FairGNN-Bagging: Balancing model performance and algorithmic fairness in graph neural networks using ensemble learning

Juan Carlos Elias Obando Valdivia[1], **Marcel Rodrigues de Barros**[1], **Artur Jordão Lima Correia**[1], **Anna Helena Reali Costa**[1]

[1]Escola Politécnica – Universidade de São Paulo (USP)

{juan.valdivia, marcel.barros, arturjordao, anna.reali}@usp.br

Abstract. *Graph Neural Networks (GNNs) are powerful tools for learning from graph-structured data, with applications in social networks, bioinformatics, and fraud detection. However, GNNs can inherit and amplify biases present in data, leading to unfair predictions regarding sensitive attributes such as gender or race. To mitigate these issues, fairness-aware frameworks like NIFTY and its variant with Biased Edge Dropout have been proposed. In this work, we introduce FairGNN-Bagging, an ensemble framework that combines NIFTY with Bagging to improve fairness in node classification tasks. Our method uses graph perturbations to generate an augmented graph dataset to train an ensemble of GNNs in parallel. These models are then aggregated via majority voting or fairness-aware selective voting. We evaluate our approach on three real-world datasets, showing that it achieves better fairness metrics while preserving or improving predictive performance.*

1. Introduction

In complex problems such as social network analysis, recommendation systems, bioinformatics, and fraud detection, graphs have demonstrated a remarkable ability to represent different possible relationships between entities encountered in these problems. However, most machine learning algorithms are designed for simple sequences or grids, while graphs have complex topological structures and can have multimodal characteristics [Hamilton et al. 2017b]. To deal with these difficulties, Graph Neural Networks (GNNs) can be used to learn vector representations of nodes called embeddings. In this way, deep neural networks can be generalized to model graph-structured data to solve several graph learning tasks such as node classification, link prediction, and graph classification. GNNs adopt the message-passing mechanism that aggregates the information of neighbors of a given node to update this node representation. This mechanism allows the learned embeddings to capture both the node attributes and its local structure [Hamilton et al. 2017b].

The performance of GNNs in terms of accuracy and generalization is often prioritized in the literature. However, it is equally important to design more trustworthy algorithms that also consider algorithmic performance with respect to fairness, robustness, privacy, and explainability [Dong et al. 2023].

Previous research has highlighted that GNNs not only inherit undesirable biases present in the data but can also amplify them through their message-passing mechanisms [Dong et al. 2023, Dai et al. 2024, Zhang et al. 2024]. These biases, often linked to sensitive attributes such as gender and race, can lead GNNs to make unfair decisions, with the message-passing process and the topological structure of the graph further intensifying these effects [Dong et al. 2022].

In this work we focus on predictions over graph nodes. In this context, a dataset is formed by a graph and each example is a node of a graph. The node classification task consists of assigning a label for each node in a graph. Success in this task depends heavily on the graph structure, the node attributes, as well as the effectiveness of the GNN model that will be used to capture local and global patterns in the graph [Hamilton et al. 2017a].

In the literature, there is a taxonomy that categorizes the existing types of fairness for GNNs as group fairness, individual fairness, and counterfactual fairness [Dong et al. 2023]. Group fairness ensures that demographic groups do not receive discriminatory treatment [Cynthia et al. 2012]. On the other hand, individual fairness aims to ensure that two similar individuals receive similar algorithmic treatment [Kang et al. 2020]. Counterfactual fairness, in turn, is inspired by the development of counterfactual learning [Kusner et al. 2017], and measures fairness from the perspective of causal inference. Thus, it is expected that when the value of the sensitive attribute is changed in a counterfactual (unobservable) scenario, the predictions given by the model remain the same. For example, in a credit scoring context, in which race can be a sensitive attribute, it is expected that if a person of a certain race received a high credit approval based on their characteristics, they would also have obtained a high credit approval in a counterfactual scenario in which their race was different.

The most widely used node classification framework that deals with counterfactual fairness for graphs is NIFTY [Agarwal et al. 2021], which aims to learn fair representations for predictions while ensuring stability. Stability here refers to the property of not changing the predictions when the graph structure and node attributes are subjected to small perturbations. In NIFTY, two new graphs are created from the original graph, a counterfactual graph and a noisy graph. The counterfactual graph is generated by flipping the value of the sensitive attribute of all nodes. The noisy graph is created by dropping edges and slightly modifying the values of the non-sensitive attributes of the nodes. In NIFTY framework, a triple loss function maximizes the similarity of the embeddings of the original graph and the two new graphs created. Experiments show that NIFTY significantly enhances fairness and stability without sacrificing predictive performance in terms of AUROC and F1-score [Agarwal et al. 2021].

According to Spinelli et al.(2021), nodes with similar characteristics are more likely to be connected to each other. This property is called homophily. From a fairness perspective, homophily associated with sensitive attributes exerts a direct influence on the GNN prediction process, potentially introducing and amplifying inequalities in the predictions [Spinelli et al. 2021]. In this direction, Spinelli et al.(2021) proposed FairDrop, based on the hypothesis that removing homophilous edges can help improve group fairness. Franco et al. (2024) simplify the formulation of FairDrop to empower NIFTY's framework using a technique called Biased Edge Dropout that balances homophilous and heterophilous sensitive connections.

Following another approach line, some proposals found in the literature use GNN ensembles to improve the algorithmic effectiveness of individual

GNNs [Nagarajan et al. 2022, Lin et al. 2022, Wei et al. 2023]. In this line, there is no concern about ensuring fairness in decisions, since the interest is only in predictive performance. Among the GNN architectures used for these ensembles are GCN [Kipf and Welling 2017] and GraphSage [Hamilton et al. 2017a], which are GNN frameworks that also only attempt to improve algorithmic predictive performance.

In this work, we propose FairGNN-Bagging, a novel ensemble-based framework that improves the trade-off between fairness and predictive performance in node classification tasks over graphs. Our contributions are as follows: (1) we integrate the NIFTY framework into an ensemble strategy, using graph perturbations to create diverse training graphs and train GNNs in parallel, aggregated via majority or fairness-aware voting; (2) we design an augmented graph dataset method using structured perturbations (DropEdge and Biased Edge Dropout) to promote ensemble diversity while preserving relevant information; (3) we show that FairGNN-Bagging improves fairness metrics (counterfactual fairness, demographic parity, and equal opportunity) across three real-world datasets, without compromising—often improving—AUROC and F1-score and (4) we provide evidence that FairGNN-Bagging achieves a more stable and reliable prediction behavior compared to single GNN models.

This paper is organized as follows: Section 2 presents the foundational concepts. Section 3 reviews the related work in graph counterfactual fairness and ensemble learning for GNNs. Our proposal is detailed in Section 4, while Section 5 offers the experimental results. Finally, Section 6 offers the conclusions.

2. Preliminaries

Let the unweighted, undirected graph $G = (V, E, X)$ be denoted by a set of N nodes $V = \{v_1, v_2, ..., v_N\}$, a set of edges $E \subseteq V \times V$ and a set of attribute vectors $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ corresponding to all nodes in the set V, where each $\mathbf{x}_v \in X$ is a d-dimensional vector and represents the attribute values for node $v \in V$. The edges of the graph G are represented by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ in which $\mathbf{A}_{uv} = 1$ if there exists an edge $e \in E$ between nodes u and v, and $\mathbf{A}_{uv} = 0$ otherwise. The node classification task consists of assigning a label $\hat{Y}_v$ to each node $v \in V$.

2.1. Information Processing in GNNs

In a GNN, information processing is typically structured into three interconnected stages: message passing, embedding generation, and prediction [Kipf and Welling 2017]. The first stage, message passing, involves iterative communication between nodes and their neighbors. At each iteration, nodes aggregate information from their direct neighbors, combining it with their own attributes to update their representations. This process leverages the graph structure to capture local connectivity patterns and attribute dependencies, allowing nodes to incorporate context from their surrounding network. As the iterations progress, the node representations evolve into embeddings $\mathbf{z}_v$, i.e., $ENC(v) = \mathbf{z}_v$, which are latent vectors that not only encode local neighborhood information, but, depending on the depth of the network, also capture broader, more global graph structures. These embeddings serve as a compact, informative representation of each node within the graph, synthesizing both its own attributes and the structural and attribute-based context provided by its neighbors [Kipf and Welling 2017, Hamilton et al. 2017a].

Finally, in the prediction stage, these learned embeddings are passed through a classifier, often implemented as a fully connected layer. This classifier f maps the embeddings to output labels, i.e., $\hat{Y}_v = f(\mathbf{z}_v)$. By combining these stages, GNNs enable effective learning from graph-structured data, offering a flexible framework for the node classification task [Kipf and Welling 2017, Hamilton et al. 2017a].

Consider a GNN with K layers and $\mathbf{h}_v^{(1)}, \mathbf{h}_v^{(2)}, ..., \mathbf{h}_v^{(K)}$ are the outputs of each of these layers for each node $v \in V$. Also consider the message (MSG), aggregation (AGG) and combination ($COMB$) operators [Wu et al. 2020]. Formally the k-th layer of the GNN for a node v is formulated as follows:

$$\begin{aligned} \mathbf{m}_u^{(k)} &= MSG(\mathbf{h}_u^{(k-1)}), u \in \{N_v \cup v\} \\ \mathbf{h}_v^{(k)} &= COMB(AGG\left(\mathbf{m}_u^{(k)} | u \in N_v)\right), \mathbf{m}_v^{(k)}), \end{aligned} \tag{1}$$

where N_v is the set of neighbors of node v. Each node v collects the messages of its neighboring nodes $u \in N_v$ using the MSG operator. These messages can be the neighbors' attribute vectors or some transformation function of these vectors. The AGG operator aggregates the messages of all neighbors of a given node v using, for example, sum, average, or a nonlinear function. After aggregating the neighbors' messages, the $COMB$ operator combines the result of the aggregation with the message of node v itself to update its own representation in layer k using, for example, a linear transformation followed by a nonlinear activation function. The final representation of node v in the last layer is $\mathbf{z}_v = \mathbf{h}_v^{(K)}$ [Wu et al. 2020]. There are several classical architectures in the literature for the layers of a GNN, among them are GCN (*Graph Convolutional Networks*) [Kipf and Welling 2017] and GraphSage [Hamilton et al. 2017a].

2.2. Counterfactual Fairness

In this work, we focus on counterfactual fairness, a concept in fair machine learning that ensures decisions made by a model are unbiased with respect to sensitive attributes (e.g., race, gender, age) when considering counterfactual scenarios. Counterfactual fairness is closely related to the counterfactual learning concept [Kusner et al. 2017], as both rely on counterfactual reasoning to analyze and predict outcomes in alternative scenarios. However, counterfactual learning uses these scenarios to optimize model performance, while counterfactual fairness ensures ethical and unbiased decisions.

Counterfactual fairness is deeply connected to Judea Pearl's structural causal model (SCM) [Pearl 2009] because SCM provides the mathematical and conceptual framework for reasoning about counterfactual and causal relationships. A causal model consists of a causal graph and structural equations. The causal graph is a directed acyclic graph in which each node represents a variable and each directed edge represents a causal relationship. The structural equations describe these causal relationships between the variables. The idea of counterfactual fairness is that if the value of the sensitive attribute is changed in a counterfactual scenario, the predictions given by the algorithm remain the same. The counterfactual value $\hat{Y}_{S \leftarrow s'}$ represents a hypothetical scenario for the output $\hat{Y}$ if the value of the sensitive attribute S had changed from s to s'.

2.3. Ensembles and Bagging

Ensemble learning in classification tasks is a machine learning technique that combines the predictions of n hypotheses $l_1, l_2, l_3, ..., l_n$, often referred to as base learners, to improve overall performance and robustness. Typically, these base learners are combined using methods such as voting, averaging, or stacking, leveraging the diversity of the models to reduce errors and enhance generalization. Classical ensemble learning methods include *Bagging*, *Boosting*, and *Stacking*. The framework proposed in this work is based on Bagging.

Bagging [Breiman 1996], short for Bootstrap Aggregating, improves model accuracy and reduces overfitting by training multiple homogeneous base learners (same machine learning algorithms) on different subsets of the training data ($D_1, \cdots, D_n$), created through random sampling with replacement from the original dataset D, and combining their predictions, typically using averaging for regression or majority voting for classification. Furthermore, *Bagging* is efficient because base learners can be computed in parallel.

Concerning GNNs, creating diversity in GNN ensembles is more challenging compared to other types of neural networks. GNN predictions often resemble those of the label propagation algorithm [Wang and Leskovec 2020], a semi-supervised learning algorithm that spreads labels from labeled nodes to unlabeled nodes in a graph. This occurs because GNN predictions are heavily influenced by the input graph's topology and the labeled nodes. Models trained on the same graph with the same training set are likely to produce similar predictions and repeat the same errors [Nagarajan et al. 2022]. In Section 4 we propose a data augmentation procedure to mitigate this effect.

3. Related Work

3.1. Graph counterfactual fairness

NIFTY is the most effective framework that deals with graph counterfactual fairness. and generates embeddings that are fair and stable [Agarwal et al. 2021]. To achieve this goal, NIFTY uses optimization with regularization in which a triple objective function maximizes the similarity of the embeddings of the input graph G and the counterfactual and noisy graphs created. To maximize this similarity, a Siamese network is used, which is a neural network architecture composed of two or more identical networks that share the same weights and parameters [Bromley et al. 1993]. In addition, Lipschitz normalization is used in the layers of the GNN to improve the message passing mechanism. In experiments, NIFTY incorporates several GNN architectures, such as GCN [Kipf and Welling 2017] and GraphSAGE [Hamilton et al. 2017a].

To generate NIFTY's counterfactual graph, node sensitive attributes are flipped to ensure counterfactual fairness (if the value of the sensitive attribute is changed in a counterfactual scenario, the predictions given by the algorithm should remain the same).

To generate NIFTY's noisy graph, the attributes of the original nodes in G are slightly perturbed and edges are randomly dropped using DropEdge [Rong et al. 2019]. To add node attribute noise, $\mathbf{x}_v$ (attributes of node v) are perturbed using a mask vector $r \in \{0, 1\}^d$ that follows a Bernoulli distribution, i.e. $r \sim B(p_n)$, such that p_n is the probability of independently perturbing each attribute (except the sensitive attribute). Thus, the new attribute vector generated is $\tilde{\mathbf{x}}_v = \mathbf{x}_v + r \circ \delta$, where the symbol $\circ$ is the element-wise multiplication operator and $\delta \in \mathbb{R}^d$ is sampled from a normal distribution. In other words, r indicates which attributes will be perturbed, and $r \circ \delta$ has the value of the noise that will be added to the attribute vector. DropEdge uses a random binary mask from a Bernoulli distribution, i.e., $\mathbf{R}_e \sim B(1{-}p_e)$, such that $\mathbf{R}_e \in \{0, 1\}^{N \times N}$ and p_e denotes the probability with which an edge is dropped from G. Thus, the new adjacency matrix is $\tilde{\mathbf{A}} = \mathbf{A} \circ \mathbf{R}_e$.

This perturbations ensure stability.

Biased Edge Dropout [Franco et al. 2024] modifies NIFTY's DropEdge strategy that is focused on enforcing only stability. To also enforce fairness, Biased Edge Dropout generates a noisy graph with a desired homophily rate (i.e., edges that connect nodes with the same values of the sensitive attributes). According to Spinelli et al. (2021), removing homophilous edges can help improve group fairness. Biased Edge Dropout is a variation of FairDrop algorithm [Spinelli et al. 2021] and introduces a new hyperparameter, the homophilous rate ρ, that specifies the maximum ratio of allowed homophilous connections. Given the new adjacency matrix $\tilde{\mathbf{A}}$ for the noisy graph and the original adjacency matrix $\mathbf{A}$, let E_A be the set of edges not dropped by NIFTY's DropEdge and $E_{\tilde{A}}$ be the set of edges dropped by NIFTY's DropEdge. Using E_A, the percentage of homophilous edges is computed. If this percentage is greater than ρ, homophilous connections in E_A are replaced with heterophilous connections in $E_{\tilde{A}}$ until ρ is reached [Franco et al. 2024].

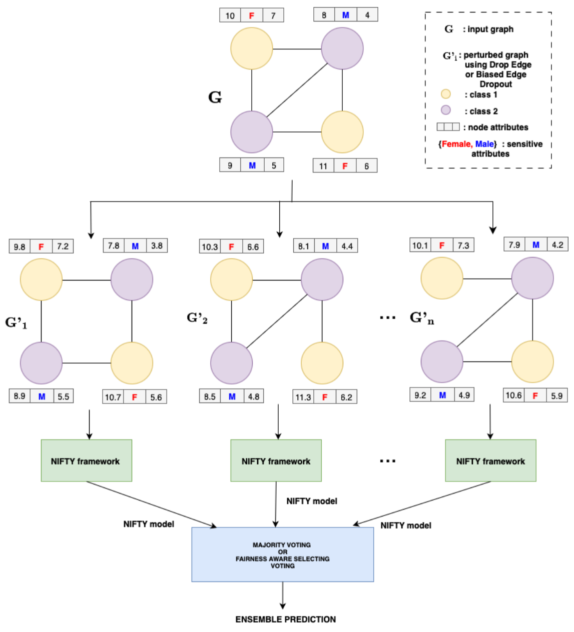
3.2. Ensemble learning in GNNs

There are works in the literature that use GNN ensembles for applications in specific domains [Kosasih et al. 2021, Chakravarty et al. 2020]. They experimentally showed that the use of GNN ensembles brings benefits, obtaining better algorithmic performance than individual GNNs. Among the GNN ensemble frameworks that have been proposed are GEENI [Nagarajan et al. 2022], GNN-Ensemble [Wei et al. 2023] and GEL [Lin et al. 2022]. GEENI uses several models of different GNN architectures that are trained using different hyperparameters and regularizers. For each of these models, a score is calculated for how much the model improves the diversity and accuracy of the ensemble. GNN-Ensemble combines hundreds of base learners using the *Bagging* method and performs three steps. First, substructures are randomly selected from the topological space of the input graph and a subset of attributes are randomly selected from the original attribute space. Second, different base learners are trained with these subgraphs and attribute subsets. Third, to perform the final classification, the decisions of the base learners are aggregated using different voting methods. In GEL, a loss function is proposed that takes into account the transfer of knowledge between the base learners. Two strategies are adopted to aggregate the predictions: (i) serialized ensemble (similar to *Boosting*) and (ii) parallel ensemble (similar to *Bagging*). None of these ensemble approaches consider algorithmic fairness, focusing solely on predictive performance.

4. *FairGNN-Bagging*

We introduce a novel framework, *FairGNN-Bagging*, designed to strike an improved balance between fairness and predictive performance in node classification tasks. This approach leverages ensemble learning, with NIFTY framework to train base learners.

Our approach is guided by two key assumptions:(1) Bagging techniques reduce variance and improve the robustness of statistical models [Breiman 1996], and (2) ensembles of models trained directly on the same graph dataset tend to present little variability [Wei et al. 2023], hence the need for a data augmentation mechanism. To address this, our framework uses three concepts: (1) data augmentation from the perturbation of non-sensitive attributes and DropEdge (or Biased Edge Dropout), that are techniques that drop edges to generate variability in the population of models in the ensemble, (2) parallelized training of n models with the NIFTY framework, and (3) aggregation of the responses of the ensemble models through majority voting or fairness-aware selective voting.



!t

Figure 1. The *FairGNN-Bagging* framework where n different perturbed graphs G_i' are created from the input graph G. For example, to obtain G_1', a homophilous edge (a connection between two nodes with the same sensitive attribute was randomly dropped) and all non-sensitive attributes were slightly perturbed. Each G_i' is fed into the NIFTY framework to train base learners, whose predictions are combined to form the ensemble using majority voting or fairness-aware selective voting.

4.1. Data augmentation.

To generate diversity in the ensemble, its members should be diverse; for this, as we commented before, we create an augmented graph-dataset to train an ensemble of GNNs. This prevents overfitting to a specific data distribution. To create the augmented graph-dataset we use the same strategy presented in NIFTY [Agarwal et al. 2021] to generate the perturbed (noisy) graph. We add node attribute noise using p_n as the probability of independently perturbing each attribute (except the sensitive attribute). We also use two alternative strategies to drop edges: (1) DropEdge using p_e' as the probability with which an edge is randomly dropped from G, and (2) Biased Edge Dropout to balance homophilous and heterophilous sensitive connections using ρ to specify the maximum ratio of allowed homophilous connections in the graph G and also using p_e'. The *FairGNN-Bagging* framework is showed in Fig. 1 where n different perturbed graphs $G_i' = (V, E', \tilde{\mathbf{X}})$ are created adding attribute noise and dropping the edges of the input graph G using DropEdge or Biased Edge Dropout.

4.2. Parallelized training and aggregation.

Each perturbed graph G_i' is fed into the NIFTY framework to train base learners, which are them combined to form the ensemble. After training multiple NIFTY models on the augmented graph dataset, we combined their predictions using (i) majority voting, where each base learner casts a vote for a class, and the most frequent label is assigned to each node or (ii) fairness-aware selective voting, where test-time predictions are made using

Table 1. Methods used in the experiments and its acronyms

Method	Acronym
GCN [Kipf and Welling 2017], baseline architecture for graph-based tasks	GCN
NIFTY [Agarwal et al. 2021] incorporated into GCN that uses DropEdge	NIFTY
FairGNN-Bagging using DropEdge, evaluated with n=10 and majority voting	NE10
FairGNN-Bagging using DropEdge, evaluated with n=20 and majority voting	NE20
FairGNN-Bagging using DropEdge, evaluated with n=10 and fairness-aware selective voting	sNE10
FairGNN-Bagging using DropEdge, evaluated with n=20 and fairness-aware selective voting	sNE20
NIFTY [Agarwal et al. 2021] incorporated into GCN that used Biased Edge Dropout	beNIFTY
FairGNN-Bagging using Biased Edge Dropout, evaluated with n=10 and majority voting	beNE10
FairGNN-Bagging using Biased Edge Dropout, evaluated with n=20 and majority voting	beNE20
FairGNN-Bagging using Biased Edge Dropout, evaluated with n=10 and fairness-aware selective voting	sbeNE10
FairGNN-Bagging using Biased Edge Dropout, evaluated with n=20 and fairness-aware selective voting	sbeNE20

only a percentage ψ of base learners that achieved the highest counterfactual fairness scores during training (see Fig. 1). For example, if $n = 20$ and $\psi = 80\%$, only the 16 best base learners will participate in the voting. Finally, we evaluated the predictive performance and fairness metrics.

We demonstrate that training multiple NIFTY models in parallel captures diverse biases, and combining their predictions enhances predictive performance, improves fairness without sacrificing predictive performance.

5. Experiments

This section presents a comparison among the methods shown in Table 1.

5.1. Datasets and Metrics

The datasets *German credit* [Dua et al. 2017], *Recidivism* [Jordan and Freiburger 2015] and *Credit defaulter* [Yeh and hui Lien 2009] were used. In these datasets, the sensitive attributes are binary and the task is binary node classification. These datasets have attribute bias and structural bias that affects information propagation in GNNs [Dong et al. 2022]. The graph of the **German credit** dataset [Dua et al. 2017] has 1,000 nodes representing customers that are connected based on the similarity of their credit accounts and has 22,242 edges. The gender of the customers is the sensitive attribute and the task is to differentiate customers with good credit risk from bad credit risk. The graph of the **Recidivism** dataset [Jordan and Freiburger 2015] has 18,876 nodes representing defendants who were released on bail in US state courts (1990-2009). The defendants are connected based on the similarity of their criminal records and demographics. The graph has 321,308 edges. The sensitive attribute is race, and the goal is to classify defendants into bail (i.e. unlikely to commit a violent crime if released) vs. no bail (i.e. likely to commit a violent crime). The **Credit defaulter** dataset graph has 30,000 nodes representing customers who are connected based on similar spending and payment patterns. The graph has 1,436,858 edges. Customer age is the sensitive attribute (customers over 25 and customers under or equal to 25 years of age). The task is to classify whether a customer will default on credit card payments.

Following the evaluation criteria used in [Agarwal et al. 2021], the AUROC and F1-score metrics are used to evaluate node classification performance. Two metrics are used to quantify group fairness: Demographic Parity ($\Delta_{DP} = |P(\hat{Y}_v = 1|s = 0) - P(\hat{Y}_v = 1|s = 1)|$) and Equal Opportunity ($\Delta_{EO} = |P(\hat{Y}_v = 1|Y_v = 1, s = 0) - P(\hat{Y}_v = 1|Y_v = 1, s = 1)|$). In addition, to measure counterfactual fairness, the CF score is used as the percentage of test nodes for which the node class prediction changes when the node's sensitive attribute is flipped. Finally, the Instability score is represented as the percentage of test nodes for which the node class prediction changed when random noise was added to the node attributes.

5.2. Experimental Setup

NIFTY, base learners of NEn and sNEn were trained using the hyperparameter values guided in [Agarwal et al. 2021], that are, $p_n = 0.1$ and $p_e = 0.001$, using Adam as the optimizer, a learning rate $\lambda = 0.001$, 1000 epochs and using a GCN as the NIFTY's encoder. In NEn, sNEn, beNEn and sbeNEn, for creating G'_i, the probability of independently perturbing each attribute of $\mathbf{x}_v$ was also set to $p_n = 0.1$, and the probability of randomly drop edges was set to $p'_e = 0.3$. We used this p'_e value to generate more diversity for the ensemble, thus eliminating more edges. The number of base learners, n, for NEn, sNEn, beNEn and sbeNEn was set to $\{10, 20\}$. beNIFTY and each base learner used in beNEn and sbeNEn use $p_n = 0.1$ and $p_e = 0.3$. We set ρ (homophilous rate) to 0.3 for enforcing more heterophilous connections in the graphs created and use a high value of $p'_e = 0.3$ to allow the swap between homophilous and heterophilous edges in Biased Edge Dropout. We use $\psi = 80\%$ to select the base learners that achieved the highest counterfactual fairness scores during training for sNEn and sbeNEn. GCN, NIFTY, beNIFTY and each base learner in the ensembles configurations were configured with single-layer GNN encoder with hidden dimensionality set to 16, and we used 50%, 25% and 25% of nodes for training, validation and test following the setup used in the original NIFTY paper [Agarwal et al. 2021].

5.3. Results

The results of the experiments comparing all the methods are shown in Table 2. For each metric, the mean and standard deviation were calculated considering 10 simulations with different random seeds. As expected, for all datasets, CF, Δ_{DP}, Δ_{EO} and Instability score of GCN are much worse than the others GNNs that lead with fairness.

In the *German Credit* dataset, ensemble-based models outperformed individual baselines across all fairness and stability metrics, while maintaining or improving predictive performance. Moreover, sbeNE20 reported the lowest instability, demographic parity (Δ_{DP}), and equal opportunity (Δ_{EO}) and also achieved perfect counterfactual fairness, demonstrating the effectiveness of fairness-aware selective voting. These models also showed consistent performance across multiple runs, indicating improved robustness. Overall, these results confirm that FairGNN-Bagging, particularly with fairness-aware selective voting and Biased Edge Dropout, improves counterfactual and group fairness without compromising predictive performance and leads to more stable predictions.

For *Recidivism*, one of the ensemble variants has the best mean result for Instability score (sNE10), Δ_{DP} (NE20) and Δ_{EO} (NE10). The *Recidivism* dataset exhibits less structural and attribute bias [Dong et al. 2022] compared to the other datasets employed in this study, limiting the effect of fairness methods. This characteristic may account for the degradation in predictive performance metrics observed in fairness-oriented methods when compared to GCN. The performance loss is particularly notable in the F1-score. While GCN achieved the highest AUROC and F1-score, it exhibited the worst fairness and instability metrics.

Table 2. Comparison of method metrics. The arrow ↑ indicates that the higher the value is better, while ↓ indicates that the lower the value is better.

Dataset	Method	AUROC (↑)	F1-score (↑)	CF (↓)	Instability (↓)	Δ_{DP} (↓)	Δ_{EO} (↓)
	GCN	67.62±3.52	80.39±1.99	3.88±2.16	8.08±4.68	5.45±5.57	4.23±3.96
	NIFTY	68.07±2.59	82.06±0.38	0.96±0.98	1.64±0.94	3.11±2.04	1.80±1.70
	NE10	70.44±1.35	82.44±0.22	0.12±0.18	0.48±0.75	1.66±1.55	1.54±2.08
German	NE20	69.92±0.52	82.63±0.28	**0.00**±0.00	0.24±0.32	0.95±0.52	1.83±1.38
	sNE10	69.85±1.89	82.38±0.28	**0.00**±0.00	**0.08**±0.16	0.90±0.90	0.74±1.02
	sNE20	69.32±0.88	**82.81**±0.27	0.04±0.12	0.16±0.27	1.12±0.74	0.49±0.4
	beNIFTY	66.57±4.65	81.76±0.61	1.82±2.17	2.36±1.71	2.64±2.10	2.36±2.03
	beNE10	68.25±3.65	81.66±1.07	1.04±0.84	2.76±2.22	4.59±3.96	4.15±3.61
	beNE20	**70.52**±1.37	82.39±0.33	0.52±0.54	1.20±1.01	2.51±2.27	2.72±3.31
	sbeNE10	69.55±2.88	82.21±0.58	0.04±0.12	0.36±0.38	0.90±1.05	1.58±3.17
	sbeNE20	68.92±2.30	82.56±0.26	**0.00**±0.00	**0.08**±0.16	**0.68**±0.77	**0.25**±0.39
	GCN	**93.65**±0.21	**83.44**±0.63	5.08±0.47	18.03±0.81	6.55±0.16	5.30±0.19
	NIFTY	85.93±0.54	64.92±4.49	0.88±0.56	8.50±0.93	3.49±0.39	1.30±0.87
	NE10	88.28±0.66	63.10±4.81	1.66±0.50	7.74±0.96	3.04±0.22	**0.57**±0.32
Recidivism	NE20	88.91±0.46	63.73±3.08	2.14±0.38	8.33±0.80	**2.93**±0.19	0.71±0.39
	sNE10	87.55±0.41	59.25±5.61	1.29±0.44	**6.95**±1.20	2.96±0.31	0.64±0.46
	sNE20	88.12±0.52	60.76±2.71	1.83±0.49	7.57±0.93	2.97±0.22	0.61±0.40
	beNIFTY	85.65±0.51	63.74±3.26	**0.82**±0.59	7.94±0.71	3.77±0.51	2.20±1.24
	beNE10	88.00±0.38	62.94±3.03	1.01±0.26	7.59±0.71	3.42±0.28	1.33±0.61
	beNE20	88.82±0.40	62.71±2.20	1.14±0.14	7.41±0.78	3.44±0.18	1.29±0.59
	sbeNE10	87.53±0.25	61.81±2.97	0.95±0.32	7.52±0.90	3.40±0.15	1.31±0.58
	sbeNE20	88.27±0.57	60.66±3.48	1.23±0.28	7.04±0.70	3.18±0.22	0.89±0.40
	GCN	**69.28**±0.33	**87.67**±0.12	5.58±1.50	6.47±2.15	5.81±0.87	3.05±0.73
	NIFTY	68.56±0.14	87.59±0.05	0.75±0.63	3.46±0.47	1.96±1.12	1.18±0.64
	NE10	69.05±0.06	87.60±0.04	0.12±0.08	2.65±0.57	1.06±0.62	0.65±0.30
Credit	NE20	69.14±0.02	87.58±0.02	0.06±0.07	2.33±0.36	0.89±0.13	0.38±0.11
	sNE10	69.06±0.08	87.59±0.03	0.08±0.05	2.31±0.50	0.97±0.32	0.53±0.13
	sNE20	69.16±0.02	87.57±0.02	**0.03**±0.02	**2.12**±0.39	**0.82**±0.11	**0.36**±0.08
	beNIFTY	68.61±0.15	87.61±0.06	0.76±0.67	3.64±0.55	1.79±0.93	1.17±0.64
	beNE10	69.11±0.04	87.61±0.02	0.11±0.07	2.44±0.25	0.88±0.28	0.47±0.27
	beNE20	69.19±0.02	87.59±0.02	0.06±0.02	2.13±0.20	0.93±0.11	0.46±0.05
	sbeNE10	69.09±0.04	87.59±0.02	0.09±0.06	2.42±0.29	0.98±0.62	0.55±0.32
	sbeNE20	69.17±0.02	87.57±0.01	0.05±0.03	2.17±0.20	0.84±0.06	0.38±0.09

In the *Credit Defaulter* dataset, FairGNN-Bagging models achieved the best overall trade-off between predictive performance and fairness. Ensemble variants reached low counterfactual fairness, low instability, and small values in Δ_{DP} and Δ_{EO}. Importantly, these gains were achieved without degrading AUROC or F1-score, which remained comparable to GCN. The ensemble models also produced consistent outcomes across different runs. Additionally, the results show that the use of DropEdge alone was sufficient to promote diversity and fairness improvements, with Biased Edge Dropout offering only marginal gains in this particular dataset. Fairness-aware selective voting and DropEdge (sNE20) achieves the best fairness and instability metrics without compromising predictive performance.

6. Conclusion

In this work, we proposed FairGNN-Bagging, an ensemble-based framework designed to improve the trade-off between predictive performance and algorithmic fairness in node classification tasks on graphs. By leveraging the NIFTY framework with structured graph perturbations and Bagging, our method generates diverse and fairness-aware base models, which are then aggregated using majority or selective voting. We also introduced a selective voting strategy based on counterfactual fairness scores during training, allowing only the most fair base learners to participate in the ensemble decision.

Experimental results on three real-world datasets demonstrated that FairGNN-

Bagging consistently reduces counterfactual fairness, group fairness and instability, while maintaining or even enhancing predictive performance compared to traditional GNN and NIFTY baselines. The framework proves particularly effective in scenarios where fairness and reliability are critical. Furthermore, our results show that DropEdge or Biased Edge Dropout offer improvements depending on the datasets. These findings highlight the potential of ensemble learning to produce fairer and more stable predictions.

7. Acknowledgments

We thank the support given by the Brazilian National Council for Scientific and Technological Development (CNPq grant number 312360/2023-1 and 162886/2023-3) and Itaú Unibanco S.A. through the PBI program of the Centro de Ciência de Dados (C2D) of Escola Politécnica at Universidade de São Paulo.

References

Agarwal, C., Lakkaraju, H., and Zitnik, M. (2021). Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pages 2114–2124. PMLR.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24:123–140.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, page 737–744.

Chakravarty, A., Sarkar, T., Ghosh, N., Sethuraman, R., and Sheet, D. (2020). Learning decision ensemble using a graph neural network for comorbidity aware chest radiograph screening. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 1234–1237.

Cynthia, Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 214–226, New York, NY, USA. ACM.

Dai, E., Zhao, T., Zhu, H., Xu, J., Guo, Z., Liu, H., Tang, J., and Wang, S. (2024). A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *Machine Intelligence Research*, pages 1–51.

Dong, Y., Liu, N., Jalaian, B., and Li, J. (2022). Edits: Modeling and mitigating data bias for graph neural networks. In *Proceedings of the ACM web conference 2022*, pages 1259–1269.

Dong, Y., Ma, J., Wang, S., Chen, C., and Li, J. (2023). Fairness in graph mining: A survey. *IEEE TKDE*, 35(10):10583–10602.

Dua, D., Graff, C., et al. (2017). UCI machine learning repository, 2017. 7(1):62.

Franco, D., D'Amato, V. S., Pasa, L., Navarin, N., and Oneto, L. (2024). Fair graph representation learning: Empowering nifty via biased edge dropout and fair attribute preprocessing. *Neurocomput.*, 563(C).

Hamilton, W., Ying, Z., and Leskovec, J. (2017a). Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017b). Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*, 40(3):52–74.

Jordan, K. L. and Freiburger, T. L. (2015). The effect of race/ethnicity on sentencing: Examining sentence type, jail length, and prison length. *Journal of Ethnicity in Criminal Justice*, 13(3):179–196.

Kang, J., He, J., Maciejewski, R., and Tong, H. (2020). Inform: Individual fairness on graph mining. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 379–389, NY, USA. ACM.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, Conference Track Proceedings*. OpenReview.net.

Kosasih, E. E., Cabezas, J., Sumba, X., Bielak, P., Tagowski, K., Idanwekhai, K., Tjandra, B. A., and Jamasb, A. R. (2021). On graph neural network ensembles for large-scale molecular property prediction. *arXiv preprint arXiv:2106.15529*.

Kusner, M. J., Loftus, J., Russell, C., and Silva, R. (2017). Counterfactual fairness. *Advances in neural information processing systems*, 30.

Lin, Q., Yu, S., Sun, K., Zhao, W., Alfarraj, O., Tolba, A., and Xia, F. (2022). Robust graph neural networks via ensemble learning. *Mathematics*, 10(8).

Nagarajan, A., Stevens, J. R., and Raghunathan, A. (2022). Efficient ensembles of graph neural networks. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pages 187–192.

Pearl, J. (2009). Causal inference in statistics: An overview.

Rong, Y., Huang, W., Xu, T., and Huang, J. (2019). Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*.

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

Wang, H. and Leskovec, J. (2020). Unifying graph convolutional neural networks and label propagation. *arXiv preprint arXiv:2002.06755*.

Wei, W., Qiao, M., and Jadav, D. (2023). GNN-Ensemble: Towards Random Decision Graph Neural Networks . In *2023 IEEE International Conference on Big Data (BigData)*, pages 956–965, Los Alamitos, CA, USA. IEEE Computer Society.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Yeh, I.-C. and hui Lien, C. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Syst. Appl.*, 36:2473–2480.

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