FACS-GCN: Fairness-Aware Cost-Sensitive Boosting of Graph Convolutional Networks

Francisco Santos, Junke Ye, Farzan Masrour, Pang-Ning Tan, and Abdol-Hossein Esfahanian
Dep. of Computer Science and Engineering
Michigan State University
East Lansing, United States
{santosf3,yejunke,masrours,ptan,esfahanian}@msu.edu

2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9892919

***Abstract*—Graph neural networks (GNNs) have emerged as a powerful tool for modeling graph data due to their ability to learn a concise representation of the data by integrating the node attributes and link information in a principled fashion. However, despite their promise, there are several practical challenges that must be overcome to effectively use them for node classification problems. In particular, current approaches are vulnerable to different kinds of biases inherent in the graph data. First, if the class distribution is imbalanced, then the GNNs' loss function is biased towards classifying the majority class correctly rather than the minority class, which hurts the performance of the latter class. Second, due to homophily effect, the learned representation and subsequent downstream tasks may favor certain demographic groups over others when applied to social network data. To mitigate such biases, we propose a novel framework called Fairness-Aware Cost Sensitive Graph Convolutional Network (FACS-GCN) for classifying nodes in networks with skewed class distributions. Our approach combines a cost-sensitive exponential loss with an adversarial learning component to alleviate the ill-effects of both biases. The framework employs a stagewise additive modeling approach to ensure there is no significant loss in accuracy when imparting fairness into the GNN. Experimental results on 6 benchmark graph data demonstrate the effectiveness of FACS-GCN against comparable baseline methods in terms of promoting fairness while maintaining a high model accuracy on the majority of the datasets.**

***Index Terms*—Fairness; Graph neural networks**

I. Introduction

Node classification is the task of categorizing the nodes in a network into their respective labels. It can be applied to many applications, from spam [1] and financial fraud [2] detection to customer churn [3] and recidivism [4] prediction. Classical techniques for node classification include label propagation [5], matrix alignment [6], and Iterative Classification Algorithms (ICA) [7] [8]. More recently, neural network methods inspired by deep learning such as Deepwalk [9], GraphSage [10], and Graph Convolutional Networks (GCN) [11] have become increasingly popular to address the node classification problem. GCN achieves state-of-the-art performance by employing a message-passing paradigm [11] to aggregate the feature representation of each node along with its neighbors to create a new and more concise feature embedding.

Despite the growing success of graph neural networks (GNNs), there are several practical limitations that could hinder their use in real-world applications. For example, the social implications of their prediction results are not fully accounted by conventional GNNs. In particular, fairness has become a growing topic of concern, as current GNNs may discriminate against certain demographic groups in the population [12]. For example, consider the application of GNNs to professional networking sites. As certain professions could be dominated by individuals from a particular demographic group, this will likely be reflected in the link structure of the network due to the so-called homophily effect [13]. As a result, methods such as GNN may further reinforce this demographic-based segregation, which leads to inequalities observed in other protected groups. Protected groups here refer to the collection of individuals who have been traditionally marginalized and may potentially be re-victimized by the unfair algorithmic decisions. Such groups are typically defined based on certain sensitive attributes such as race, gender, and age. To date, several metrics have been developed, such as Statistical Parity [14] and Equality of Opportunity [15] to assess fairness in the algorithmic decisions. As conventional GNNs are oblivious to the presence of such attributes, the key challenge here is to impart fairness into current GNN implementations.

There have been several recent efforts to overcome this limitation [16]–[18]. For example, FairGNN [16] employs an adversarial learning technique to remove the potential biases present in the learned representation of GNNs. However, current approaches are limited in two ways. First, they are not designed to handle the imbalanced class distribution of the network data, in which a subset of the target class of interest (e.g., spammer or customer churn) occurs less frequently than other classes. This presents a major challenge to current GNNs as their loss functions are not designed to handle bias due to such skewed class distribution. Second, the improved fairness often comes at the expense of sacrificing the classification accuracy. The problem is further compounded by the fact that current GNNs are limited to shallow network architectures due to the over-smoothing effect [19], which causes degradation of their model performance beyond 2 or 3 layers. To address the over-smoothing problem, Sun et al. [20] proposed the AdaGCN framework, which combines GCN with AdaBoost [21] to enable better aggregation of the node representation, allowing the network to have deeper architectures to achieve better performance. However, none of the existing fairness-

978-1-7281-8671-9/22/$31.00 ©2022 IEEE

aware GNN methods are designed to take advantage of such deeper networks to compensate for the loss in model accuracy when incorporating fairness into their GNN framework. As boosting and AdaGCN employs an exponential loss, a key question here is: ***How to incorporate fairness and handle class imbalanced in GNNs with exponential loss?***

To overcome these challenges, we propose a novel GNN framework called FACS-GCN (Fairness Aware Cost Sensitive Graph Convolutional Network), which employs a cost sensitive exponential loss function to handle the imbalanced classes as well as an adversarial learning technique to mitigate biases against the protected groups. FACS-GCN is composed of three main components: (1) a generator that uses a cost-sensitive version of AdaGCN [20] to learn the feature embedding of the nodes, (2) a discriminator that plays the role of an adversary attempting to uncover the protected attribute value of a node given its feature embedding, and (3) a forward stagewise additive modeling component for node classification. Together, the adversarial learning implemented using the cost-sensitive generator and discriminator will create to an unbiased feature representation to be utilized by the node classifier to generate fair and accurate predictions.

In summary, our contributions in this paper are as follows:

- We present a novel GNN framework based on cost-sensitive exponential loss to help improve the joint representation learning and node classification for networks with an imbalanced class distribution.
- We develop a fairness-aware framework that combines adversarial learning with a forward stagewise additive modeling to handle the trade-off between maximizing fairness and accuracy.
- We perform extensive experiments on numerous real world datasets to demonstrate efficacy of our framework.

II. Related Work

There has been extensive research on the development of node classification techniques for network data [6]–[8]. This includes random-walk based approaches such as label propagation [5], which infer the label of a node by aggregating the labels of its neighbors, and matrix factorization [6] methods, which seeks to find a decomposition of the adjacency and node feature matrices in a way that can be used to infer the node labels. A survey of prior research on node classification can be found in [22], [23]. More recently, graph neural network (GNN) methods such as DeepWalk [9], graph convolutional networks (GCNs) [11], and their variants [10], [20], have grown in popularity due to their ability to learn a latent representation of the nodes that can be provided to a fully-connected network to perform node classification.

There have also been growing interests to extend the GNN framework to address the imbalanced class distribution. For example, Distance-wise Prototypical GNN (DPGNN) [24] uses prototype-driven training to handle the trade-off the training loss for the majority and minority classes. In contrast, Dual-Regularized GCN (DR-GCN) [25] uses a class-conditional adversarial training and a latent distribution regularization to reconcile the loss function between the two classe while GraphSMOTE [26] creates synthetic nodes for the minority class to balance the skewed distribution. However, none of these approaches have been used for fairness-aware node classification and remain susceptible to the over-smoothing problem [19]. Furthermore, methods such as GraphSMOTE would modify the network, which might have legal implications in terms of fairness.

Algorithmic fairness has emerged as a topic of growing concern among machine learning researchers. Numerous preprocessing [27]–[30], in-processing [31]–[34], and postprocessing [15], [35], [36] methods have been developed to address this challenge. The issue of fairness will likely be more pronounced in network data due to the homophily effect, which leads to biased predictions [16]. There has been some recent attempts to incorporate fairness into GNNs for representation learning and link prediction problems. For example, Masrour et al. [37] presented an approach called FLIP for fair link prediction to alleviate the filter bubble problem. FairDrop [18] generates a fair random copy of the adjacency matrix by reducing the number of edges between nodes sharing the same sensitive attribute. FairWalk [38] modifies the random walk algorithm to generate a fair node representation. For node classification, Dai et al. [16] proposed FairGNN to debias the node classification results through adversarial learning while Agarwal et al. [17] proposed NIFTY (uNIfying Fairness and stabiliTY) which enforces fairness and stability by introducing an objective function that maximizes the agreement between the original graph, its counterfactual and the noisy views. However, none of the methods are designed to also handle the imbalanced class distribution as well as over-smoothing effect that hinders the applicability of GNN to real-world networks.

III. Preliminaries

Consider an attributed network $N = (V, E, X, Y)$, where V is the set of nodes, $E \subseteq V \times V$ is the set of edges, $X \in \mathbb{R}^{|V| \times d}$ is the feature matrix containing the attributes of all the nodes in V, and Y is the class label of the nodes. The feature matrix X can be divided into submatrices X^p and X^u, where X^p refers to the protected attributes and X^u corresponds to the unprotected attributes. For brevity, we assume a binary classification problem, though in principle, the proposed methodology can be extended to a multi-class setting. The class label of each node v is represented by a 2-dimensional vector, $y_v = [-1, 1]$ for the positive class or $y_v = [1, -1]$ for the negative class. Furthermore, we assume the 2-dimensional vector y_v can be mapped to its corresponding label as follows:

$$c(y_v) = \begin{cases} 1 & \text{if } y_v = [1, -1] \\ 2 & \text{if } y_v = [-1, 1] \end{cases} \tag{1}$$

Let A denote the adjacency matrix representation of E and $\tilde{A} = A + I$, where I is the identity matrix. The normalized adjacency matrix is given by $\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$, where $\tilde{D}$ is the degree matrix of $\tilde{A}$.

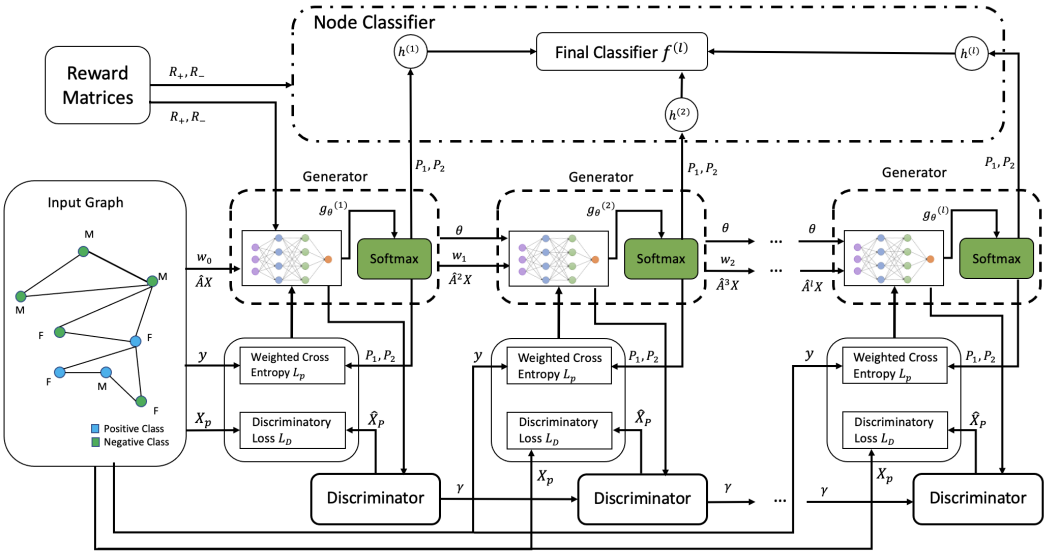

Fig. 1. A schematic illustration of the proposed FACS-GCN architecture.

A. AdaGCN

AdaGCN [20] is a forward stagewise additive modeling framework designed to boost the performance of GCN by overcoming the over-smoothing problem when the number of graph convolutional layers increases. Specifically, the framework combines GNN with AdaBoost [21] by considering each GNN as a weak learner that can be stacked together to form a 'stronger' classifier, i.e., $f^{(l)}(X) = f^{(l-1)}(X) + h^{(l)}(X) = \sum_l h^{(l)}(X)$, where $f^{(l)}(X)$ is the resulting classifier after combining l weak learners. Inspired by AdaBoost [5], [21], AdaGCN minimizes the following exponential loss function:

$$\begin{aligned} &\text{argmin}_{h(x)} E_{Y|x}\left(\exp\left[-\frac{1}{2} Y^T (f^{l-1}(x) + h^{(l)}(x)) \right] \middle| x \right) \\ &\text{s.t.} \quad h_1^{(l)}(x) + h_2^{(l)}(x) = 0 \end{aligned} \tag{2}$$

where $h^{(l)}(x) = [h_1^{(l)}(x); h_2^{(l)}(x)]$ corresponds to the weak learner trained in layer l.

Minimizing the exponential loss given in (2) with respect to $h_k^{(l)}$ will lead to the following solution [39]:

$$h_1^{(l)}(x) = \frac{1}{2} \log \left[\frac{p_1^{(l)}(x)}{p_2^{(l)}(x)} \right], h_2^{(l)}(x) = \frac{1}{2} \log \left[\frac{p_2^{(l)}(x)}{p_1^{(l)}(x)} \right], \tag{3}$$

where $p_k^{(l)}(x_v) = \text{Softmax}\left[g_\theta^{(l)}\left(c(y_v) = k \middle| x_v \right) \right]$ is the probabilistic output of some graph neural network g used to train the weak learner $h_k^{(l)}$. AdaGCN employs the following 2-layer fully-connected network to generate the output $g_\theta^{(l)}(X) \in \mathbb{R}^{|V| \times 2}$ for its weak learner:

$$g_\theta^{(l)}(X) = ReLU(\hat{A}^l X \theta_1)\theta_2 \tag{4}$$

where l is the index of a layer, $\hat{A}^l = \prod_{i=1}^{l} \hat{A}$ is the l-hop normalized adjacency matrix, $\theta_1 \in \mathbb{R}^{d \times H}$ is the input-to-hidden weight matrix, and $\theta_2 \in \mathbb{R}^{H \times 2}$ is the hidden-to-output weight matrix (assuming a binary class problem). For brevity, we denote $\theta = (\theta_1, \theta_2)$ as the neural network parameters. $g_\theta^{(l)}(X)$ enables the weak learner to extract a representation of the nodes based on their current embedding as well as those from their l-th hop neighbors.

The graph neural network $g_\theta^{(l)}(X)$ is trained to minimize the following weighted cross-entropy loss function:

$$L^P = \sum_{v \in V} w_v^{(l)} [y_v \log(p_1^{(l)}(x_v) + (1 - y_v) \log(1 - p_1^{(l)}(x_v))] \tag{5}$$

where y_v is the true class of node v, $p_1^l(x_v)$ is the conditional probability of class 1 for node v, and $w_v^{(l)}$ is the corresponding node weight. Similar to AdaBoost, the node weights are initialized to have uniform weights. As each weak learner makes its predictions, the weights are updated iteratively in a stage-wise fashion allowing the next weak learner to focus more on previously misclassified nodes. The formula to update the weights is as follows:

$$w_v \leftarrow w_v \cdot \exp\left[-\frac{1}{2} y_v^\top \log p^{(l)}(x_v) \right], \; v = 1, ..., V \tag{6}$$

The weights are then normalized to sum up to 1. After successively training all L weak learners, the final prediction for every node x_v is computed as follows:

$$\hat{y}_v = f^{(l)}(x_v) = \sum_{l=0}^{L} h_k^{(l)}(x_v) \tag{7}$$

B. Generative Adversarial Networks

Generative Adversarial Networks (GAN) is a deep-learning-based approach for generative modeling. It consists of two parts: a generator G, which will generate synthetic samples from an input noise z and a discriminator D, whose goal is to distinguish between the synthetic and real samples. GAN

has been widely-used for adversarial learning and is trained to solve the following objective function:

$$\min_{\theta}\max_{\phi} V(G_\theta, D_\phi) = E_{x\sim p_X}[\log(D_\phi(x))] + E_{z\sim p(z)}[\log(1 - D_\phi(G_\theta(z))] \quad (8)$$

IV. Proposed FACS-GCN Framework

Our proposed framework extends the AdaGCN framework to mitigate the biases due to both imbalanced class distribution and discrimination against individuals belonging to the protected groups. FACS-GCN uses a cost-sensitive version of AdaGCN to handle the skewness of class distribution with an adversarial learning component to debias the learned embedding from discriminating against certain demographic groups. The framework consists of the following two major components:

- A cost-sensitive classifier to predict the class of the nodes.
- An adversarial learning component to generate fair representation of the nodes.

We will describe each component in more details below.

A. Node Classification for Imbalanced Class Distribution

To handle the bias due to skewness of the class distribution, we design the following cost-sensitive exponential loss function for FACS-GCN:

$$\min_{h(x)} E_{Y|x}\left[e^{-\frac{1}{2}Y^T R_+ f^{(l)}(x)} I(c(Y)=1) + e^{-\frac{1}{2}Y^T R_- f^{(l)}(x)} I(c(Y)=2)\right] \quad (9)$$

where $Y \in \mathbb{R}^2$ is the true class of node x, $f^{(l)}(x) \in \mathbb{R}^2$ is the predicted output given by Equation (7) while R_+ and R_- are the reward matrices, which are defined as follows:

$$R_+ = \begin{bmatrix} R_{11} & R_{12} \\ R_{12} & R_{11} \end{bmatrix}, \quad R_- = \begin{bmatrix} R_{22} & R_{21} \\ R_{21} & R_{22} \end{bmatrix}$$

where $R_{11}, R_{22} > R_{12}, R_{21} \geq 0$. Intuitively, the main diagonal of each matrix represents the reward for correct classification while the antidiagonal corresponds to the "reward" for misclassification. If $R_{11} > R_{22}$, then there will be a higher reward for classifying the positive class (i.e., $c(Y) = 1$) correctly than the negative class (i.e., $c(Y) = 2$). We use a reward matrix instead of a cost matrix due to the form of the exponential loss used by AdaGCN. Given the 2-dimensional vector form of Y used by AdaGCN, as shown in Equation (1), minimizing the exponential loss would require a matrix R whose main diagonal must have higher values for correct classification than its antidiagonal values for incorrect classification, which is why R is interpreted as a reward matrix. The reward matrices will allow the user to determine how much 'attention' should be given to the correct prediction of the positive class in order to boost its performance (assuming the positive class is the minority class of interest). If both reward matrices are identity matrices, then the loss function reduces to Equation (2), which is the exponential loss used in AdaGCN.

Theorem 1: Given the forward stagewise additive model $f^{(l)}(x) = f^{(l-1)}(x) + h^{(l)}(x)$, the weak learner $h^{(l)}(x)$ that minimizes the exponential loss given in (9) is:

$$h_1^{(l)}(x) = \frac{1}{Z}\log\frac{(R_{11}-R_{12})p_1^{(l)}(x)}{(R_{22}-R_{21})p_2^{(l)}(x)}$$
$$h_2^{(l)}(x) = \frac{1}{Z}\log\frac{(R_{22}-R_{21})p_2^{(l)}(x)}{(R_{11}-R_{12})p_1^{(l)}(x)} \quad (10)$$

where $h_1^{(l)}(x) + h_2^{(l)}(x) = 0$ and $Z = R_{11} + R_{22} - R_{12} - R_{21}$.

Proof: Let $Y = [y_1, y_2]$ and $h^{(l)}(x) = [h_1^{(l)}(x), h_2^{(l)}(x)]$. The exponential loss in Equation (9) can be expressed as follows:

$$e^A I(c(y)=1)p(c(y)=1|x) + e^B I(c(y)=2)p(c(y)=2|x),$$

where

$$A = -\frac{1}{2}R_{11}h_1^{(l)}(x) - \frac{1}{2}R_{12}h_2^{(l)}(x) + \frac{1}{2}R_{12}h_1^{(l)}(x) + \frac{1}{2}R_{11}h_2^{(l)}(x)$$
$$B = \frac{1}{2}R_{22}h_1^{(l)}(x) + \frac{1}{2}R_{21}h_2^{(l)}(x) - \frac{1}{2}R_{21}h_1^{(l)}(x) - \frac{1}{2}R_{22}h_2^{(l)}(x)$$

By using the Lagrange multiplier method, the Lagrangian of the optimization problem is given by:

$$\begin{aligned}\mathcal{L} &= e^A I(c(y)=1)p(c(y)=1|x) + e^B I(c(y)=2)p(c(y)=2|x) \\ &\quad - \lambda(h_1^{(l)}(x) + h_2^{(l)}(x)) \\ &= e^A p(c(y)=1|x) + e^B p(c(y)=2|x) - \lambda(h_1^{(l)}(x) + h_2^{(l)}(x))\end{aligned}$$

Let $p(c(y)=1|x) = P_1$ and $p(c(y)=2|x) = P_2$. Taking the partial derivative of $\mathcal{L}$ with respect to $h_1^{(l)}(x)$ and setting it to zero yields the following:

$$\begin{aligned}&(\frac{R_{12}-R_{11}}{2})e^A P_1 + (\frac{R_{22}-R_{21}}{2})e^B P_2 \\ =\ &(\frac{R_{11}-R_{12}}{2})e^A P_1 + (\frac{R_{21}-R_{22}}{2})e^B P_2\end{aligned}$$

After some manipulation, the preceding equation simplifies to the following form: $\frac{e^A}{e^B} = \frac{(R_{21}-R_{22})P_2}{(R_{12}-R_{11})P_1}$, or equivalently, $e^{A-B} = \frac{(R_{21}-R_{22})P_2}{(R_{12}-R_{11})P_1}$. By replacing A and B into the expression and noting that $h_1^{(l)}(x) + h_2^{(l)}(x) = 0$, the expression reduces to the following form:

$$\begin{aligned}e^{(R_{12}+R_{21}-R_{11}-R_{22})h_1^{(l)}(x)} &= \frac{(R_{21}-R_{22})P_2}{(R_{12}-R_{11})P_1} \\ (R_{12}+R_{21}-R_{11}-R_{22})h_1^{(l)}(x) &= \log\frac{(R_{21}-R_{22})P_2}{(R_{12}-R_{11})P_1}\end{aligned}$$

Thus, we have:

$$h_1^{(l)}(x) = \frac{1}{(R_{11}+R_{22}-R_{12}-R_{21})}\log\frac{(R_{11}-R_{12})P_1}{(R_{22}-R_{21})P_2}$$
$$h_2^{(l)}(x) = \frac{1}{(R_{11}+R_{22}-R_{12}-R_{21})}\log\frac{(R_{22}-R_{21})P_2}{(R_{11}-R_{12})P_1}$$

The proof follows by replacing Z with $R_{11}+R_{22}-R_{12}-R_{21}$.

FACS-GCN predicts the node v to be from the positive class if $h_1(x_v) > h_2(x_v)$. Otherwise, it is predicted to be from the negative class. The weak learner derived from the

cost-sensitive exponential loss function allows the class with higher reward to increase its chance of being predicted, thus addressing the class imbalanced problem.

Once $h^{(l)}$ is calculated, the node weights are updated before training the next weak learner. The weights are updated according to the formula shown below:

$$w_v \leftarrow w_v \cdot \left[e^{-\frac{1}{Z} Y_v^\top R_1 \log(\beta \cdot P^l(x_v))} I(c(Y)=1) + e^{-\frac{1}{Z} Y_v^\top R_2 \log(\beta \cdot P^l(x_v))} I(c(Y)=2) \right] \quad (11)$$

where β is a 2×2 matrix defined as follows:

$$\begin{bmatrix} R_{11} - R_{12} & 0 \\ 0 & R_{22} - R_{21} \end{bmatrix}$$

The proof for the weight update formula is omitted due to lack of space. As shown in Equation (11), the node weights are affected by the reward matrices, allowing the misclassified nodes of the higher reward to have a larger weight than other types of misclassified nodes. This strategy enables the next weak learner to concentrate more on the inaccurately classified higher reward nodes by using the weighted cross entropy formula in (5). This helps to alleviate the bias in the loss function due to skewness in the class distribution. Once all the weak learners are computed, the final prediction is obtained by adding the predictions from all the weak learners together, as shown in Equation (7).

B. Adversarial Learning for Fairness

FACS-GCN employs adversarial learning to ensure fairness towards the protected attribute groups. The adversarial learning component consists of a generator and a discriminator. The goal of the discriminator is predict the node's protected attribute. The discriminator is optimized by the following loss:

$$L^D = -\frac{1}{|V|} \sum_{v \in V} [X_v^P log(\hat{P}_v) + (1 - X_v^P) log(1 - \hat{P}_v)] \quad (12)$$

Intuitively, if the discriminator is unable to predict the protected attribute accurately, this implies that the bias towards the protected attribute would have been eliminated from the learned embedding. Therefore, the classification of the node will be fair. In our case, we choose a two-layer GCN as our discriminator to optimize the loss function.

Our generator's goal is to create an embedding that can fool the discriminator and produce a high node classification accuracy. To accomplish this goal, $g_\theta^{(l)}(X)$ optimizes a new loss function L^T, which is defined as the following:

$$L^T = (1 - \alpha) L^P - \alpha L^D \quad (13)$$

L^P is used to maximize the weighted conditional probability used in the node classifier and L^D is used to minimize performance of the discriminator at a point where it's random guessing the protected attribute. α acts as a hyper-parameter that controls the trade-off between maximizing fairness and accuracy. Once L^T is optimized, $g_\theta^{(l)}(X)$ will produce an embedding that is fairer than before. This embedding will then be passed to the discriminator and will be optimized simultaneously for each epoch, as demonstrated in Algorithm 1. FACS-GCN is trained end-to-end using Adam as our optimizer.

Algorithm 1: FACS-GCN Algorithm

Input: Feature matrix X, normalized adjacency matrix $\hat{A}$, number of layers L, reward matrices R_1 and R_2.
Output: Weak learners, $\{h_\theta^{(1)}, h_\theta^{(2)}, \cdots, h_\theta^{(L)}\}$
Initialize the nodes to have uniform weights, $\forall v : w_v = \frac{1}{|V|}$
Initialize $f^{(-1)}$ to vector of zeros
for $l = 0$ to L **do**
  for $i = 0$ to max_epoch **do**
    Compute $g_\theta^{(l)}(X)$ Compute the total loss L^T using Equation (13) Update network parameters θ by backpropagation
  end for
  Compute $h^{(l)}$ using Equation (3)
  Set $f^{(l)}(x) = f^{(l-1)}(x) + h^{(l)}(x)$
  Update the nodes weight w_v using Equation (11)
  Renormalize the weights: $\forall v : w_v \leftarrow \frac{w_v}{\sum_{v \in V} w_v}$
end for

TABLE I
SUMMARY DESCRIPTION OF DATASETS.

Dataset	$\|V\|$	$\|E\|$	$\|X\|$	% Protected	Class skew
Tagged	71127	71265	58	0.519:0.481	0.383:0.617
Recidivism	18877	403978	18	0.942:0.058	0.376:0.624
Facebook	1045	53498	55	0.342:0.658	0.318:0.682
Credit	30000	198989	12	0.532:0.468	0.221:0.779
Pokec-n	66569	729129	265	0.513:0.487	0.064:0.936
German	1000	24970	28	0.690:0.310	0.300:0.700

V. EXPERIMENTAL EVALUATION

This section describes the experiments performed to demonstrate the effectiveness of FACS-GCN. The code is available at https://github.com/frsantosp/FACS-GCN.

A. Data Description

We have performed our experiments on the following 6 real-world network datasets. Table I summarizes the characteristics of each dataset.

- **Tagged** [40]: This is a benchmark social spammer dataset obtained from LINQS[1] website. The classification task is to predict whether an individual is a spammer or non-spammer using gender as protected attribute.
- **Facebook** [41]: This benchmark network dataset of Facebook social circles was obtained from Stanford Network Analysis Project (SNAP). The target class to be predicted

[1]https://linqs-data.soe.ucsc.edu/public/social_spammer/

corresponds to education level while the protected attribute is gender.

- **Recidivism** [42]: The dataset contains information about defendants who were granted bail in U.S. states court between 1990 and 2009. The target class corresponds to bail or no bail, with race as protected attribute.
- **Credit Defaulter** [43]: This dataset contains individuals whose links are established based on similarity of their spending and payment patterns. The task here is to predict whether an individual will default on their credit card payments using age as the protected attribute.
- **German Credit** [44]: This network data corresponds to clients of a bank, whose links are generated based on similarity of their credit accounts. The task here is to predict the credit risk of the clients using gender as protected attribute.
- **Pokec-n** [41]: The data is from a social networking website in Slovakia. The task here is to predict whether an individual likes sports or not using gender as the protected attribute.

Note that the Recidivism, Credit Defaulter, and German Credit data were used in [17] to evaluate the performance of the NIFTY algorithm while Pokec-n was used in [16] to evaluate the performance of their fairness-aware method.

B. Experimental Evaluation

We compared FACS-GCN against the following baselines:

- **GCN** [11]: A graph neural network that uses a 'message-passing' technique for a certain number of layers. In this experiment, we used a GCN with two hidden layers.
- **AdaGCN** [20]: A boosting-like method for GCN, where each weak learner l is trained to extract information for a node using its l-th hop neighbors.
- **GraphSage** [10]: This approach uses an aggregator function to create a new embedding of a node by concatenating the representation of its neighbors. For our experiments we use a 2-layer GraphSage implementation
- **FairGNN** [16]: This approach uses an adversarial debiasing approach to generate a fair representation of the nodes in a network when some of the sensitive attribute values are unknown.
- **NIFTY** [17]: This approach aims to enforce network stability by randomly perturbing the node attributes and edges as well as counterfactual fairness by perturbing the sensitive attributes. Their objective is to maximize the agreement between the original and perturbed graphs.

Given the skewed class distribution, we compute the overall area under ROC (AUC) score to evaluate the predictive accuracy of each method. We also use the following difference in equality of opportunity [15] as our fairness criterion:

$$\Delta_{EO} = |P(\hat{y} = 1|Y = 1, X^p = 0) - P(\hat{y} = 1|Y = 1, X^p = 1)|$$

The smaller are their differences, the more fair are the classification results.To verify that the cost-sensitive approach used by FACS-GCN helps to promote more accurate classification for the positive examples, we also assess the recall performance of the classifier:

$$Recall = \frac{\text{\# True Positives}}{\text{\# True Positives + \# False Negatives}} \tag{14}$$

All the baselines are trained using the hyperparameters suggested by their authors. For AdaGCN and FACS-GCN, we have implemented a two-layer GCN with 32 hidden dimensions as its weak learner. We set the number of weak learners to be 5. A two-layer GCN with 32 hidden dimensions was also implemented for all other baseline methods. Each method was trained for 1000 epochs. The experiments for FACS-GCN, FairGCN, and Nifty were repeated 5 times with different seeds.

C. Experimental Results

Performance Comparison. There is often a trade-off between maximizing model accuracy and achieving fairness. A model that focuses on achieving fairness may often sacrifice its AUC. Thus, the comparison of model performance should consider both its AUC and fairness metric in concert, as illustrated in Figure 2, which plots the fairness metric, Δ_{EO}, on the x-axis and AUC on the y-axis. The ideal situation is to have the Δ_{EO} to be 0 and AUC to be 1. Thus, the closer the result is to the upper left corner, the better is the model.

For the 3 datasets shown in first row in fig. 2, FACS-GCN performed better than all other baselines as it is closer to the upper left corner than other method. Our method promotes a better AUC than the other fairness baselines and achieves it with a comparable or better Δ_{EO} on Pokec, Recidivism and Tagged datasets than other methods. This demonstrates the effectiveness of FACS-GCN to achieve fairness without sacrificing its AUC, especially when compared against FairGNN and NIFTY, two competing fairness-aware GNN methods. In the bottom row, for the German and Credit datasets, our model achieves Δ_{EO} to be 0 tying FairGCN and beating NIFTY, though it suffers in terms of AUC. For facebook, our model beats all the baselines in terms of AUC but has lower Δ_{EO}, showing again the trade-off between AUC and fairness.

For their 1-to-1 comparison, Table II shows the win-loss comparison between the methods. The results suggest that FACS-GCN is superior in terms of AUC than the other fairness methods, beating them on majority of the datasets. with comparable performance to GCN. In terms of Δ_{EO}, FACS-GCN outperforms the baselines in at least 4 datasets, which shows its effectiveness in terms of achieving fairness.

D. Ablation Study

Ablation studies were conducted to understand the effect of the adversarial loss, the cost sensitive method and the effects of over-smoothing. The ablation studies were performed on the Tagged dataset.

Effects of Adversarial and Cost Sensitive Learning. To measure the effects of the adversarial learning, we set α to 0. As shown in Table III, the model without the adversarial learning shows an increase in Δ_{EO} for both FACS-GCN and AdaGCN. FACS-GCN without Adversarial compared to

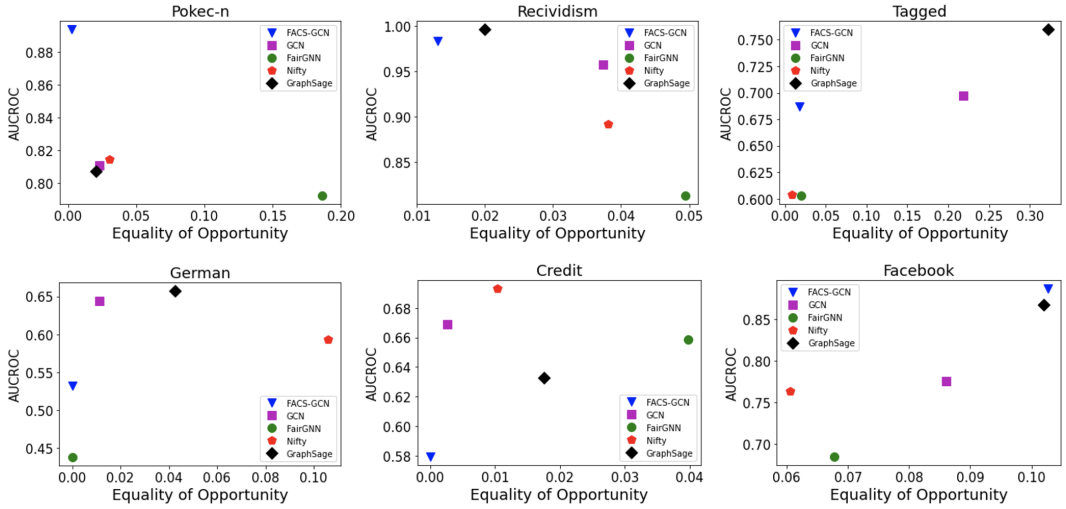

Fig. 2. Comparison of area under ROC (AUROC) and equality of opportunity (EO) metrics for various methods on 6 real-world datasets.

TABLE II
WIN-LOSS DRAW COMPARISON BETWEEN FACS-GCN AND OTHER BASELINE METHODS IN TERMS OF THEIR AUC AND Δ_{EO}.

AUC	GCN	GraphSage	FairGNN	NIFTY	FACS-GCN
GCN	-	2-4	6-0	4-2	3-3
GraphSage	4-2	-	5-1	4-2	4-2
FairGNN	0-6	1-5	-	0-6	1-5
NIFTY	2-4	2-4	6-0	-	2-4
FACS-GCN	3-3	2-4	5-1	4-2	-

Δ_{EO}	GCN	GraphSage	FairGNN	NIFTY	FACS-GCN
GCN	-	4-2	3-3	4-2	1-5
GraphSage	2-4	-	3-3	3-3	1-5
FairGNN	3-3	3-3	-	1-5	1-1-4
NIFTY	2-4	3-3	5-1	-	2-4
FACS-GCN	5-1	5-1	4-1-1	4-2	-

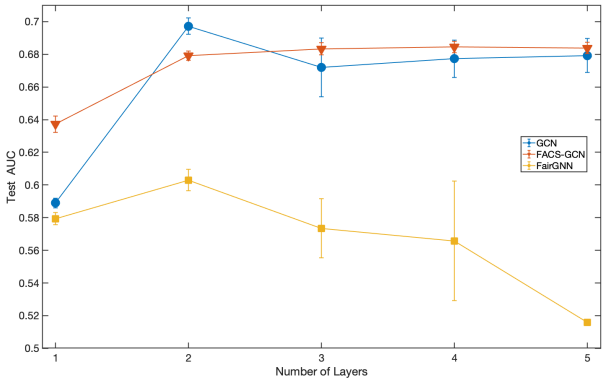

Fig. 3. Effect of over-smoothing as number of layers increases for the Tagged dataset. FairGNN performance drops significantly compared to FACS-GCN. Performance drop in GCN is not as significant for the given dataset.

FACS-GCN showed an increase by more than 20% in Δ_{EO} and its AUC also increased. To measure the effects of cost sensitive learning, the reward matrices were set to identity matrices, which reduces to AdaGCN. For a fair comparison, α was set to 0.75 for both FACS-GCN and AdaGCN. The difference in AUC for both model are minimal but there is a 6% difference in the recall, showing us it does give the positive class a better classification than without the cost sensitive learning. α was then set to 0 and the cost sensitive showed a even bigger effect by increasing recall by more than 15%.

Over-smoothing Effects. In Figure 3, we plotted GCN, FairGNN and FACS-GCN and we then plot the AUC as the layers increase. As shown in the figure, FACS-GCN does not suffer from oversmoothing as it is able to maintain or increase its AUC as layers are added. The same cannot be said about FairGNN as it decreases after 3 layers, dropping to 0.5 AUC by the fifth layer. As for GCN, the performance drop was observed from 2 to 3 layers but somewhat stabilizes after that.

TABLE III
RESULTS OF ABLATION STUDIES.

	AUC	Recall	Δ_{EO}
AdaGCN	0.718±0.004	0.314±0.003	0.116±0.008
FACS no-adv	0.712±0.012	0.467±0.005	0.235±0.004
AdaGCN-adv	0.681±0.003	0.139±0.004	0.060±0.014
FACS-GCN	0.687±0.001	0.201±0.020	0.013±0.008

VI. CONCLUSION

This paper presents a novel graph neural network framework called FACS-GCN to alleviate two types of biases present in real-world network data. The framework employs a cost-sensitive exponential loss approach to overcome the class imbalance problem as well as an adversarial learning strategy to debias the algorithm from discriminating against certain protected groups. By leveraging the AdaGCN framework, we demonstrate that FACS-GCN can overcome the oversmoothing problem, thereby reducing the accuracy loss when imparting fairness into the learning framework. Experimental results on

6 benchmark network datasets validated the effectiveness of the proposed framework.

VII. Acknowledgement

This material is based upon work supported by the NSF Program on Fairness in AI in collaboration with Amazon under grant IIS-1939368. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Amazon.

References

[1] F. Scarselli, A. C. Tsoi, M. Hagenbuchner, and L. Di Noi, "Solving graph data issues using a layered architecture approach with applications to web spam detection," *Neural Networks*, vol. 48, pp. 78–90, 2013.

[2] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, and Y. Qi, "A semi-supervised graph attentive network for financial fraud detection," *Proc of IEEE Int'l Conf on Data Mining (ICDM)*, pp. 598–607, 2019.

[3] A. K. Ahmad, A. Jafar, and K. Aljoumaa, "Customer churn prediction in telecom using machine learning in big data platform," *Journal of Big Data*, vol. 6, no. 1, pp. 1–24, 2019.

[4] G. Jin, Q. Wang, C. Zhu, Y. Feng, J. Huang, and J. Zhou, "Addressing crime situation forecasting task with temporal graph convolutional neural network approach," *12th Int'l Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, pp. 474–478, 2020.

[5] Z. Xiaojin and G. Zoubin, "Learning from labeled and unlabeled data with label propagation," *Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University*, 2002.

[6] J. Scripps, P.-N. Tan, F. Chen, and A.-H. Esfahanian, "A matrix alignment approach for collective classification," *Proc of Int'l Conf on Advances in Social Network Analysis and Mining*, pp. 155–159, 2009.

[7] J. Neville and D. Jensen, "Iterative classification in relational data," in *Proc of AAAI-2000 Workshop on learning statistical models from relational data*, 2002.

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

[9] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk," *Proc of the 20th ACM SIGKDD Int'l Conf on Knowledge Discovery & Data Mining*, Aug 2014. [Online]. Available: http://dx.doi.org/10.1145/2623330.2623732

[10] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, pp. 1025–1035, 2017.

[11] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ArXiv*, vol. abs/1609.02907, 2017.

[12] S. Caton and C. Haas, "Fairness in machine learning: A survey," *CoRR*, vol. abs/2010.04053, 2020. [Online]. Available: https://arxiv.org/abs/2010.04053

[13] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[14] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," *Proc of the 3rd innovations in theoretical computer science conference*, pp. 214–226, 2012.

[15] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in Neural Information Processing Systems*, pp. 3315–3323, 2016.

[16] E. Dai and S. Wang, "Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information," *Proc of the 14th ACM Int'l Conference on Web Search and Data Mining*, pp. 680–688, 2021.

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

[18] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini, "Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning," *IEEE Transactions on Artificial Intelligence*, 2021.

[19] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," *Proc. of the AAAI Conf. on Artificial Intelligence*, pp. 3538–3545, 2018.

[20] K. Sun, Z. Zhu, and Z. Lin, "Adagcn: Adaboosting graph convolutional networks into deep models," in *Int'l Conf on Learning Representations*, 2020.

[21] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.

[22] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," *Social network data analytics*, pp. 115–148, 2011.

[23] S. Xiao, S. Wang, Y. Dai, and W. Guo, "Graph neural networks in node classification: survey and evaluation," *Machine Vision and Applications*, vol. 33, no. 1, pp. 1–19, 2022.

[24] Y. Wang, C. Aggarwal, and T. Derr, "Distance-wise prototypical graph neural network in node imbalance classification," *arXiv preprint arXiv:2110.12035*, 2021.

[25] M. Shi, Y. Tang, X. Zhu, D. Wilson, and J. Liu, "Multi-class imbalanced graph convolutional network learning," *Proc of the 29th Int'l Joint Conf on Artificial Intelligence*, pp. 2879–2885, 2020.

[26] T. Zhao, X. Zhang, and S. Wang, "Graphsmote: Imbalanced node classification on graphs with graph neural networks," *Proc of the 14th ACM Int'l Conf on Web Search and Data Mining*, Mar 2021.

[27] T. Adel, I. Valera, Z. Ghahramani, and A. Weller, "One-network adversarial fairness," *Proc. of the AAAI Conf. on Artificial Intelligence*, pp. 2412–2420, 2019.

[28] P. Adler, C. Falk, S. A. Friedler, T. Nix, G. Rybeck, C. Scheidegger, B. Smith, and S. Venkatasubramanian, "Auditing black-box models for indirect influence," *Knowledge and Information Systems*, vol. 54, no. 1, pp. 95–122, 2018.

[29] I. Chen, F. D. Johansson, and D. Sontag, "Why is my classifier discriminatory?" *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[30] O. Bastani, X. Zhang, and A. Solar-Lezama, "Probabilistic verification of fairness properties via concentration," *Proc of the ACM on Programming Languages*, vol. 3, no. OOPSLA, pp. 1–27, 2019.

[31] F. Kamiran, T. Calders, and M. Pechenizkiy, "Discrimination aware decision tree learning," *2010 IEEE Int'l Conf on Data Mining*, pp. 869–874, 2010.

[32] W. Zhang and E. Ntoutsi, "Faht: an adaptive fairness-aware decision tree classifier," *Proc of the 28th Int'l Joint Conf on Artificial Intelligence*, pp. 1480–1486, 2019.

[33] V. Iosifidis and E. Ntoutsi, "Adafair," *Proc of the 28th ACM Int'l Conf on Information and Knowledge Management*, Nov 2019. [Online]. Available: http://dx.doi.org/10.1145/3357384.3357974

[34] E. Krasanakis, E. S. Xioufis, S. Papadopoulos, and Y. Kompatsiaris, "Adaptive sensitive reweighting to mitigate bias in fairness-aware classification," *Proc of the 2018 World Wide Web Conference*, 2018.

[35] S. Chiappa, "Path-specific counterfactual fairness," *Proc of the AAAI Conf on Artificial Intelligence*.

[36] U. Hébert-Johnson, M. Kim, O. Reingold, and G. Rothblum, "Multicalibration: Calibration for the (computationally-identifiable) masses," *Int'l Conf on Machine Learning*, pp. 1939–1948, 2018.

[37] F. Masrour, T. Wilson, H. Yan, P. Tan, and A. Esfahanian, "Bursting the filter bubble: Fairness-aware network link prediction," *Proc of the 34th AAAI Conf on Artificial Intelligence*, pp. 841–848, 2020.

[38] T. Rahman, B. Surma, M. Backes, and Y. Zhang, "Fairwalk: Towards fair graph embedding," *Proc of the 28th Int'l Joint Conf on Artificial Intelligence*, pp. 3289–3295, 7 2019.

[39] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class AdaBoost," *Statistics and Its Interface*, vol. 2, pp. 349–360, 2009.

[40] S. Fakhraei, J. Foulds, M. Shashanka, and L. Getoor, "Collective spammer detection in evolving multi-relational social networks," *Proc of the 21th ACM SIGKDD Int'l Conf on Knowledge Discovery & Data Mining*, pp. 1769–1778, 2015.

[41] J. Leskovec and R. Sosič, "Snap: A general-purpose network analysis and graph-mining library," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 1, p. 1, 2016.

[42] K. Jordan and T. Freiburger, "The effect of race/ethnicity on sentencing: Examining sentence type, jail length, and prison length," *Journal of Ethnicity in Criminal Justice*, vol. 13, pp. 1–18, 11 2014.

[43] I.-C. Yeh and C.-h. Lien, "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert systems with applications*, vol. 36, no. 2, pp. 2473–2480, 2009.

[44] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml