± 1.05	$\mathbf{4.54} \pm 0.24$	$\mathbf{1.95} \pm 0.36$	96.00 ± 0.23	$\mathbf{8.62} \pm 0.80$	$\mathbf{1.29} \pm 0.68$
	Amazon Photo			Amazon Computer		
Link Prediction	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
GNN	97.01 ± 0.26	32.65 ± 0.95	8.01 ± 0.52	$\mathbf{96.13} \pm 0.06$	23.70 ± 0.79	5.51 ± 0.79
$\lambda = 0.01$	$\mathbf{97.25} \pm 0.11$	27.75 ± 0.52	7.11 ± 0.41	$\mathbf{96.13} \pm 0.08$	23.58 ± 0.63	5.46 ± 0.56
$\lambda = 0.05$	94.85 ± 0.32	24.61 ± 0.96	6.24 ± 1.22	96.17 ± 0.13	22.74 ± 0.39	5.50 ± 0.51
$\lambda = 0.1$	88.43 ± 5.12	$\mathbf{17.78} \pm 7.27$	$\mathbf{1.48} \pm 1.00$	92.66 ± 0.21	$\mathbf{14.45} \pm 0.32$	$\mathbf{1.28} \pm 0.45$
	Citeseer			Amazon Photo		
Graph Generation	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)	AUC (%)	Δ_{SP} (%)	Δ_{EO}(%)
$\tilde{\mathcal{G}}$	$\mathbf{92.19} \pm 1.06$	37.56 ± 1.29	13.52 ± 0.92	$\mathbf{94.45} \pm 0.21$	33.49 ± 0.28	10.01 ± 0.56
$\lambda = 0.01$	92.03 ± 0.80	36.43 ± 1.44	13.11 ± 0.71	93.41 ± 0.34	27.18 ± 5.45	5.45 ± 0.80
$\lambda = 0.05$	91.89 ± 1.37	29.40 ± 4.34	9.56 ± 1.29	93.88 ± 0.40	25.27 ± 0.76	3.13 ± 0.30
$\lambda = 0.1$	91.27 ± 2.78	$\mathbf{18.35} \pm 6.91$	$\mathbf{7.80} \pm 2.66$	88.43 ± 5.12	$\mathbf{17.78} \pm 7.27$	$\mathbf{1.48} \pm 1.00$

Table 10: Sensitivity Analysis for Graph Generation on Cora

	Cora		
	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)
$\tilde{\mathcal{G}}$	87.29 ± 1.09	35.72 ± 1.74	13.27 ± 0.81
$\lambda = 1.0$	$\mathbf{88.76} \pm 1.23$	27.86 ± 2.84	10.76 ± 1.50
$\lambda = 10.0$	86.49 ± 2.79	12.91 ± 6.35	4.31 ± 3.59
$\lambda = 50.0$	82.91 ± 8.06	$\mathbf{9.86} \pm 6.54$	$\mathbf{3.47} \pm 2.58$

I Additional Graph Generation Results

Due to limited space, we present the comparative results for graph generation on Amazon Photo in Table 11 for link prediction.

Table 11: Graph generation results on Amazon-Photo

	Amazon Photo		
	AUC (%)	Δ_{SP} (%)	Δ_{EO} (%)
$\mathcal{G}$	96.91	32.58	8.24
$\tilde{\mathcal{G}}$	$\mathbf{94.45} \pm 0.21$	33.49 ± 0.28	10.01 ± 0.56
Adversarial	94.24 ± 1.20	29.17 ± 2.83	7.06 ± 2.63
FairWire	93.88 ± 0.40	$\mathbf{25.27} \pm 0.76$	$\mathbf{3.13} \pm 0.30$

Similar to the link prediction experiments (Table 2), the results of FairAdj for the Amazon Photo network could not be obtained due to computational limitations. Overall, the results reported in Table 11 conclude again that FairWire can provide the best utility/fairness trade-off compared to other fairness-aware baselines.

J Computational Resources

Experiments are carried over on 4 NVIDIA RTX A4000 GPUs.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: For c1, please see Subsection 4.1. The contribution in c2 is presented in 4.2. For c3 and c4, please check Subsections 5.1 and 5.2, respoectively. Finally, the experimental results emphasized in c5 are presented in Section 6.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see Limitations in Section 7.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Our assumptions are explicitly presented in Subsection 4.1. Furthermore, all proofs are available in Appendices B and C.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The proposed regularizer is presented with its derivation in Subsections 4.1 and 4.2. Furthermore, for reproducibility of the empirical evaluation, our codes are provided within the supplementary material of this submission. All experimental settings are also described in Subsection 6.1, and Appendix G.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Codes to reproduce all the results in Section 6 are provided in the supplementary material to this submission.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see Subsection 6.1 and Appendix G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All experimental results are obtained for multiple random seeds and their average together with standard deviations are reported.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational resources used for the experiments are clarified in Appendix J.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: To the best of our knowledge, our research is completely compliant with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This work addresses possible bias propagated by ML algorithms. Hence, it in fact helps mitigate a crucial negative impact of ML.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Citation and Amazon networks are provided by the DGL library (67), which is explained and the corresponding license is explicitly provided in the README file in our supplementary material. Furthermore, the Pokec dataset, used in this study, was created and presented in (68), which is again mentioned in the README file in our supplementary material.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.