Are Your Models Still Fair? Fairness Attacks on Graph Neural Networks via Node Injections

Zihan Luo, Hong Huang†, Yongkang Zhou, Jiping Zhang, Nuo Chen, Hai Jin
Huazhong University of Science and Technology, China
{zihanluo,honghuang,yongkangzhou_,jipingz,nuo_chen,hjin}@hust.edu.cn

Abstract

Despite the remarkable capabilities demonstrated by *Graph Neural Networks* (GNNs) in graph-related tasks, recent research has revealed the fairness vulnerabilities in GNNs when facing malicious adversarial attacks. However, all existing fairness attacks require manipulating the connectivity between existing nodes, which may be prohibited in reality. To this end, we introduce a *Node Injection-based Fairness Attack* (NIFA), exploring the vulnerabilities of GNN fairness in such a more realistic setting. In detail, NIFA first designs two insightful principles for node injection operations, namely the uncertainty-maximization principle and homophily-increase principle, and then optimizes injected nodes' feature matrix to further ensure the effectiveness of fairness attacks. Comprehensive experiments on three real-world datasets consistently demonstrate that NIFA can significantly undermine the fairness of mainstream GNNs, even including fairness-aware GNNs, by injecting merely 1% of nodes. We sincerely hope that our work can stimulate increasing attention from researchers on the vulnerability of GNN fairness, and encourage the development of corresponding defense mechanisms. Our code and data are released at: https://github.com/CGCL-codes/NIFA.

1 Introduction

Due to the strong capability in understanding graph structure, *Graph Neural Networks* (GNNs) have achieved much progress in graph-related domains such as social recommendation [30, 31] and bioinformatics [4, 27]. Nevertheless, despite the impressive capabilities demonstrated by GNNs, more and more in-depth research has revealed shortcomings in the fairness of GNN models, which greatly restricts their applications in the real world.

In fact, studies [5, 42] have found that the biases and prejudices existed in training data would be further amplified through the message propagation mechanism of GNNs, leading to model predictions being correlated with certain sensitive attributes, such as *gender* and *race*. Such correlations are usually undesired and can result in fairness issues and societal harm. For instance, in online recruitment, a recommender based on GNNs may be associated with the gender of applicants, leading to differential treatments towards different demographics and consequently giving rise to group unfairness. To address fairness issues in GNNs, researchers have proposed solutions such as adversarial learning [5, 28], data augmentation [17, 22] and others, which have achieved promising results.

However, recent research in the machine learning domain indicates that fairness is actually susceptible to adversarial attacks [3, 26, 29]. Given this, we cannot help but wonder: *"Is the fairness of GNN*

†Hong Huang is the corresponding author. Zihan Luo, Hong Huang, Yongkang Zhou, Jiping Zhang and Hai Jin are affiliated with the National Engineering Research Center for Big Data Technology and System, Service Computing Technology and Systems Laboratory, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-2286

models also highly vulnerable?" For example, in e-commerce, if attackers could exacerbate performance disparities between male and female user groups by attacking GNN-based recommendation models, they could ultimately cause the e-commerce platform to provoke dissatisfaction from specific user demographics and gradually lose its appeal among these users. Several studies [11, 13, 41] have explored the vulnerability of GNN fairness and proposed effective attack strategies. Unlike conventional attacks, these fairness attacks aim to undermine GNN fairness without excessively compromising its utility. However, all these works require altering connectivity between existing nodes, whose authority is typically limited in the real world, such as modifying the relationship between real users. In contrast, injecting fake nodes into the original graph is a more practical way to launch an attack without manipulating the existing graph [12, 32, 34], which is still under-explored in the field of GNN fairness attack. To address this gap, we aim to be the first to launch an attack on GNN fairness via node injection, examining their vulnerabilities under such a more realistic setting.

Specifically, launching a node injection-based fairness attack on GNNs is non-trivial, whose challenges can be summarized as follows: **RQ1:** *How to determine the node injection strategy?* The node injection can be decomposed into two steps, including selecting appropriate target nodes and connecting the injected nodes with them, both will impact the effectiveness of the attack. **RQ2:** *How to determine the features of the injected nodes after node injection?* Like the real nodes, injected nodes will also participate in the message propagation process of GNNs, thereby affecting their neighbors and even the whole graph. Given the key role of massage propagation in GNN fairness [40, 45], proposing suitable strategies to determine the features of injected nodes is also important.

To address these challenges, we propose a gray-box poisoning attack method namely *Node Injection-based Fairness Attack* (NIFA) during the GNN training phase. In detail, for the two steps in the first challenge, NIFA innovatively designs two corresponding principles. The first is the *uncertainty-maximization principle*, which asks to select real nodes with the highest model uncertainty as target nodes for injection. The idea is that nodes with higher uncertainty are typically more susceptible to attacks, thereby ensuring the attack's effectiveness. After selecting target nodes, NIFA follows the second principle, the *homophily-increase principle*, to connect target nodes with injected nodes. This principle aims to deteriorate GNN fairness by enhancing message propagation within sensitive groups [22, 40]. For the second challenge, multiple novel objective functions are proposed after node injection to guide the optimization of the injected nodes' features, which could further impact the victim GNN's fairness from a feature perspective. In summary, our contributions are as follows:

- To the best of our knowledge, we are the first to conduct fairness attacks on GNNs via node injections, and our work successfully highlights the vulnerability of GNN fairness. We also summarize several key insights for the future defense of GNNs' fairness attacks from the success of NIFA.
- We propose a node injection-based gray-box attack named NIFA. To be concrete, NIFA first designs two novel principles to guide the node injection operations from a structure perspective, and then proposes multiple objective functions for the injected nodes' feature optimization.
- We conduct extensive experiments on three real-world datasets, which consistently show that NIFA can effectively attack existing GNN models with only a 1% perturbation rate and an unnoticeable utility compromise, even including fairness-aware GNN models. Comparisons with other state-of-the-art baselines also verify the superiority of NIFA in achieving fairness attacks.

2 Related work

Fairness on GNNs. Researchers have discovered various fairness issues of GNNs, which often lead to societal harms [22, 37] and performance deterioration [21, 33] in practical applications. Algorithmic fairness on GNNs can be categorized into two main types based on the definition: individual fairness [1, 6] and group fairness [5, 37, 46]. Individual fairness requires that similar individuals should receive similar treatment, while group fairness aims to protect specific disadvantaged groups [16]. In detail, many researchers have delved into studies focusing on fairness grounded in sensitive attributes. For instance, Dai et al. [5] reduce the identifiability of sensitive attributes in node embeddings through adversarial training to enhance fairness. FairVGNN [38] goes a step further by introducing a feature masking strategy to address the problem of sensitive information leakage during the feature propagation process in GNNs. Graphair [17] achieves fairness through an automated data augmentation method and FairSIN [40] designs a novel sensitive information neutralization method for fairness. Beyond fairness related to sensitive attributes, some researchers also direct

attention to fairness related to graph structures, like DegFairGNN [21] and Ada-GNN [23]. In this work, we mainly focus on attacks on the group fairness of GNNs based on sensitive attributes.

Attacks on GNNs. Finding out potential vulnerabilities thus improving the security of GNNs remains a pivotal concern in the field of trustworthy GNNs [42]. From the perspective of attackers, they aim to compromise the GNNs' performance on graph data via manipulating graph structures [24, 32, 47], node attributes [48], or node labels [19]. Among these methods, node injection attacks, given the attackers' limited authority to manipulate the connectivity between existing nodes, emerge as one of the most prevalent methods [32, 34, 43]. However, existing attacks, including node injection attacks, mainly focus on undermining GNN's utility, with little attention to the vulnerability of GNN fairness. Different from attacks on GNN utility, fairness-targeted attacks aim to deteriorate the fairness without significantly compromising the accuracy. FA-GNN [11], FATE [13], and G-FairAttack [41] stand out as the few ones that we are aware of to explore attacks on GNN fairness. FA-GNN's empirical findings suggest that adding edges with certain strategies can significantly compromise GNN fairness without affecting its performance [11]. FATE [13] formulates the fairness attacks as a bi-level optimization problem and proposes a meta-learning-based framework. G-FairAttack [41] designs a novel surrogate loss with utility constraints to launch the attacks in a non-gradient manner. Nevertheless, all these works require modifying the link structure between existing nodes, which may be prohibited in reality due to the lack of authority.

3 Preliminary

Here we will introduce some basic notations and concepts, and then give our problem definition.

3.1 Notations

A graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$, where $\mathcal{V}$ is the node set, and $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ represents the adjacency matrix. $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times D}$ denotes the feature matrix, in which D is the feature dimension. Under the settings of node classification, each node $v \in \mathcal{V}$ will be assigned with a label $y_v \in \mathcal{Y}$, and a GNN-based mapping function $f_\theta : \{\mathcal{V}, \mathcal{G}\} \to \{1, 2, ..., |\mathcal{Y}|\}^{|\mathcal{V}|}$ with parameters θ is learned to leverage the graph signals for label prediction, where $\mathcal{Y}$ represents the true label set.

3.2 Fairness-related concepts

In alignment with prior works [5, 7, 17], we mainly focus on group fairness where each node will be assigned with a binary sensitive attribute $s \in \{0, 1\}$, although our attack could also be generalized to the settings of multi-sensitive groups and we leave this as our future work. Based on the sensitive attributes, the nodes can be divided into two non-overlapped groups $\mathcal{V} = \{\mathcal{V}_0, \mathcal{V}_1\}$, and we employ the following two kinds of fairness related definitions:

Definition 1. ***Statistical Parity (SP).*** *The Statistical Parity requires the prediction probability distribution to be independent of sensitive attributes, i.e. for any class $y \in \mathcal{Y}$ and any node $v \in \mathcal{V}$:*

$$P(\hat{y}_v = y|s = 0) = P(\hat{y}_v = y|s = 1), \tag{1}$$

where $\hat{y}_v$ denotes the predicted label of node v.

Definition 2. ***Equal Opportunity (EO).*** *The Equal Opportunity requires that the probability of predicting correctly is independent of sensitive attributes, i.e. for any class $y \in \mathcal{Y}$ and any node $v \in \mathcal{V}$, we can have:*

$$P(\hat{y}_v = y|y_v = y, s = 0) = P(\hat{y}_v = y|y_v = y, s = 1). \tag{2}$$

Based on the above definitions, we can define two kinds of metrics Δ_{SP} and Δ_{EO} to quantitatively measure fairness. For both metrics, smaller values indicate better fairness:

$$\Delta_{SP} = \mathbb{E}|P(\hat{y} = y|s = 0) - P(\hat{y} = y|s = 1)|, \tag{3}$$

$$\Delta_{EO} = \mathbb{E}|P(\hat{y} = y|y = y, s = 0) - P(\hat{y} = y|y = y, s = 1)|. \tag{4}$$

 https://doi.org/10.52202/079017-2286

3.3 Problem definition

In this paper, our goal is to launch fairness-targeted attacks on GNN models through the application of node injection during the training phase, i.e. poisoning attack. Following the line of previous attacks on GNNs [32, 41], our attack is under the prevalent gray-box setting, where the attackers can obtain the graph $\mathcal{G}$ with node labels $\mathcal{Y}$, and the sensitive information s, but can not access the model architecture and parameters θ. Detailed introduction to our attack settings is provided in Appendix B. Specifically, through injecting malicious node set $\mathcal{V}_I$ into the graph, the original graph $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$ is poisoned as $\mathcal{G}' = (\mathcal{V}', \mathbf{A}', \mathbf{X}')$, where

$$\mathcal{V}' = \mathcal{V} \cup \mathcal{V}_I, \;\; \mathbf{X}' = \begin{bmatrix} \mathbf{X} \\ \mathbf{X}_I \end{bmatrix}, \mathbf{X}_I \in \mathbb{R}^{|\mathcal{V}_I| \times D}, \tag{5}$$

$$\mathbf{A}' = \begin{bmatrix} \mathbf{A} & \mathbf{V}_I \\ \mathbf{V}_I^T & \mathbf{A}_I \end{bmatrix}, \mathbf{V}_I \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}_I|}, \mathbf{A}_I \in \mathbb{R}^{|\mathcal{V}_I| \times |\mathcal{V}_I|}. \tag{6}$$

Both $\mathbf{V}_I$ and $\mathbf{A}_I$ are matrices for illustrating the connectivity related to injected nodes, and $\mathbf{X}_I$ is the feature matrix of injected nodes $\mathcal{V}_I$. The true label set $\mathcal{Y}$ will not be poisoned by injected nodes in our settings, as such information is typically hard to modify in reality. For conciseness, we denote $\mathcal{F}(\cdot)$ and $\mathcal{M}(\cdot)$ as the evaluation functions on fairness and utility for the learned mapping function f_θ, respectively. Then our goal as an injection-based attack on fairness could be formulated as:

$$\begin{aligned} &\max_{\mathcal{G}'} |\mathcal{F}(f_{\theta^*}(\mathcal{V}, \mathcal{G}'))| \\ &\text{s.t. } \arg\max_{\theta^*} \mathcal{M}(f_{\theta^*}(\mathcal{V}, \mathcal{G}')), \;\; \mathcal{G}' = (\mathcal{V}', \mathbf{A}', \mathbf{X}'), \;\; |\mathcal{V}_I| \leq b, \;\; deg(v)_{v \in \mathcal{V}_I} \leq d. \end{aligned} \tag{7}$$

As a poisoning attack, the first constraint in Eq. (7) requires to train the victim model f_{θ^*} with parameters θ^* on the poisoned graph $\mathcal{G}'$, so that the predictions of f_{θ^*} are as correct as possible before evaluating the attack performance. The following constraints in Eq. (7) make sure that the proposed attack is unnoticeable and deceptive to the defenders, i.e. the number of injected nodes is below a predefined budget b[1] and the degrees of injected nodes are constrained by a budget d. Our goal is to find a poisoned graph $\mathcal{G}'$ to deteriorate the fairness of victim models f_{θ^*} as severely as possible, i.e. maximize the fairness metrics Δ_{SP} and Δ_{EO} introduced previously.

4 Methodology

In this section, we first give an overview of our attack method NIFA. Then we will elaborate on the details of each module and summarize the implementation algorithms at last.

4.1 Framework overview

The overall framework of NIFA is illustrated in Figure 1. As mentioned before, NIFA first employs two principles to guide the node injection operations. For the first uncertainty-maximization principle, NIFA utilizes the Bayesian GNN for model uncertainty estimation of each node, and then selects target nodes with the highest uncertainty (a). As for the second homophily-increase principle, NIFA requires each injected node can only establish connections to target nodes from one single sensitive group (b), thus increasing the homophily-ratio and enhancing information propagation within sensitive groups. After node injection, multiple objective functions are designed to guide the optimization of injected nodes' feature matrix, where we introduce an iterative optimization strategy for avoiding over-fitting issues (c). The details of each part will be introduced later.

4.2 Node injection with principles

The first step of NIFA is conducting node injections, which aims to ensure the effectiveness of NIFA from a structure perspective. In detail, we propose two novel principles to guide the node injection operations, namely *Uncertainty-maximization principle* and *Homophily-increase principle*.

[1]Same as the prior work [32], we define the perturbation rate as the ratio of injected nodes to the labeled nodes in the original graph, i.e. $|\mathcal{V}_I|/|\mathcal{V}_L|$, where $\mathcal{V}_L$ denotes the labeled node set.

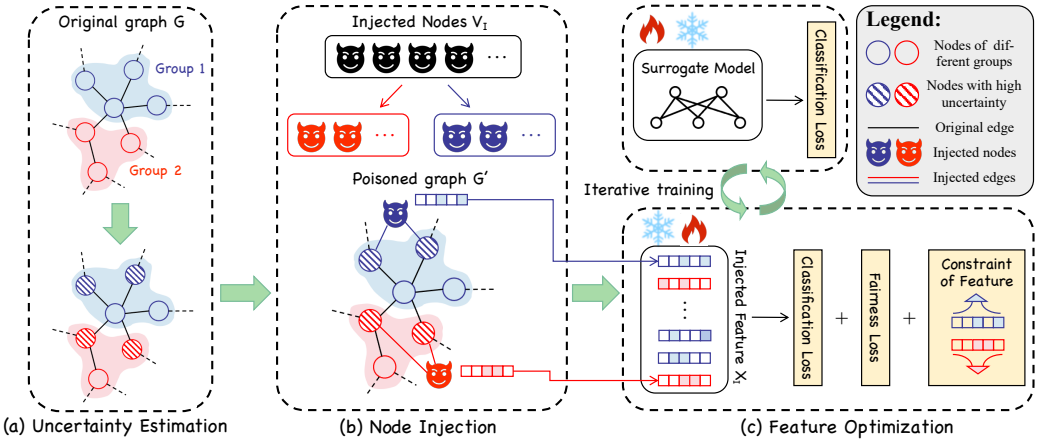


Figure 1: The overall framework of NIFA: (a) Utilizing uncertainty estimation, nodes exhibiting high uncertainty (depicted as shaded nodes) are designated as targeted nodes. (b) Injected nodes are equally assigned to each sensitive group, and only connect targeted nodes with the same sensitive attribute. (c) After node injection, the injected feature matrix and surrogate model are optimized iteratively by diverse objective functions.

Uncertainty-maximization principle. Intuitively, nodes with higher model uncertainties are positioned closer to the decision boundary, which means their predicted labels are more vulnerable and easier to flip when facing adversarial attacks. We acknowledge that the model uncertainty may not be the only method to measure the vulnerability of nodes, and we will discuss potential alternative approaches in Appendix H.2. Inspired by [20], we utilize a Bayesian GNN to estimate the model uncertainty of each node, where we employ the Monte Carlo dropout approach [9] to approximate the distributions of the sampled model parameters. Given a GNN with parameters $\theta_{\mathcal{B}}$, we obtain different model parameters through T times independent Bernoulli dropout sampling processes, i.e.:

$$\begin{aligned} P(M_i) &\sim \text{Bernoulli}(p) \\ \theta_{\mathcal{B}_i} &= M_i \odot \theta_{\mathcal{B}} \end{aligned}, i \in \{1, 2, \ldots, T\}, \tag{8}$$

where M_i is the ith sampled binary mask following the Bernoulli distribution with parameter p, and $\odot$ denotes the dot production operation. Here we take a two-layer GCN [10] with parameters $\theta_{\mathcal{B}}$ as the Bayesian GNN for estimating the uncertainty of each node, and $\theta_{\mathcal{B}}$ is optimized by minimizing the following objective function, which consists of a cross-entropy loss plus a regularization term:

$$L(\theta_{\mathcal{B}}) = -\frac{1}{T}\sum_{i=1}^{T} \mathcal{Y} \log(f_{\theta_{\mathcal{B}_i}}(\mathcal{V}, \mathcal{G})) + \frac{1-p}{2T}\|\theta_{\mathcal{B}}\|_2^2, \tag{9}$$

where $\mathcal{Y}$ denotes the true labels, $\|\cdot\|_2^2$ denotes the L2 regularization, T is the number of sampling processes and $f_{\theta_{\mathcal{B}_i}}(\cdot)$ is the mapping function with the ith sampled parameters $\theta_{\mathcal{B}_i}$. After the training process, model uncertainty can be estimated by calculating the variance of T times predictions with the sampled parameters $\{\theta_{\mathcal{B}_i}\}_{i=1}^{T}$. Intuitively, nodes with lower variance are more confident in their predictions and vice versa. Thus, the model uncertainty scores $U \in \mathbb{R}^{|\mathcal{V}|}$ are positively correlated with the model prediction variance, and we simply estimate U with the following formulation:

$$U = \underset{i=1}{\overset{T}{Var}}(f_{\theta_{\mathcal{B}_i}}(\mathcal{V}, \mathcal{G})). \tag{10}$$

Under the guidance of the uncertainty-maximization principle, we will select nodes with the top $k\%$ model uncertainty U in each sensitive group as the target nodes, where k is a hyper-parameter.

Homophily-increase principle. After selecting target nodes, the next step is to connect the injected nodes with them. In particular, we first present our strategy in this step, with more rationales provided later: the injected nodes $\mathcal{V}_I$ are first equally assigned to each sensitive group in the graph, then each injected node will exclusively connect to d random target nodes with the same sensitive attribute, as illustrated in Figure 1(b), where d is a hyper-parameter. At this stage, the node injection operations are completed, with the structure of the original graph $\mathcal{G}$ manipulated.

Intuitively, compared with random node injection, our strategy prevents information propagation between nodes of different sensitive groups through the injected nodes, making it easier to accentuate

https://doi.org/10.52202/079017-2286

differences in embeddings between groups and thereby exacerbate unfairness issues [37]. We also provide a brief theoretical analysis to show that such a strategy could lead to the increase of node-level homophily ratio. Similar to [8], we define the node-level homophily-ratio $\mathcal{H}_u$ as the ratio of neighbors of node u that have the same sensitive attribute as node u, i.e. $\mathcal{H}_u = \frac{\sum_{v \in \mathcal{N}_u} \mathbb{1}(s_u = s_v)}{|\mathcal{N}_u|}$, where $\mathcal{N}_u$ denotes the neighbors of node u and $\mathbb{1}(\cdot)$ is an indicator function. Then we can have:

Lemma 1. *For target node u that will connect with injected nodes, our proposed node injection strategy will lead to the increase of node-level homophily-ratio $\mathcal{H}_u$.*

Due to space limitation, the proof for Lemma 1 is provided in Appendix C. It is worth noting that $\mathcal{H}_u$ is also equivalent to the probability of choosing neighbors with the same sensitive attribute for node u. From the perspective of message propagation, higher node-level homophily-ratio indicates that more sensitive-related information will be aggregated to the target node, thus leading to more severe unfairness issues on sensitive attributes[2]. We believe that such characteristics could empower our node injection strategy with stronger capability on fairness attacks.

4.3 Feature optimization

In this part, we will introduce the details of optimizing injected nodes' features $\mathbf{X}_I$, which helps further advance the effectiveness of NIFA. Generally, under a gray-box attack setting, there is no visible information about the victim models for the attackers, thus requiring attackers to propose a surrogate GNN model $\mathcal{S}$ at first for assessing their attacks. To be specific, similar to the training process of victim models as described in Eq. (7), the surrogate model $\mathcal{S}$ will be trained on the poisoned graph $\mathcal{G}'$ and optimize its parameter $\theta_{\mathcal{S}}$ to maximize the utility. Conversely, $\mathbf{X}_I$ is designed to mislead $\mathcal{S}$, ensuring that even a well-trained surrogate model will still maintain high unfairness under attacks. Instead of employing a pre-trained frozen surrogate model $\mathcal{S}$, NIFA asks two components, i.e. $\mathcal{S}$ and $\mathbf{X}_I$ to be trained iteratively with different objective functions, which avoids the attack being over-fitting to specific model parameters. In detail, the surrogate model $\mathcal{S}$ follows the common training procedure of a GNN classifier with cross-entropy loss, while for the injected nodes' feature optimization, we devise multiple effective objective functions as follows:

Classification loss. Although our primary goal is to maximize the unfairness of a GNN model, it is crucial to ensure that the utility of the victim model will not experience a significant decrease after training on a poisoned graph [11, 13, 41], thus being unnoticeable for utility-based attack detection. To this end, we set cross-entropy loss as our first objective function, i.e.:

$$L_{CE} = -\frac{1}{|\mathcal{V}^{tr}|} \sum_{u \in \mathcal{V}^{tr}} y_u \log h_u, \tag{11}$$

where $\mathcal{V}^{tr}$ denotes the original training node set, and h_u denotes the output logits of node u.

Fairness loss. Aiming at enlarging the unfairness on GNNs, we then design two kinds of fairness loss based on the definitions of Δ_{SP} and Δ_{EO}, which are formulated as:

$$L_{SP} = -\| \frac{1}{|\mathcal{V}_0^{tr}|} \sum_{u \in \mathcal{V}_0^{tr}} h_u - \frac{1}{|\mathcal{V}_1^{tr}|} \sum_{u \in \mathcal{V}_1^{tr}} h_u \|_2^2, \tag{12}$$

$$L_{EO} = -\| \sum_{y \in \mathcal{Y}} (\frac{1}{|\mathcal{V}_{0,y}^{tr}|} \sum_{u \in \mathcal{V}_{0,y}^{tr}} h_{u,y} - \frac{1}{|\mathcal{V}_{1,y}^{tr}|} \sum_{u \in \mathcal{V}_{1,y}^{tr}} h_{u,y}) \|_2^2, \tag{13}$$

where $h_u \in \mathbb{R}^{|\mathcal{Y}|}$ denotes the raw output of node u, and $h_{u,y} \in \mathbb{R}$ denotes the raw output of node u on class y. $\mathcal{V}_{i,y}^{tr}$ denotes the training nodes with sensitive attribute i and label y. By minimizing L_{SP} and L_{EO}, the gap in output between different groups increases, leading to high unfairness.

Constraint of feature. To further accentuate the differences between different sensitive groups, it is important to ensure that the information introduced by injected nodes for different sensitive groups

[2]Similar conclusions have been concluded from multiple prior works [22, 26, 40]. For better a understanding of the relationship between the homophily-ratio and unfairness, one can also refer to [40], which provides the corresponding theoretical analysis from a massage propagation perspective.

is distinct during the message propagation process. To this end, we devise the following constraint function on the injected node feature matrix $\mathbf{X}_I$:

$$L_{CF} = -\|\frac{1}{|\mathcal{V}_{I,0}|}\sum_{u\in\mathcal{V}_{I,0}}\mathbf{X}_{I,u} - \frac{1}{|\mathcal{V}_{I,1}|}\sum_{u\in\mathcal{V}_{I,1}}\mathbf{X}_{I,u}\|_2^2, \tag{14}$$

where $\mathcal{V}_{I,i}$ is the injected node set linking to the ith sensitive group during the node injection.

Overall loss. By combining the aforementioned objective terms, the overall loss L for injected nodes' features optimization can be formulated as:

$$L = L_{CE} + \alpha \cdot L_{CF} + \beta \cdot (L_{SP} + L_{EO}), \tag{15}$$

where α and β are two hyper-parameters to control the weights of different objective functions.

4.4 Implementation algorithm

Training process. Due to space limitation, we summarize the pseudo-code of NIFA in Algorithm 1 in Appendix D. Initially, we perform node injection operations based on two proposed principles (lines 2-4). Subsequently, an iterative training strategy is utilized to optimize the surrogate model and injected nodes' feature (lines 5-15). Specifically, after each inner loop for $\mathbf{X}_I$ training, it is clamped to fit the range of the original feature $\mathbf{X}$ (line 14) so that the defenders cannot filter out the injected nodes easily through abnormal feature detection. For datasets with discrete features, $\mathbf{X}_I$ is rounded to the nearest integer at the end of the training process (lines 16-18).

Inference process. As a poisoning attack, the original clean graph $\mathcal{G}$ is poisoned as $\mathcal{G}'$ after malicious node injection and feature optimization. The victim models will re-train on the poisoned graph $\mathcal{G}'$ normally, and we take the predictions from the poisoned victim model for final evaluation.

5 Experiments

5.1 Experimental settings

Datasets. Experiments are conducted on three real-world datasets namely Pokec-z, Pokec-n, and DBLP. Both **Pokec-z** and **Pokec-n** are subgraphs sampled from Pokec, one of the largest online social networks in Slovakia, according to the provinces of users [5]. Each node in these graphs represents a user, while each edge represents an unidirectional following relationship. The datasets provide node attributes including age, gender, and hobbies, and the classification task is to predict the working fields of users. **DBLP** is a coauthor network dataset [11], where each node represents an author and two authors will be connected if they publish at least one paper together. The node features are constructed based on the words selected from the corresponding author's published papers. The final classification task is to predict the research area of the authors. The detailed dataset statistics are summarized in Table 1.

Table 1: Dataset statistics

Dataset	Pokec-z	Pokec-n	DBLP
# of nodes	67,796	66,569	20,111
# of edges	617,958	517,047	57,508
feature dim.	276	265	2,530
# of labeled nodes	10,262	8,797	3,196

Victim models. As a gray-box attack method, we target multiple classical GNNs as victim models, including GCN [14], GraphSAGE [10], APPNP [15], and SGC [39]. We also include three well-established fairness-aware GNNs — FairGNN [5], FairVGNN [38], and FairSIN [40] as our selected victim models. The details of these victim models will be elaborated in Appendix E.

Baselines. Depending on the attack goals, we mainly consider the following two kinds of graph attack methods as our baselines, including *1) Utility attack*: AFGSM [36], TDGIA [47] , and $\mathrm{G}^2\mathrm{A2C}$ [12], and *2) Fairness attack*: FA-GNN [11], FATE [13], and G-FairAttack [41]. The details of these baselines will be further introduced in Appendix F.

Implementation details. As shown in Table 1, only a part of the nodes have the label information, and we randomly select 50%, 25%, and 25% labeled nodes as the training set, validation set, and test set, respectively. In line with the prior work, [11], we choose *region* as the sensitive attribute for Pokec-z and Pokec-n, and *gender* for DBLP. For all victim models, we employ a two-layer GCN model as the surrogate model. Due to space limitations, please refer to Appendix G for more reproducibility details.

https://doi.org/10.52202/079017-2286

Table 2: Attack performance of NIFA on different victim GNN models. The results are reported in percentage (%). We **bold** the results when NIFA successfully deteriorates the fairness of victim GNN models (smaller Δ_{SP} and Δ_{EO} indicate better fairness, and we aim to maximize the fairness metrics for a fairness attack).

		Pokec-z			Pokec-n			DBLP		
		Accuracy	Δ_{SP}	Δ_{EO}	Accuracy	Δ_{SP}	Δ_{EO}	Accuracy	Δ_{SP}	Δ_{EO}
GCN	before	71.22±0.28	7.13±1.21	5.10±1.28	70.92±0.66	0.88±0.62	2.44±1.37	95.88±1.61	3.84±0.34	1.91±0.75
	after	70.50±0.30	**17.36±1.16**	**15.59±1.08**	70.12±0.37	**10.10±2.10**	**9.85±1.97**	93.37±1.48	**13.49±2.83**	**20.33±3.82**
GraphSAGE	before	70.79±0.62	4.29±0.84	3.46±1.12	68.77±0.34	1.65±1.31	2.34±1.04	96.58±0.29	4.27±1.09	2.78±0.91
	after	70.05±1.25	**6.20±1.63**	**4.20±1.77**	68.93±1.19	**3.32±1.88**	**3.56±1.91**	93.92±0.74	**10.16±2.24**	**16.65±3.30**
APPNP	before	69.79±0.42	6.83±1.25	5.07±1.26	68.73±0.64	3.39±0.28	3.71±0.28	96.58±0.38	3.98±1.18	2.20±1.08
	after	69.12±0.70	**18.44±1.41**	**16.85±1.50**	67.90±0.76	**13.47±3.22**	**13.52±3.56**	92.46±0.94	**13.88±3.20**	**20.20±4.25**
SGC	before	69.09±0.99	7.28±1.50	5.45±1.42	66.95±1.69	2.74±0.85	3.21±0.78	96.53±0.48	4.70±1.26	3.11±1.24
	after	67.83±0.70	**17.65±1.01**	**16.09±1.06**	66.72±1.21	**10.59±2.40**	**10.67±2.61**	92.56±1.09	**13.88±3.37**	**20.25±4.44**
FairGNN	before	68.75±1.12	1.89±0.63	1.51±0.47	69.41±0.66	1.42±0.35	2.32±0.57	93.12±1.23	1.95±0.99	3.09±1.81
	after	69.38±2.07	**5.71±2.52**	**4.22±1.89**	69.97±0.42	**6.13±5.81**	**6.33±5.77**	92.56±1.49	**5.89±2.52**	**10.48±3.82**
FairVGNN	before	68.57±0.45	3.79±0.51	2.59±0.59	67.77±1.00	1.90±1.23	3.10±1.20	95.18±0.54	1.90±0.52	2.91±1.05
	after	67.65±0.38	**11.01±2.79**	**9.28±2.87**	65.74±1.42	**3.51±1.51**	**3.65±1.56**	91.56±1.13	**7.96±1.49**	**13.57±2.57**
FairSIN	before	67.33±0.22	1.73±1.49	2.61±1.44	67.18±0.30	0.39±0.89	2.40±1.02	94.72±0.62	0.23±0.15	0.45±0.16
	after	66.55±0.44	**9.48±2.62**	**10.39±1.06**	66.20±0.12	**11.82±0.75**	**14.58±0.22**	92.46±0.32	**10.90±2.12**	**23.65±7.77**

Table 3: Accuracy and Fairness performance of attack launched by the different attackers. The results are reported in percentage (%). The best attack performance on fairness is **bolded**.

	Pokec-z			Pokec-n			DBLP		
	Accuracy	Δ_{SP}	Δ_{EO}	Accuracy	Δ_{SP}	Δ_{EO}	Accuracy	Δ_{SP}	Δ_{EO}
Clean	71.22±0.28	7.13±1.21	5.10±1.28	70.92±0.66	0.88±0.62	2.44±1.37	95.88±1.61	3.84±0.34	1.91±0.75
Utility attacks on GNNs									
AFGSM	67.01±0.24	3.07±1.67	3.45±0.22	68.21±0.23	5.35±0.15	5.68±0.14	95.38±0.30	5.44±3.48	2.78±0.41
TDGIA	62.20±0.04	1.66±0.16	0.77±0.10	63.57±0.08	7.28±0.35	6.95±0.34	93.42±0.29	0.93±0.70	1.82±0.87
G²A2C	39.41±0.94	6.89±0.91	6.11±0.48	34.30±1.71	2.23±1.40	3.76±1.04	86.28±0.25	4.21±0.66	3.80±0.42
Fairness attacks on GNNs									
FA-GNN	69.80±0.48	6.62±1.21	8.67±1.28	70.80±0.97	2.64±0.76	3.45±0.54	95.48±0.48	3.32±1.65	8.74±1.23
FATE	-	-	-	-	-	-	94.87±0.41	3.62±1.49	2.12±1.01
G-FairAttack	-	-	-	-	-	-	95.12±0.38	6.80±0.59	2.94±1.10
NIFA	70.50±0.30	**17.36±1.16**	**15.59±1.08**	70.12±0.37	**10.10±2.10**	**9.85±1.97**	93.37±1.48	**13.49±2.83**	**20.33±3.82**

5.2 Main attack performance

To comprehensively evaluate NIFA's effectiveness, we employ multiple mainstream GNNs including GCN, GraphSAGE, APPNP, and SGC, besides three classical fairness-aware GNNs namely FairGNN, FairVGNN, and FairSIN as our victim models. We record the average accuracy, Δ_{SP} and Δ_{EO} before and after conducting our poisoning attack on the victim models five times. The experimental results are reported in Table 2, and we can have the following observations:

- The proposed attack demonstrates consistent effectiveness on all datasets with different mainstream GNNs as victim models. For instance, the Δ_{SP} and Δ_{EO} of GCN on Pokec-z increase significantly from 7.13%, 5.10% to 17.36% and 15.59%, respectively. Such observation successfully reveals the vulnerability of GNN fairness under our node injection-based attacks.
- On three fairness-aware models, FairGNN, FairVGNN, and FairSIN, NIFA still causes noticeable fairness impacts. For example, the Δ_{EO} of FairVGNN on Pokec-z increases from 2.59% to 9.28%, a nearly fourfold increase. It indicates that even fairness-aware GNN models are also vulnerable to our attack, highlighting the urgency of proposing more robust fairness mechanisms.
- Instead of sacrificing the utility of victim GNNs for better fairness attack results, all victim models' accuracy is only slightly impacted on all datasets, which illustrates the distinction between fairness attacks and utility attacks, and underscores NIFA's deceptive nature for administrators.

5.3 Comparison with other attack Models

In this section, we aim to compare NIFA with several competitors on graph attacks. Specifically, we choose six well-established attackers on either utility or fairness as our baselines, including AFGSM [36], TDGIA [47], G^2A2C [12], FA-GNN [11], FATE [13], and G-FairAttack [41]. For all baselines, the victim model is set as GCN, and the numbers of injected nodes or modified edges are set to be the same as ours for a fair comparison. Note that, both FATE and G-FairAttack fail

to deploy on Pokec-z and Pokec-n due to scalability issues[3]. Results after repeating five times are shown in Table 3.

It can be seen that NIFA consistently achieves the state-of-the-art fairness attack performance on three datasets. The reasons might be two-fold: 1) the utility attack methods are mainly designed to impact the accuracy of victim models, while overlooking the fairness objectives. 2) As pioneering works in fairness attacks on GNNs, all baselines on fairness attacks need to modify the original graph, such as adding or removing some edges or modifying features of real nodes. For deceptiveness consideration, their modifications are usually constrained by a small budget. However, NIFA introduces new nodes into the original graph through node injection and can optimize the injected nodes in a relatively larger feature space. Such superiority of node injection attack helps NIFA have a greater impact on the original graph from the feature perspectives.

5.4 Ablation study

In this part, we conduct ablation experiments to prove the effectiveness of the uncertainty-maximization principle, homophily-increase principle, and iterative training strategy, respectively. In detail, we consider the following three variants of NIFA. 1) *NIFA-U*: the uncertainty-maximization principle is removed, and we randomly choose targeted nodes from the labeled nodes. 2) *NIFA-H*: we still choose real nodes with the top k% model uncertainty as the targeted nodes, but the homophily-increase principle is removed, i.e. each injected node may connect with targeted nodes from different sensitive groups simultaneously. 3) *NIFA-I*: we remove the iterative training strategy here, which means that the surrogate model is trained on the clean graph in advance, and the feature optimization process will only involve the training process of injected feature matrix. For all variants, we set GCN as the victim model.

The results are reported in Figure 2, where we can have the following observations. Firstly, after removing the uncertainty-maximization principle (NIFA-U), the fairness attack performance consistently decreases on three datasets. This is expected since the concept of uncertainty helps NIFA find more vulnerable nodes, thus improving the attack effectiveness. Secondly, after removing the homophily-increase principle (NIFA-H), the attack performance drops obviously, which verifies the homophily-ratio is crucial in GNN fairness. Finally, without iterative training during feature optimization (NIFA-I), the attack performance decreases slightly on all datasets. The main reason is that the iterative training strategy could help NIFA to have better robustness to dynamic victim models.

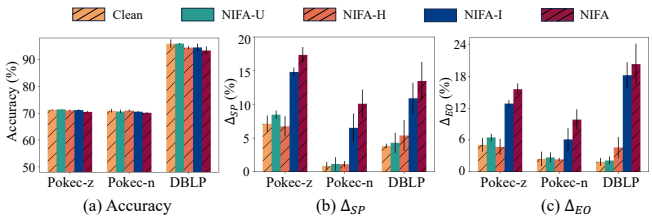


Figure 2: Ablation study of each module in NIFA

6 Defense discussion to fairness attacks on GNNs

As previously emphasized, our intrinsic aim is to unveil the vulnerabilities of existing GNN models in terms of fairness, thereby inspiring related defense research. In fact, as an emerging field that is just beginning to be explored, defense strategies against GNN fairness attacks are relatively scarce. However, we still can summarize several key insights from NIFA for further careful study:

Reliable training nodes. One key assumption in NIFA is that the nodes with high model uncertainty will be much easier to be attacked, which can also be supported by the ablation study in Section 5.4. In this way, administrators can pay more attention to these nodes and their abnormal neighbors for defense

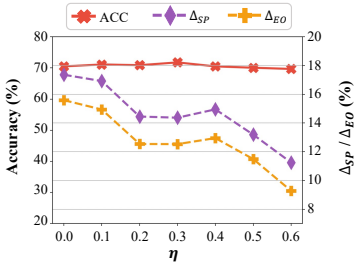


Figure 3: Defense performance on Pokec-z with masking η training nodes with the highest uncertainty

[3]On Pokec-z and Pokec-n datasets, FATE reports OOM errors and G-FairAttack fails to complete the attack within three days. More scalability analysis will be given in Appendix H.4.

https://doi.org/10.52202/079017-2286

purposes. For example, engineers can pre-train a model to detect the abnormal nodes or edges in advance, especially those that emerged recently in the training data, and weaken their impacts on the model by randomly masking these nodes or edges in the input graph or decreasing their weights in the message propagation during the training of GNNs.

To verify our assumption, we conduct a simple experiment by removing a proportion of nodes (η) with the highest uncertainty U from supervision signals after the attack. Similarly, GCN [10] is employed as the victim model and we gradually tune η from 0 to 0.6 with step 0.1, where $\eta = 0$ means no defense is involved. The performance of NIFA on the Pokec-z dataset with different η is illustrated in Figure 3. It can be seen that, since NIFA mainly focuses on attacking nodes with high uncertainty, after masking a part of these nodes during the training stage, the fairness attack performance of NIFA gradually decreases with a small fluctuation in accuracy. However, it is worth noting that although such an intuitive strategy can defend the attack from NIFA to some extent, there is still obvious fairness deterioration compared with the performance of clean GCN in Table 2 (Δ_{SP}=7.13, Δ_{EO}=5.10). More dedicated and effective defense mechanisms in the future are still in demand.

Strengthen the connections among groups. One main reason behind the success of NIFA in fairness attack is the guidance of the homophily-increase principle during node injection. The ablation study in Appendix 5.4 also provides empirical evidence for this claim. As we analyze in Section 4.2, NIFA will lead to the increase of node-level homophily-ratio, which means more sensitive-related information will be aggregated and enlarged within the group. Given this, we believe that an effective defensive strategy is to strengthen the information propagation among different sensitive groups, thus preventing the risks of information cocoons [22, 25] and fairness issues.

Fairness auditing. At last, we find that a crucial assumption in NIFA and other research [11, 13, 41] is that GNN model administrators will only audit the utility metrics of the models, such as accuracy or F1-scores. Therefore, as long as attackers can ensure that the model utility is not affected excessively, it will be hard for administrators to realize the attack. Consequently, we strongly suggest that model administrators should also incorporate fairness-related metrics into their monitoring scopes, especially before model deployment or during the beta testing phase, thus, mitigating the potential broader negative impacts and social risks. For instance, if an updated GNN model suddenly demonstrates obvious fairness deterioration compared with the previous versions, the model administrators should be careful about the potential fairness attacks. However, the challenge of this approach mainly lies in the diverse definitions of fairness, such as group fairness [5, 23], individual fairness [6], etc., and group fairness based on different sensitive attributes [5, 38] or structures [21, 22] may further lead to different definitions. Therefore, model administrators might need prior knowledge or expertise to determine what kinds of fairness metrics to be included in their monitoring scopes.

7 Conclusion

In this work, we aim to examine the vulnerability of GNN fairness under adversarial attacks, thus mitigating the potential risks when applying GNNs in the real world. All existing fairness attacks on GNNs require modifying the connectivity or features of existing nodes, which is typically infeasible in reality. To this end, we propose a node injection-based poisoning attack namely NIFA. In detail, NIFA first proposes two novel principles for node injection operations and then designs multiple objective functions to guide the feature optimization of injected nodes. Extensive experiments on three datasets demonstrate that NIFA can effectively attack most mainstream GNNs and fairness-aware GNNs with an unnoticeable perturbation rate and utility degradation. Our work highlights the vulnerabilities of GNNs to node injection-based fairness attacks and sheds light on future research about robust fair GNNs and defensive mechanisms for potential fairness attacks.

Limitations. Firstly, NIFA is still under the settings of gray-box attacks, which requires accessibility to the labels and sensitive attributes. We acknowledge that such information may not always be available and we leave the extensions to the more realistic black-box attack settings as future work. Moreover, although we present some insights on the defense strategies of GNN fairness, more effective defense measures are still under-explored, calling for more future research efforts. At last, currently we only focus on fairness based on sensitive attributes, while neglecting the fairness based on graph structures. Since different fairness may stem from different sources, we leave this as our future work.

Acknowledgement

The work is supported by the National Natural Science Foundation of China (No.62172174). The authors would also like to thank Xiran Song and Jianxun Lian for their suggestions and contributions.

References

[1] C. Agarwal, H. Lakkaraju, and M. Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Proceedings of the Thirty-seventh Conference on Uncertainty in Artificial Intelligence, UAI 2021, Virtual Event, July 27-30, 2021*, volume 161 of *Proceedings of Machine Learning Research*, pages 2114–2124. AUAI Press, 2021.

[2] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann. Netgan: Generating graphs via random walks. In *Proceedings of the Thirty-fifth International Conference on Machine Learning, ICML 2018*, pages 610–619. PMLR, 2018.

[3] A. Chhabra, P. Li, P. Mohapatra, and H. Liu. Robust fair clustering: A novel fairness attack and defense framework. In *Proceedings of the Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[4] M. Coşkun and M. Koyutürk. Node similarity-based graph convolution for link prediction in biological networks. *Bioinformatics*, 37(23):4501–4508, 2021.

[5] E. Dai and S. Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining, WSDM 2021, Virtual Event, Israel, March 8-12, 2021*, pages 680–688. ACM, 2021.

[6] Y. Dong, J. Kang, H. Tong, and J. Li. Individual fairness for graph neural networks: A ranking based approach. In *Proceedings of the Twenty-seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2021, Virtual Event, Singapore, August 14-18, 2021*, page 300–310, New York, NY, USA, 2021. ACM.

[7] Y. Dong, N. Liu, B. Jalaian, and J. Li. EDITS: modeling and mitigating data bias for graph neural networks. In *Proceedings of the Thirty-first ACM Web Conference, WWW 2022, Virtual Event, Lyon, France, April 25-29, 2022*, pages 1259–1269. ACM, 2022.

[8] L. Du, X. Shi, Q. Fu, X. Ma, H. Liu, S. Han, and D. Zhang. Gbk-gnn: Gated bi-kernel graph neural networks for modeling both homophily and heterophily. In *Proceedings of the Thirty-first ACM Web Conference, WWW 2022, Virtual Event, Lyon, France, April 25-29, 2022*, page 1550–1558. ACM, 2022.

[9] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the Thirty-third International Conference on Machine Learning, ICML 2016*, pages 1050–1059. PMLR, 2016.

[10] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Proceedings of Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, NeurIPS 2017, Long Beach, CA, USA, December 4-9, 2017*, pages 1024–1034, 2017.

[11] H. Hussain, M. Cao, S. Sikdar, D. Helic, E. Lex, M. Strohmaier, and R. Kern. Adversarial inter-group link injection degrades the fairness of graph neural networks. In *Proceedings of the 2022 IEEE International Conference on Data Mining, ICDM 2022, Orlando, FL, USA, November 28 - Dec. 1, 2022*, pages 975–980. IEEE, 2022.

[12] M. Ju, Y. Fan, C. Zhang, and Y. Ye. Let graph be the go board: gradient-free node injection attack for graph neural networks via reinforcement learning. In *Proceedings of the Thirty-seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 4383–4390. AAAI Press, 2023.

[13] J. Kang, Y. Xia, R. Maciejewski, J. Luo, and H. Tong. Deceptive fairness attacks on graphs via meta learning. In *Proceedings of the Twelfth International Conference on Learning Representations, ICLR 2024, Vienna Austria, May 7-11, 2024*. OpenReview.net, 2024.

[14] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the Fifth International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017*. OpenReview.net, 2017.

 https://doi.org/10.52202/079017-2286

[15] J. Klicpera, A. Bojchevski, and S. Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *Proceedings of the Seventh International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[16] Y. Li, H. Chen, Z. Fu, Y. Ge, and Y. Zhang. User-oriented fairness in recommendation. In *Proceedings of the Thirtieth ACM Web Conference, WWW 2021, Virtual Event, Ljubljana, Slovenia, April 19-23, 2021*, pages 624–632. ACM, 2021.

[17] H. Ling, Z. Jiang, Y. Luo, S. Ji, and N. Zou. Learning fair graph representations via automated data augmentations. In *Proceedings of the Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[18] X. Liu, W. Jin, Y. Ma, Y. Li, H. Liu, Y. Wang, M. Yan, and J. Tang. Elastic graph neural networks. In *Proceedings of the Thirty-eigth International Conference on Machine Learning, ICML 2021, Virtual Event, July 18-24, 2021*, pages 6837–6849. PMLR, 2021.

[19] X. Liu, S. Si, J. Zhu, Y. Li, and C. Hsieh. A unified framework for data poisoning attack to graph-based semi-supervised learning. In *Proceedings of Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, December 8-14, 2019*, pages 9777–9787, 2019.

[20] Y. Liu, X. Ao, F. Feng, and Q. He. UD-GNN: uncertainty-aware debiased training on semi-homophilous graphs. In *Proceedings of the Twenty-eighth ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2022, Washington, DC, USA, August 14-18, 2022*, pages 1131–1140. ACM, 2022.

[21] Z. Liu, T. Nguyen, and Y. Fang. On generalized degree fairness in graph neural networks. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 4525–4533. AAAI Press, 2023.

[22] Z. Luo, H. Huang, J. Lian, X. Song, X. Xie, and H. Jin. Cross-links matter for link prediction: Rethinking the debiased gnn from a data perspective. In *Proceedings of Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10-16, 2023*, 2023.

[23] Z. Luo, J. Lian, H. Huang, H. Jin, and X. Xie. Ada-gnn: Adapting to local patterns for improving graph neural networks. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM 2022, Virtual Event / Tempe, AZ, USA, February 21-25, 2022*, pages 638–647. ACM, 2022.

[24] Y. Ma, S. Wang, T. Derr, L. Wu, and J. Tang. Graph adversarial attack via rewiring. In *Proceedings of the Twenty-seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2021, Virtual Event, Singapore, August 14-18, 2021*, pages 1161–1169. ACM, 2021.

[25] F. Masrour, T. Wilson, H. Yan, P. Tan, and A. Esfahanian. Bursting the filter bubble: Fairness-aware network link prediction. In *Proceedings of the Thirty-fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 841–848. AAAI Press, 2020.

[26] N. Mehrabi, M. Naveed, F. Morstatter, and A. Galstyan. Exacerbating algorithmic bias through fairness attacks. In *Proceedings of the Thirty-fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9, 2021*, pages 8930–8938. AAAI Press, 2021.

[27] B. Rozemberczki, C. T. Hoyt, A. Gogleva, P. Grabowski, K. Karis, A. Lamov, A. Nikolov, S. Nilsson, M. Ughetto, Y. Wang, T. Derr, and B. M. Gyori. Chemicalx: A deep learning library for drug pair scoring. In *Proceedings of the Twenty-eighth ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2022, Washington, DC, USA, August 14-18, 2022*, pages 3819–3828. ACM, 2022.

[28] U. Singer and K. Radinsky. Eqgnn: Equalized node opportunity in graphs. In *Proceedings of the Thirty-sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Virtual Event, February 22 - March 1, 2022*, pages 8333–8341. AAAI Press, 2022.

[29] D. Solans, B. Biggio, and C. Castillo. Poisoning attacks on algorithmic fairness. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020*, pages 162–177. Springer, 2020.

[30] X. Song, J. Lian, H. Huang, Z. Luo, W. Zhou, X. Lin, M. Wu, C. Li, X. Xie, and H. Jin. xgcn: An extreme graph convolutional network for large-scale social link prediction. In *Proceedings of the Thirty-second ACM Web Conference, WWW 2023, Austin, Texas, USA, April 30 - May 4, 2023*, pages 349–359. ACM, 2023.

[31] X. Song, J. Lian, H. Huang, M. Wu, H. Jin, and X. Xie. Friend recommendations with self-rescaling graph neural networks. In *Proceedings of the Twenty-eighth ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2022, Washington, DC, USA, August 14-18, 2022*, pages 3909–3919, 2022.

[32] Y. Sun, S. Wang, X. Tang, T. Hsieh, and V. G. Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *Proceedings of the Twenty-ninth ACM Web Conference, WWW 2020, Taipei, Taiwan, April 20-24, 2020*, pages 673–683. ACM / IW3C2, 2020.

[33] X. Tang, H. Yao, Y. Sun, Y. Wang, J. Tang, C. C. Aggarwal, P. Mitra, and S. Wang. Investigating and mitigating degree-related biases in graph convoltuional networks. In *Proceedings of the Twenty-ninth ACM International Conference on Information and Knowledge Management, CIKM 2020, Virtual Event, Ireland, October 19-23, 2020*, pages 1435–1444. ACM, 2020.

[34] S. Tao, Q. Cao, H. Shen, J. Huang, Y. Wu, and X. Cheng. Single node injection attack against graph neural networks. In *Proceedings of the Thirtieth ACM International Conference on Information and Knowledge Management, CIKM 2021, Virtual Event, Queensland, Australia, November 1-5, 2021*, pages 1794–1803. ACM, 2021.

[35] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

[36] J. Wang, M. Luo, F. Suya, J. Li, Z. Yang, and Q. Zheng. Scalable attack on graph data by injecting vicious nodes. *Data Min. Knowl. Discov.*, 34(5):1363–1389, 2020.

[37] N. Wang, L. Lin, J. Li, and H. Wang. Unbiased graph embedding with biased graph observations. In *Proceedings of the Thirty-first ACM Web Conference, WWW 2022, Virtual Event, Lyon, France, April 25-29, 2022*, pages 1423–1433. ACM, 2022.

[38] Y. Wang, Y. Zhao, Y. Dong, H. Chen, J. Li, and T. Derr. Improving fairness in graph neural networks via mitigating sensitive attribute leakage. In *Proceedings of the Twenty-eighth ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2022, Washington, DC, USA, August 14-18, 2022*, pages 1938–1948. ACM, 2022.

[39] F. Wu, A. H. S. Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *Proceedings of the Thirty-sixth International Conference on Machine Learning, ICML 2019, Long Beach, California, USA, Jun 9-15, 2019*, pages 6861–6871. PMLR, 2019.

[40] C. Yang, J. Liu, Y. Yan, and C. Shi. Fairsin: Achieving fairness in graph neural networks through sensitive information neutralization. In *Proceedings of the Thirty-eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Vancouver, Canada, February 20-27, 2024*, pages 9241–9249. AAAI Press, 2024.

[41] B. Zhang, Y. Dong, C. Chen, Y. Zhu, M. Luo, and J. Li. Adversarial attacks on fairness of graph neural networks. In *Proceedings of the Twelfth International Conference on Learning Representations, ICLR 2024, Vienna Austria, May 7-11, 2024*. OpenReview.net, 2024.

[42] H. Zhang, B. Wu, X. Yuan, S. Pan, H. Tong, and J. Pei. Trustworthy graph neural networks: Aspects, methods, and trends. *Proc. IEEE*, 112(2):97–139, 2024.

[43] X. Zhang, P. Bao, and S. Pan. Maximizing malicious influence in node injection attack. In *Proceedings of the Seventeenth ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*, pages 958–966. ACM, 2024.

[44] X. Zhang and M. Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. In *Proceedings of Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, December 6-12, 2020*, 2020.

[45] H. Zhu, G. Fu, Z. Guo, Z. Zhang, T. Xiao, and S. Wang. Fairness-aware message passing for graph neural networks. *CoRR*, abs/2306.11132, 2023.

 https://doi.org/10.52202/079017-2286

[46] Y. Zhu, J. Li, L. Chen, and Z. Zheng. The devil is in the data: Learning fair graph neural networks via partial knowledge distillation. In *Proceedings of the Seventeenth ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*, pages 1012–1021. ACM, 2024.

[47] X. Zou, Q. Zheng, Y. Dong, X. Guan, E. Kharlamov, J. Lu, and J. Tang. TDGIA: effective injection attacks on graph neural networks. In *Proceedings of the Twenty-seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2021, Virtual Event, Singapore, August 14-18, 2021*, pages 2461–2471. ACM, 2021.

[48] D. Zügner, A. Akbarnejad, and S. Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the Twenty-fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2847–2856. ACM, 2018.

A Ethical consideration

In this study, we propose a fairness attack towards GNN models via node injections. It is worth noting that the main purpose of this work is to reveal the vulnerability of current GNN models to fairness attacks, thereby inspiring and motivating both industrial and academic researchers to pay more attention to future potential attacks and enhancing the robustness of GNN fairness. We acknowledge the potential for our research to be misused or exploited by malicious hackers and to have real-world implications or even harm. Therefore, we will open-source our code under the CC-BY-NC-ND license[4] in the future, which means that the associated code cannot be used for any commercial purposes, and no derivatives or adaptations of the work are permitted. Additionally, we discuss some feasible defense mechanisms in Section 6, which we believe can to some extent mitigate the fairness attacks proposed in our work and hopefully inspire future fairness defenses.

B Attack settings

In this section, we would like to explicitly introduce our attack settings from the following aspects:

Attack stage. Attacks can be categorized into two types according to the time when the attacks take place [42]: *poisoning attack* and *evasion attack*. Poisoning attacks occur at the training phase of victim models, which will lead to poisoned models. In contrast, evasion attacks target the inference phase, and can not affect the model parameters. In this work, NIFA belongs to the poisoning attacks.

Attacker's knowledge. Generally, according to the knowledge of attackers, the attack methods can be categorized into three types including white-box attack, black-box attack and gray-box attack [42]. As we introduced in Section 3, we propose NIFA within the gray-box attack settings to make our attack more practical in the real world, which is also consistent with multiple prior research on GNN attacks [11, 32, 41]. Different from white-box attacks and black-box attacks, gray-box attacks mean that attackers can only access the training data, including the input graph $\mathcal{G}$, the labels $\mathcal{Y}$ and the sensitive attribute s of each node. Note that, the model architecture and parameters are invisible to attackers under the gray-box attack settings, which leads to that the attackers need to train a surrogate model in advance to assess the effectiveness of their proposed attacks.

Attacker's capability. One merit of NIFA is that there is no need for the attackers to have the authority to modify the existing graph structure, such as adding or deleting edges between existing real nodes, or modifying the existing real nodes' features. In contrast, NIFA injects $\mathcal{V}_I$ malicious nodes and poisons the graph $\mathcal{G}$ into $\mathcal{G}'$ to launch an attack, which is much more practical for the attackers in the real world. For example, in social networks the attackers only need to sign up multiple zombie accounts and interact with real accounts. Note that, different from some injection-based attack [12, 32], NIFA will not modify the training set and true node label set $\mathcal{Y}$, as such operations are typically infeasible in the real world. The intrinsic idea of NIFA is to impact the GNN training process through massage propagation on a poisoned graph.

Within the gray-box attack settings, we also assume that the attackers have sufficient computational resources and budget to train a surrogate model and have access to the real graph as input. Similar to prior attack methods [32], attackers are also required to set thresholds b and d for the number of injected nodes and their degrees respectively to make NIFA deceptive and unnoticeable.

C Proof

Lemma 1. *For target node u that will connect with injected nodes, our proposed node injection strategy will lead to the increase of node-level homophily-ratio $\mathcal{H}_u$.*

Proof. Given a target node u with k neighbors that have the same sensitive attribute with u, we simply assume it will connect with n injected nodes after node injection. Since all injected nodes in our proposed node injection belong to the same sensitive attribute as target node u, then the node-level homophily-ratio after node injection $\mathcal{H}'_u$ is:

$$\mathcal{H}'_u = \frac{k+n}{|\mathcal{N}_u|+n} \geq \frac{k}{|\mathcal{N}_u|} = \mathcal{H}_u \tag{16}$$

[4] https://creativecommons.org/licenses/by-nc-nd/4.0/

https://doi.org/10.52202/079017-2286

Algorithm 1 Training process of NIFA

Input: Graph $\mathcal{G}$. Hyper-parameters: node budget b, edge budget d, α and β, learning rates γ^S, γ^F.
Output: Poisoned graph $\mathcal{G}'$.

Initialize Bayesian GNN's parameter $\theta_{\mathcal{B}}$, surrogate model's parameter $\theta_{\mathcal{S}}$ and injected nodes' feature matrix $\mathbf{X}_I$.
Train the Bayesian GNN by Eq. (8) and Eq. (9).
Estimate the node uncertainty U in the original graph $\mathcal{G}$ by Eq. (10) and select the target nodes.
Connect the injected nodes with target nodes according to the homophily-increase principle.
for $iter = 1$ **to** max_iter **do**
  for $step = 1$ **to** max_step **do**
    Compute cross-entropy loss L_{CE} for $\mathcal{S}$ by Eq. (11).
    Update surrogate model: $\theta_{\mathcal{S}} \leftarrow \theta_{\mathcal{S}} + \gamma^S \cdot \nabla_{\theta_{\mathcal{S}}} L_{CE}$.
  end for
  for $step = 1$ **to** max_step **do**
    Compute L by Eq. (15).
    Update injected feature: $\mathbf{X}_I \leftarrow \mathbf{X}_I + \gamma^F \cdot \nabla_{\mathbf{X}_I} L$.
  end for
  Clamp $\mathbf{X}_I$ between min and max of $\mathbf{X}$.
end for
if $\mathbf{X}$ is discrete **then**
  Round $\mathbf{X}_I$ into integer.
end if
return $\mathcal{G}'$

Such inequality holds true when $|\mathcal{N}_u| \geq k$. □

D Implementation algorithm

Due to space limitation, here we provide the complete training process of NIFA in Algorithm 1. The training process and evaluation process are also literally described in Section 4.4.

E Additional descriptions on victim models

In this part, we give an introduction to the victim models used in our experiment.

- **GCN** [10]: Borrowing the concept of convolution from the computer vision domain, GCN employs convolution operation on the graph from a spectral perspective to learn the node embeddings.
- **GraphSAGE** [14]: Given the potential neighborhood explosion issues in GCN, GraphSAGE samples a fixed number of neighbors at each layer during neighborhood aggregation, which greatly improves the training efficiency.
- **APPNP** [15]: Inspired by PageRank, APPNP decouples the prediction and propagation in the training process, which resolves inherent limited-range issues in message-passing models without introducing any additional parameters.
- **SGC** [39]: SGC empirically finds the redundancy of non-linear activation function, and achieves comparable performance with much higher efficiency.
- **FairGNN** [5]: Through proposing a sensitive attribute estimator and an adversarial learning module, FairGNN maintains high classification accuracy while reducing unfairness in scenarios with limited sensitive attribute information.
- **FairVGNN** [38]: FairVGNN discovers the leakage of sensitive information during information propagation of GNN models, and generates fair node features by automatically identifying and masking sensitive-correlated features.
- **FairSIN** [40]: Instead of filtering out sensitive-related information for fairness, FairSIN deploys a novel sensitive information neutralization mechanism. Specifically, FairSIN will learn to introduce additional fairness facilitating features (F3) during message propagation to neutralize sensitive information while providing more non-sensitive information.

F Additional descriptions on baselines

In this section, we will introduce more details about the baselines used in our experiments.

- **AFGSM** [36]: As a node injection-based attack, AFGSM designs an approximation strategy to linearize the victim model and then generates adversarial perturbation efficiently. In general, AFGSM is scalable to much larger graphs.
- **TDGIA** [47]: Aiming at attacking the model performance, TDGIA first introduces a topological edge selection strategy to select targeted nodes for node injection, and then generate injected features through smooth feature optimization.
- **$\mathbf{G^2A2C}$** [12]: Similar to NIPA [32], $\mathrm{G^2A2C}$ also proposes a node injection attack through reinforcement learning – Actor Critic. Specifically, $\mathrm{G^2A2C}$ devises three core modules including *Node Generator*, *Edge Sampler*, *Value Predictor* to model the full process of node injection.
- **FA-GNN** [11]: To the best of our knowledge, FA-GNN is the first work to conduct a fairness attack on GNNs. In detail, FA-GNN empirically discovers several edge injection strategies, which could impact the GNN fairness with slight utility compromise.
- **FATE** [13]: FATE is a meta-learning-based fairness attack framework for GNNs. To be concrete, FATE formulates the fairness attacks as a bi-level optimization problem, where the lower-level optimization guarantees the deceptiveness of an attack while the upper-level optimization is designed to maximize the bias functions.
- **G-FairAttack** [41]: As a poisoning fairness attack, G-FairAttack consists of two modules, including a surrogate loss function and following constrained optimization for deceptiveness. Like FATE and FA-GNN, G-FairAttack also belongs to graph modification attacks, i.e., the original link structure between existing nodes will be modified during the attack.

G Reproducibility details

The implementation details are first provided in Section 5.1 in the original paper. Here we would like to provide more implementation details from the following four aspects:

Environment. All experiments are conducted on a server with Intel(R) Xeon(R) Gold 5117 CPU @ 2.00GHz and 32 GB Tesla V100 GPU. The experimental environment is based on Ubuntu 18.04 with CUDA 11.0, and our implementation is based on Python 3.8 with PyTorch 1.12.1 and Deep Graph Library (DGL) 1.1.0.

Victim models. For all victim models, we set the learning rate as 0.001, and the hidden dimension as 128 after careful tuning. For most victim models, the dropout ratio is set to 0 by default. Specifically, for GCN and GraphSAGE, the layer number is set to 2, and we employ mean pooling aggregation for GraphSAGE. For SGC, the hop-number k is set to 1. For APPNP, following the suggestions in the official paper, the teleport probability α is set as 0.2, and we set the iteration number k as 1 after careful tuning. For FairGNN, we employ the GAT as the backbone model, which shows a better performance in the original paper, and set the dropout ratio as 0.5. The objective weights α and β are set as 4 and 0.01, respectively. For FairVGNN, GCN is set to be the backbone model, and we set the dropout ratio as 0.5. The prefix cutting threshold ϵ is searched from $\{0.01, 0.1, 1\}$, and the mask density α is 0.5. The epochs for the generator, discriminator, and classifier are selected from $\{5, 10\}$ as suggested in the official implementation. For FairSIN, we also utilize GCN as the backbone model, and we set the weight of neutralized feature δ as 4 after tuning, and set the hidden dimension as 128 for a fair comparison. All learning rates involved in FairSIN are set to 0.001 after careful fine-tuning and other hyper-parameters are set according to the official implementations[5].

Baselines details. For all baselines that involve graph structure manipulation, the numbers of injected nodes and edges on three datasets are made identical to the settings of our model for a fair comparison. For methods that only inject new nodes, such as AFGSM, TDGIA and G2A2C, we require the number of injected nodes and average degree of injected nodes to be the same as ours. For methods that require to modify the graph structure, such as FA-GNN, FATE and G-FairAttack, we set the number of modified edges to be the same with our injected edges, i.e. the added edges between injected nodes and original nodes. To be concrete, for the AFGSM, across all datasets,

[5]https://github.com/BUPT-GAMMA/FairSIN/

https://doi.org/10.52202/079017-2286

we utilize the direct attack setting and allow the model to perturb node features. For TDGIA, the weights k_1, and k_2, which are used for calculating topological vulnerability are set to 0.9 and 0.1, respectively. For G^2A2C, the temperature of Gumbel-Softmax is set to 1.0, the discount factor is set to 0.95, and the Adam optimizer is utilized with a learning rate of 10^{-4}. Furthermore, we adopt early stopping with a patience of three epochs. For FA-GNN, we utilize the DD strategy, which has the best attack performance in the original paper. For FATE, the perturbation mode is *filp* and the attack step is set to 3 as the official implementation[6] suggested. For G-FairAttack, the proportion of candidate edges is set to 0.0001 for fast computation on our datasets, and we follow the default settings in the official repository[7] for other hyper-parameters. It is worth noting that, G-FairAttack can be utilized as either an evasion attack or a poisoning attack according to the original paper, and we follow the poisoning attack settings to be the same with NIFA.

Details for implementing NIFA. We employ a two-layer GCN model as the surrogate model, whose hidden dimension is set to 128, and the dropout ratio is set to 0. The learning rates for optimizing the surrogate model and injected features are both set to 0.001 after tuning. The sampling times T of the Bayesian Network is 20. The objective weights α and β for all datasets are searched from $\{0.005, 0.01, 0.02, 0.05, 0.1, 0.2\}$ and $\{2, 4, 8, 16\}$, respectively. The number of injected nodes is set to 1% of the number of labeled nodes in the original graph, and the degree of injected nodes d is set to be 50, 50, and 24 on three datasets, respectively, which are the average node degrees in the original graph. As for the uncertainty threshold $k\%$, we search in a range of $\{0.1, 0.25, 0.5, 0.75\}$. The hyper-parameter analysis will be further elaborated in Appendix H.3. The max_iter and max_step in Algorithm 1 and other proposed hyper-parameters for each dataset are summarized in Table A1.

Table A1: Hyper-parameters statistics

Notations	Pokec-z	Pokec-n	DBLP
α	0.01	0.01	0.1
β	4	4	8
b	102	87	32
d	50	50	24
k	0.5	0.5	0.5
max_step	50	50	50
max_iter	20	20	10

Table A2: Comparison of the statistics on the clean graph (Perturbation rate is 0.00) and the graphs poisoned by NIFA with different perturbation rates. For better illustration, we also provide the relative rate of change ($|\Delta|$) in the table.

	Perturbation Rate	Gini Coefficient		Assortativity		Power Law Exp.		Triangle Count		Rel. Edge Distr. Entropy		Characteristic Path Length	
		value	$\|\Delta\|$	value	$\|\Delta\|$	value	$\|\Delta\|$	value	$\|\Delta\|$	value	$\|\Delta\|$	value	$\|\Delta\|$
Pokec-z	0.00	0.5719	0	0.2108	0	1.6152	0	767,688	0	0.0259	0	4.5208	0
	0.01	0.5696	0.41%	0.2100	0.40%	1.6103	0.30%	767,787	0.01%	0.0259	0.13%	4.5062	0.32%
	0.02	0.5677	0.75%	0.2093	0.74%	1.6064	0.54%	767,917	0.03%	0.0259	0.24%	4.4951	0.57%
	0.05	0.5627	1.61%	0.2071	1.79%	1.5967	1.15%	768,287	0.08%	0.0260	0.42%	4.4732	1.05%
Pokec-n	0.00	0.5634	0	0.1978	0	1.6609	0	531,590	0	0.0296	0	4.6365	0
	0.01	0.5613	0.37%	0.1958	0.96%	1.6557	0.31%	531,695	0.02%	0.0296	0.08%	4.6215	0.32%
	0.02	0.5595	0.68%	0.1939	1.96%	1.6513	0.58%	531,776	0.03%	0.0296	0.13%	4.6103	0.56%
	0.05	0.5555	1.39%	0.1895	4.15%	1.6409	1.20%	532,089	0.09%	0.0297	0.20%	4.5902	1.00%
DBLP	0.00	0.4137	0	0.1335	0	2.0730	0	73,739	0	0.0820	0	6.4867	0
	0.01	0.4137	0.01%	0.1316	1.44%	2.0618	0.54%	73,744	0.01%	0.0818	0.27%	6.3627	1.87%
	0.02	0.4148	0.27%	0.1304	2.32%	2.0528	0.98%	73,751	0.02%	0.0815	0.69%	6.2886	3.02%
	0.05	0.4208	1.71%	0.1311	1.83%	2.0307	2.04%	73,774	0.05%	0.0801	2.34%	6.1789	4.72%

H Additional experiments

H.1 In-depth analysis of the poisoned graph

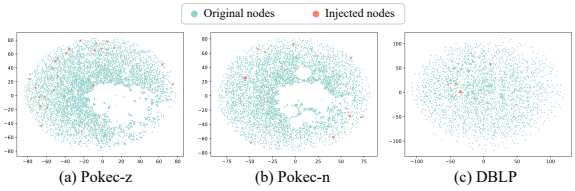
As a poisoning attack, it is crucial to ensure that after the attack, the poisoned graph's characteristics should remain similar to that of the clean graph. Otherwise, administrators can easily notice the attack through abnormal graph structures or node features. To this end, we conduct an in-depth analysis of the poisoned graph by NIFA from the following two perspectives:

[6]https://github.com/jiank2/FATE

[7]https://github.com/zhangbinchi/G-FairAttack

Structural analysis. Similar to prior work [2, 32], we investigate several key graph characteristics in this section, including *Gini Coefficient*, *Assortativity*, *Power Law Exponent*, *Triangle Count*, *Relative Edge Distribution Entropy* and *Characteristic Path Length*, whose definitions and implementations can be found here[8]. The graph statistics under different perturbation rates are shown in Table A2. It can be concluded that: (1) Thanks to the low perturbation rate required by NIFA, the poisoned graphs share quite similar characteristics with clean graphs. Under most cases, the relative rate of change $|\Delta|$ is smaller than 1%, especially when the perturbation rate is small. (2) With the increase of perturbation rate, the poisoning attack becomes more obvious, which can be verified by the increased $|\Delta|$ on all graph structure statistics. This observation further supports the necessity of requesting a low perturbation rate for an attack. (3) Several key statistics showcase a consistent trend across the three datasets as the perturbation rate increases. For example, with the increase in attack intensity, the degree assortativity consistently decreases on three datasets, indicating that there are more connections between nodes with significantly different degrees due to the injection of malicious nodes. Similar observations can also be found in key statistics like *Triangle Count*, *Gini Coefficient* and *Characteristic Path Length*, which implies that these statistics can be potentially utilized for fairness attack defense and auditing.

Feature analysis. Besides graph structural analysis, we also conduct experiments to analyze the nodes' features of the poisoned graph. In detail, after conducting fairness attacks through NIFA, for both labeled nodes in the real graph and injected nodes, we illustrate their features' visualization results based on t-SNE [35] in Figure A1. Note that, since the number of injected nodes is much smaller than that of the original labeled nodes, we slightly increase the scatter size of injected nodes for better visualization.



(a) Pokec-z (b) Pokec-n (c) DBLP

Figure A1: T-SNE visualization of poisoned graph's node features

It can be seen that, 1) the feature layouts of Pokec-z and Pokec-n are quite similar, since both datasets are sampled from the same social graph. 2) On all three datasets, the distributions of injected nodes by NIFA are relatively diverse, which have no obvious patterns, and are hard to recognize from the original labeled nodes. Such observation further verifies the feature deceptiveness of NIFA.

H.2 Alternative analysis in node selection

During the node injection process, NIFA proposes an uncertainty-maximization principle and selects target nodes with the highest model uncertainty score. In fact, besides estimating model uncertainty, there are other alternative methods for finding vulnerable nodes. For example, TDGIA [47] introduces the concept of "topological vulnerability", and selects nodes with low degrees as target nodes. In this part, we also conduct experiments with selecting target nodes with the lowest degree as a variant of NIFA. The victim model is GCN.

Table A3: Attack performance (%) with different target node selection strategies. The best attack performance is **bolded**.

	Pokec-z			Pokec-n			DBLP		
	Acc.	Δ_{SP}	Δ_{EO}	Acc.	Δ_{SP}	Δ_{EO}	Acc.	Δ_{SP}	Δ_{EO}
Clean	71.22	7.13	5.10	70.92	0.88	2.44	95.88	3.84	1.91
Degree	70.50	15.76	14.01	69.77	**12.39**	**11.93**	94.72	6.30	15.10
Uncertainty	70.50	**17.36**	**15.59**	70.12	10.10	9.85	93.37	**13.49**	**20.33**

The results are shown in Table A3. It can be seen that, degree-based node selection (denoted as "Degree") also achieves promising attack performance compared with NIFA (denoted as "Uncertainty"), which indicates that model uncertainty is not the only feasible criterion for target node selection. However, the degree-based selection may be ineffective when the graph is extremely dense and has few low-degree nodes, which deserves more careful study in the future.

H.3 Hyper-parameter analysis

To better understand the different roles of hyper-parameters in NIFA, we study the impact of α, β, b, and k in this part where GCN is employed as the victim model.

[8] https://github.com/danielzuegner/netgan/tree/master

https://doi.org/10.52202/079017-2286

The impact of α and β. As the weights of objective functions in Eq. (15), the impacts of α and β are illustrated in Figure A2 and Figure A3, respectively. For three datasets, they perform best at different α and β values, indicating that the appropriate values of α and β depend on the datasets. In contrast, the accuracy remains relatively stable with changes in α and β, which indicates that our attack is utility-friendly.

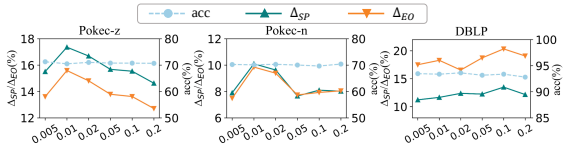


Figure A2: The impact of α on three datasets

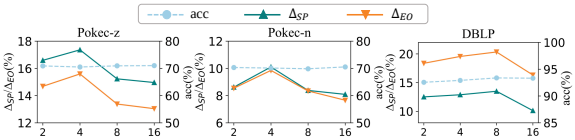


Figure A3: The impact of β on three datasets

The impact of node injection budget b. We illustrate the impact of the node injection budget b in Figure A4, where the x-axis denotes the perturbation rate, i.e. the proportion of b to the number of labeled nodes in the original graph. As expected, the increase of b leads to better fairness attack performance in most cases with more obvious accuracy compromise. Empirically, 0.01 will be a near-optimal choice while being unnoticeable.

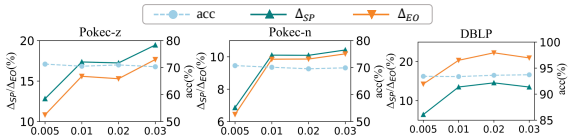


Figure A4: The impact of perturbation rate on three datasets

The impact of uncertainty threshold k. The impact of k is illustrated in Figure A5, where we tune $k\%$ in a set of $\{0.1, 0.25, 0.5, 0.75\}$. It can be seen that all datasets show a similar preference for $k\%$ with an optimal value of 0.5. Intuitively, with a higher $k\%$, it is hard for NIFA to attack the target nodes with low uncertainty, whereas, with a lower $k\%$, the impact of the attack will be limited by the insufficient number of targeted nodes.

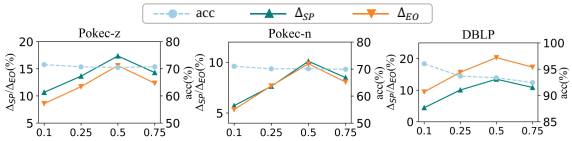


Figure A5: The impact of k% on three datasets

H.4 Efficiency analysis

To evaluate the efficiency and scalability of NIFA, we would like to compare the attack time cost and memory cost between NIFA and two competitive baselines including FATE [13] and G-FairAttack [41]. To be fair, the attack budgets for the three models are set to be the same in advance. We report their memory cost and time cost for finishing poisoning attacks on Pokec-z, Pokec-n and DBLP in Table A4. The environment configurations for our experiments are introduced in Section G. It can be seen that, NIFA can be successfully deployed on three datasets with acceptable time cost and memory cost. In contrast, both FATE and G-FairAttack face scalability issues, especially when facing large graphs such as Pokec-z and Pokec-n. As shown in Table 1, both Pokec-z and Pokec-n have much more edges compared with DBLP, which causes FATE to report OOM errors and G-FairAttack to fail to finish the attack within 72 hours.

Table A4: Efficiency Analysis of NIFA

	Pokec-z		**Pokec-n**		**DBLP**	
	Time	Memory	Time	Memory	Time	Memory
FATE	–	OOM	–	OOM	87.13 s	32258 MB
G-FairAttack	>72 h	7865 MB	>72 h	7329 MB	93048.20 s	2445 MB
NIFA	137.04 s	4319 MB	167.07 s	4213 MB	127.52 s	5829 MB

H.5 Robustness to defense strategies

Besides the defense discussions in Section 6, we analyze the effectiveness of conventional defense strategies in this section. In detail, two classic GNN defense models – GNNGuard [44] and ElasticGNN [18] are deployed to test their defense capabilities against NIFA. It is worth noting that, both defense models are originally designed for the attacks on prediction utility. We conduct experiments on three datasets, and the victim model is GCN as default. As shown in Table A5, both defense models only maintain the utility performance and failed to fully eliminate the impact of fairness at-

https://doi.org/10.52202/079017-2286

Table A5: The performance of defense strategies against NIFA.

	Pokec-z			Pokec-n			DBLP		
	Acc	Δ_{SP}	Δ_{EO}	**Acc**	Δ_{SP}	Δ_{EO}	**Acc**	Δ_{SP}	Δ_{EO}
Clean	71.22±0.28	7.13±1.21	5.10±1.28	70.92±0.66	0.88±0.62	2.44±1.37	95.88±1.61	3.84±0.34	1.91±0.75
NIFA	70.50±0.30	17.36±1.16	15.59±1.08	70.12±0.37	10.10±2.80	9.85±1.97	93.37±0.28	13.49±2.83	20.33±3.82
GNNGuard	71.50±0.35	18.13±2.31	16.32±1.44	70.55±0.43	13.82±1.76	13.38±1.42	95.73±0.28	7.42±1.82	14.94±1.58
ElasticGNN	71.36±0.20	13.53±0.92	11.36±1.23	70.32±0.28	6.90±1.02	6.44±1.32	94.22±0.37	9.76±1.65	17.04±2.01

tacks. For example, compared with the performance after the attack, although ElasticGNN reduces the Δ_{SP} from 17.36% to 13.53%, the fairness is still worse than that before the attack – 7.13%. We believe that the main reason behind such observation is that all these defense methods are designed for utility-targetted attacks, whose objectives are totally different from fairness attacks like NIFA. Such observations also indicate that more effective defense mechanisms are still in demand for fairness attacks on GNNs.

H.6 Performance with limited training ratio

As introduced in Section 5.1, our datasets are all collected from the previous work – FA-GNN [11], and our train / val / test ratios are set to be consistent with its settings, which is 50% / 25% / 25%, respectively. However, in more realistic scenarios the percentage of labeled training nodes might be significantly smaller due to the heavy cost of annotation. To evaluate the effectiveness of NIFA under such settings, we decrease the training ratio from 50% to 25%, 10% and 5%, respectively. The attack performance of NIFA on Pokec-z and Pokec-n is shown in Table A6. It can be seen that, even with much fewer labeled training nodes, NIFA still consistently demonstrates promising attack performance.

Table A6: The performance of NIFA with more limited training nodes. The victim model is GCN.

		Pokec-z			Pokec-n		
		Acc	Δ_{SP}	Δ_{EO}	**Acc**	Δ_{SP}	Δ_{EO}
25%	**before**	71.47±0.59	6.26±1.63	4.23±1.55	70.16±0.45	1.03±0.70	2.87±0.53
	after	70.48±0.41	**15.16±1.64**	**13.18±1.77**	69.49±0.31	**9.79±2.02**	**9.52±1.84**
10%	**before**	70.40±0.28	6.29±2.77	4.63±2.43	70.25±0.57	2.56±1.32	3.72±0.32
	after	70.14±0.26	**15.26±3.02**	**13.36±3.06**	69.25±0.62	**11.94±1.78**	**12.00±1.75**
5%	**before**	70.03±0.54	4.90±1.98	3.75±1.01	68.61±1.15	3.79±2.30	4.21±1.94
	after	69.49±0.65	**14.35±2.03**	**12.42±2.04**	68.05±1.63	**10.31±5.68**	**10.60±5.04**

https://doi.org/10.52202/079017-2286

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: Please refer to the abstract and Section 1.

   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 refer to Section 7 for our discussions on the limitations.

   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: Please refer to Section 4.2 and Appendix C for the complete proof.

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: Please refer to Section 5.1 and Appendix G for reproducibility details.

   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?

https://doi.org/10.52202/079017-2286

Answer: [Yes]

Justification: The url of our code and data is released in the Abstract. Section 5.1 and Appendix G also describe our reproducibility details.

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 refer to Section 5.1 and Appendix G for experimental details.

   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: We report the standard deviations of our experiments after repeating each experiment five times, which is also explained in the text. Please refer to Section 5.

   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: We describe the computer resources and environment in Appendix G.

   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: We thoroughly discuss the ethical consideration in Appendix A, in which the potential social harms are acknowledged and the code license are provided to avoid malicious use. We also discuss the feasible defensive measures in Section 6 to mitigate potential harmful consequences from our research.

   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: We thoroughly discuss the ethical consideration in Appendix A, in which the potential social harms are acknowledged and the code license are provided to avoid malicious use. We also discuss the feasible defensive measures in Section 6 to mitigate potential harmful consequences from our research.

    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.

https://doi.org/10.52202/079017-2286

- 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: [Yes]

    Justification: We describe the code license that will be released with our model in Appendix A.

    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: All models and baselines from existing assets are properly cited.

    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: [Yes]

    Justification: We thoroughly discuss the training details, license, and limitations of our proposed model in this paper.

    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: Our paper does not involve crowdsourcing nor research with human subjects.

    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: Our paper does not involve crowdsourcing nor research with human subjects.

    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.

https://doi.org/10.52202/079017-2286

- 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.