Abstract—Analysis of the fairness of machine learning (ML) algorithms has attracted many researchers’ interest. Several studies have shown that ML methods produce a bias toward different groups, which limits the applicability of ML models in many applications, such as crime rate prediction. The data used for ML may have missing values, which, if not appropriately handled, are known to further harmfully affect fairness. To address this issue, many imputation methods have been proposed to deal with missing data. However, research on the effect of missing data imputation on fairness is still rather limited. In this paper, we analyze the impact of imputation on fairness in the context of graph data (node attributes) using different embedding and neural network methods. Extensive experiments on six datasets demonstrate several issues of fairness in graph node classification when dealing with missing data and various imputation techniques. We find that the choice of the imputation method affects both fairness and accuracy. Our results provide valuable insights into fairness ML over graph data and how to handle missingness in graphs efficiently.

Index Terms—Graphs, Fairness, Experimental Study, Bias, GNNs, Equal Opportunity, Demographic Parity

I. INTRODUCTION

Many real-world datasets have some form of graph structure associated with them: social networks [11], paper citations [18], protein-protein interactions [21], etc. Numerous machine learning algorithms are proposed that particularly deal with the graph data (e.g., Graph Neural Networks (GNNs) [20], Node2Vec [19], DeepWalk [32], and Graph Summarization [3]). These algorithms have shown remarkable improvement in results on graph data [29], [36], [37]. Most of them are designed to integrate both node features and information (topological structure) about graph edges to enhance node representation and improve results [41], [42], [47]. Although these algorithms can successfully capture graph structure, the social bias in data can cause fairness issues [33]. This limits the applicability of most of the algorithms in practical applications of graph data.

Many studies regarding fairness in machine learning models have reported that typical datasets contain discrimination and social bias towards a sensitive attribute like region, age, skin color, gender, etc. [6], [10], [14], [15], [22]. Consequently, any machine learning model trained on such data can inherit bias. Recent studies on the fairness of GNNs and Node2Vec have also reported similar issues [8], [24], [33]. Although these algorithms are designed to incorporate edge information, which provides extra patterns and better representation, bias in data can still propagate through the edges. This can aggravate the fairness issue in graphs [19], [32].

Another important challenge in graph data is the presence of homophily (connections among nodes having the same attribute values) [4]. In social network graphs, nodes with similar sensitive features tend to connect with each other compared to nodes with different sensitive features. For example, older people tend to have friends in a similar age group. In such cases, aggregating features of neighbor nodes produce representations of similar age groups that are quite different from other age groups, thereby leading to severe bias in graph-specific algorithms. Recent research demonstrated that such results correlate with sensitive attributes [6], [14].

Missing data is common in every machine learning problem. And, if not dealt with properly, it has an adverse effect on fairness [5]. The most common approach to dealing with missing data is imputation (see Figure 1). Different approaches to imputation have been proposed to deal with missing data that use statistical methods and machine learning [30], [27], [40]. Missing values can further cause sensitive attribute imbalance. This implies that missing data can decrease fairness. There is limited research on the impact of missing data imputation on fairness. Recent studies have shown that missing data can contribute to bias in machine learning algorithms [5], [31], [34], [46], [45]; however, these studies do not include graph data.

In this work, we investigate the impact of missing data imputation on the fairness of node classification problem in graph data. Specifically, we perform an extensive experimental
To the best of our knowledge, no previous work has examined fairness in data imputation. Classification in graph data, fairness in node classification, and graph data. Specifically, we summarize the literature on node classification through random walks. It has shown promising algorithm to generate vector representations of nodes in a graph from nodes and edges) and then training the classifier algorithm on embeddings. A. Graph Node Classifiers

Finally, we conclude the paper in Section VI. The results are reported in Section VI. Finally, we conclude the paper in Section VII.

II. RELATED WORK

In this section, we review the research related to fairness in graph data. Specifically, we summarize the literature on node classification in graph data, fairness in node classification, and fairness in data imputation.

A. Graph Node Classifiers

Machine learning problems involving graph data are mainly solved by representing data as node embeddings (generated from nodes and edges) and then training the classifier algorithm on embeddings. DeepWalk [32] is a commonly used algorithm to generate vector representations of nodes in a graph. It generates low-dimensional representations for nodes in a graph through random walks. It has shown promising results in many graph learning problems, including node classification. Proposed in [19], Node2Vec is another well-known algorithm that uses a biased walk (breadth first, depth first) instead of a random walk. It exhibits particular promise for a variety of machine learning applications [19], [25].

Recently, graph neural networks (GNNs) have been proposed to combine node embedding and classification tasks using single neural networks. Substantial improvements in results have been reported using GNNs when compared to other node classification techniques. The GNNs can be divided into two types: spectral-based GNNs [7], [12], [23] and spatial-based GNNs [9], [21], [39]. Graph Convolutional Networks (GCNs) is an example of the spectral-based method as they implement the convolution operation on graph data [26]. Spatial-based GNNs are motivated by aggregating neighbor nodes’ information to find a node representation [21]. Graph attention networks (GATs) integrate the self-attention mechanism in spatial aggregation such that it assigns higher weights to more important nodes [39]. Both GCNs and GATs have shown excellent results on node classification tasks.

B. Fairness in Graph Node Classification

Although, fairness in machine learning is a well-studied problem [6], [10], [15], [22]. Little attention has been paid to the issue of fairness in graph node classification [17]. Extant literature explores this issue in graph node embeddings or node classifiers.

The first group of research works focuses on the design of an embedding mechanism that would be fair for the sensitive attribute [8], [24], [33]. Most of these works pertain to improving Node2Vec [19] or DeepWalk [32] algorithms by creating fair embeddings which target specific applications of graph data.

The second group of research works aims to design an end-to-end solution for a fair node classification problem. These solutions can take care of embeddings and classification simultaneously. For instance, Dai et al. [11] proposed a fair node classifier called FairGNN. It can work on data with limited sensitive attribute information. The basic idea is to design a loss function that incorporates fairness in the problem.

C. Fairness in Data Imputation

Existing research on fairness in data imputation is mainly limited to tabular data. Zhang et al. [46] studied various imputation techniques and reported that data imputation could potentially introduce more bias in machine learning models. They conclude that data imputation further harms the imbalance of sensitive attributes, causing unfairness (e.g., if the males have more missingness as compared to the females). Another work that addresses the problem of accessing fairness under missing data is presented in [45]. In this work, authors trained machine learning models on fully observed data (with completely observed features) and then tested them on missing data. They reported that missing data produced challenges in the fairness of classifiers and showed that a model which is trained using only a subset of the whole dataset (complete case...
domain) might be fair for the chosen subset but not for the whole dataset (complete data domain). Compared to the above research, we focus on the case of graph data and study the impact of data imputation on the fairness of node classifiers.

III. PRELIMINARIES

In this section, we first define the notation used throughout the paper. Then, we explain different notions of fairness used in the literature. We also introduce the methods used to introduce missingness in data and summarize the existing techniques for data imputation.

A. Notation

Assume an attributed graph that has attributes associated with nodes. For example, social networks contain attributes of individuals. Let $G = (V, E, X, Y)$ denote an attributed graph, where $V = \{v_1, v_2, ..., v_N\}$ is the set of $N$ nodes, and $E \subseteq V \times V$ is the set of edges between them. There is a set of feature vectors $X = \{x_1, x_2, ..., x_N\}$ and labels $Y = \{y_1, y_2, ..., y_N\}$ associated with $V$. Each node $v_i$ has a feature vector $x_i \in \mathbb{R}^K$, and a binary label $y_i \in \{+,-\}$ associated with it. There is a binary sensitive attribute $A \in \{a, b\}$ in the dataset, where $a$ is the majority group and $b$ is the minority group. The node classification problem is to predict a label (represented by probability $P$) for each node such that the algorithm supposes not to produce results that discriminate based on $A$.

B. Measures of Fairness

Since machine learning algorithms make predictions based on the available data, the bias is data can propagate in results. Fairness in machine learning algorithms is generally assessed based on a predefined sensitive attribute $A$. Note that, we focus on binary-sensitive attributes and binary node classification problem.

We use demographic parity and equal opportunity as measures of fairness. Both of these are widely used by the research community to assess the fairness of machine learning models in graph data [1], [8], [11], [13], [16], [24], [44]. For a binary sensitive attribute $A \in \{a, b\}$, and binary class label $y \in \{+,-\}$, such that $TP$, $TN$, $FP$, $FN$ represents the true positive, true negative, false positive, false negative with respect to $a$ or $b$, the fairness measures are defined below.

- Demographic (statistical) parity is defined as the probability of being assigned to a positive class is independent of the sensitive attribute, i.e., $P(+|a) = P(+|b)$ $\forall a, b \in A$. This requires the acceptance rates of the candidates from both groups to be equal, thus introducing fairness with respect to sensitive attributes. We use the parity difference between the majority and minority groups as a fairness measure($\Delta_{SP}$).

$$\Delta_{SP} = (TP_a + FP_a) - (TP_b + FP_b)$$

- Equal opportunity is defined as the False negative rate (FNR) for all the sensitive attribute groups is equal, i.e., $P(-|+, a) = P(-|+, b)$ $\forall a, b \in A$. We take the difference of FNR between the majority and minority groups to measure fairness($\Delta_{EO}$).

$$\Delta_{EO} = \frac{FN_a}{TP_a + FN_a} - \frac{FN_b}{TP_b + FN_b}$$

C. Types of Missing Data

We consider missing data from two perspectives: the patterns of missingness and the attributes with missingness. The patterns of missingness in the data can occur in the following three ways [28].

- **Missing Completely at Random (MCAR):** The missing data is called MCAR if the missingness is independent of the observed and missing(unobserved) data.
- **Missing at Random (MAR):** If the missingness is only dependent upon the observed data, the missing mechanism is MAR. The missing data is dependent on only the complete(observed) cases.
- **Missing Not at Random (MNAR):** For MNAR, the missingness is dependent on observed and unobserved data (missingness depends on complete and incomplete cases). In MNAR, the missingness occurs in the sensitive attribute depending on the value of the sensitive attribute.

We categorize missingness in attributes into two types. First, we consider missingness from sensitive attributes only (represented by $K_1$). Secondly, we consider missing from $K$ attributes, where $K$ is a predefined number. We select $K$ to be 10% and 40% of the total number of attributes ($K_{10}$, $K_{40}$) in the dataset. We consider real graph datasets to assess fairness. The missing data is introduced artificially in the datasets, and $K$ most correlated attributes are selected. The sensitive attribute is also a part of $K_{10}$ and $K_{40}$ even if it is not the most correlated feature. The missing data are imputed using different imputation techniques, and fairness is computed. Given a sample $x_i = [s_1, s_2, ..., s_N]$, the probability that $s_j, \forall j \in \{1, 2, ..., N\}$ is missing is given in Table I. The $a$ and $b$ in MAR (Table I) represent the majority and the minority group, respectively. Note that producing missingness in sensitive attribute based on sensitive attribute will leads MAR to MNAR, due to which the first four rows of MAR are empty. Similarly, we do not include the sensitive attribute for $3b, 4b, 7b, 8b$ as it will also lead MAR to MNAR. Also note that all the probabilities in Table I are truncated within $[0, 1]$.

D. Imputation Techniques

We use four imputation techniques that are commonly used in data imputation research.

**Mean imputation (Mean):** The method imputes the missing with the mean of all the samples with observed values of the same attribute [35]. It is easy to compute the mean, but it cannot capture the temporal trend in the data.
TABLE I: Data missing mechanisms. Note that for MAR based on iterative soft thresholding of SVD decomposition, which is same attribute based on a distance metric to sample missing value using the KNNs that have observed values in the neighborhood that the added flexibility in exploring neighborhoods is the key to learning richer representations of nodes in graphs [19], [25].

K-nearest neighbors (KNN): The method imputes the missing value using the KNNs that have observed values in the same attribute based on a distance metric to sample [35]. KNN is a robust imputation technique and can capture complex data distribution easily. However, it is computationally expensive, and the results depend on the selection of hyperparameters.

Iterative SVD (SVD): The method imputes missing values based on matrix completion with iterative low-rank SVD decomposition inspired by [38]. SVD can work for very sparse data. However, for large datasets, it is a computationally expensive operation.

Soft Imputation (SI): Matrix completion method by iterative soft thresholding of SVD decomposition, which is based on [43]. It uses the spectral regularization method for imputing incomplete matrices.

IV. NODE CLASSIFICATION ALGORITHMS

We use a range of algorithms, from fair oblivious to fair node classifiers. Detail of these algorithms is given below.

A. Fairness Oblivious Node Classifiers

Fair oblivious node classifiers do not consider or try to improve the fairness of the model. We select two fair oblivious classifiers, Node2Vec and Graph Convolutional Networks (GCN), since they are widely used by the research community in graph node classification.

Node2Vec: It is an algorithm to generate vector representations of nodes on a graph. The Node2Vec framework learns low-dimensional representations for nodes in a graph through the use of random walks. Besides reducing the engineering effort, representations learned by the algorithm lead to greater predictive power. The algorithm generalizes prior work, which is based on rigid notions of network neighborhoods, and argues that the added flexibility in exploring neighborhoods is the key to learning richer representations of nodes in graphs [19], [25].

Graph Convolutional Networks (GCN): This method is similar to convolutional neural networks (CNNs) in terms of weight sharing. The main difference lies in the data structure, where GCNs are the generalized version of CNNs that can work on data with underlying non-regular structures. Providing information about edges to GCNs enables the model to learn the features of neighboring nodes. This mechanism can be seen as a message-passing operation along the nodes within the graph [26]. After hyperparameter tuning, we select one convolution layer of size $8 \times 8$ with RELU activation function followed by a dense layer with a sigmoid activation function. This architecture is used for all datasets.

B. Fair Node Classifiers

The goal of fair node classifiers is to improve the fairness of the model without compromising accuracy. They must achieve this by modifying the classifier’s random walk process or objective function.

FairWalk [33]: This method relies on a modification of the random walks to induce fairness in node embeddings. This method modifies the transition probability of Node2Vec for the generation of unbiased traces. It works well on graphs in which the neighboring nodes have different sensitive attributes.

CrossWalk [24]: This method enhances the fairness of various graph algorithms applied to node embeddings, including influence maximization, link prediction, and node classification. It is based on a re-weighting procedure for building
the random walks and can therefore be used with any random walk-based algorithms, including Node2Vec. CrossWalk aims to address the shortcomings of FairWalk by assigning more weights to both edges that connect nodes on the boundary of the same groups and edges connecting nodes from different groups.

V. EXPERIMENTAL SETUP

In this section, we describe the datasets used in our study of fairness. We also provide implementation details such as pre-processing of data and experimental parameters.

A. Dataset Statistics

For this study, we used six datasets, ranging from social network graphs to credit defaulter and bail datasets. The statistics of these datasets are given in Table II. The details are given below.

1) Pokec: It is a social network in Slovakia [11], similar to Facebook and Twitter. Nodes are user profiles, and edges are friendships between the nodes. This dataset contains anonymized data from the whole social network in 2012. User profiles contain gender, age, hobbies, interests, education, working field and etc. The original Pokec dataset contains millions of users. Based on the provinces that users belong to, we sampled two datasets named Pokec-z and Pokec-n. The Pokec-z and Pokec-n consists of users belonging to two major regions of the corresponding provinces. We treat the region as a sensitive attribute. The classification task is to predict the working field of the users.

2) NBA: It contains performance statistics of players in the 2016-2017 [11] season along with information such as nationality, age, and salary for 400 NBA basketball players. Nodes are players, and edges are their relationship on Twitter. To obtain the graph that links the NBA players together, we collect the relationships of the NBA basketball players on Twitter with its official crawling API. We binarize the nationality into two categories, i.e., U.S. players and overseas players, which are used as sensitive attributes. The classification task is to predict whether or not the player’s salary is over the median.

3) German Credit Graph: The German credit graph represents clients in a German bank that are connected based on the similarity of their credit accounts. The task is to classify clients into good vs. bad credit risks considering the client’s gender as a sensitive attribute. The German Graph credit dataset [1], [13], [44] classifies people described by a set of attributes as good or bad credit risks. It consists of attributes like gender, loan amount, and other account-related features of 1,000 clients(nodes) and 25000 edges. The task is to classify clients into good vs. bad credit risks considering the client’s gender as a sensitive attribute. Other attributes include age, single, loan information, employment, customer finance, and assets data.

4) Credit Defaulters Graph: The Credit defaulter graph has 30000 nodes representing individuals that we connected based on the similarity of their spending and payment patterns [1], [13], [16], [44]. The task is to predict whether or not an individual will default on the credit card payment or not while considering age as a sensitive attribute. It contains features like education, credit history, age, and features derived from their spending and payment patterns.

5) Recidivism: The Recidivism graph has 18876 nodes representing defendants who got released on bail at the U.S state courts during 1990-2009 [1], [13], [16], [44]. Defendants are connected based on the similarity of past criminal records and demographics. 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), considering race information as the protected attribute. The data contains education, age, and other personal attributes.

| Dataset | |V_G| | |E_G| |Average Degree | Density | |Attributes| |
|---------|-------|-------|-------|-------|-------|----------------|--------|-------|----------------|
| Pokec-n | 66569 | 517047 | 4.71 | 0.00071 | 268 |
| Pokec-z | 67796 | 617958 | 5.17 | 0.00064 | 279 |
| NBA | 400 | 10621 | 45.90 | 0.148 | 98 |
| German Credit | 1000 | 21742 | 43.48 | 0.043 | 30 |
| Recidivism | 18876 | 311870 | 33.04 | 0.0017 | 19 |
| Credit Defaulters | 30000 | 1421858 | 94.79 | 0.0031 | 15 |

Fig. 3: Comparison of FairWalk with and without node attributes for all datasets. FairWalk technique represents only node embeddings without introducing missingness in the data. German and Credit represent German Credit and Credit Defaulter datasets, respectively. Figure best seen in color.
Fig. 4: Results comparison using different imputation techniques for German Credit dataset and MCAR. Y-axis represents the difference of results in terms of $L_2$ norm. $D_1$ represents the difference in results from KNN: KNN-Mean CrossWalk, $D_2$: KNN-SVD CrossWalk, $D_3$: KNN-SI CrossWalk, $D_4$: KNN-Mean FairWalk, $D_5$: KNN-SVD FairWalk, $D_6$: KNN-SI FairWalk, $D_7$: KNN-Mean GCN, $D_8$: KNN-SVD GCN, $D_9$: KNN-SI GCN, $D_{10}$: KNN-Mean Node2Vec, $D_{11}$: KNN-SVD Node2Vec, $D_{12}$: KNN-SI Node2Vec. Figure best seen in color.

B. Implementation Details

The NBA dataset contains missing values. We remove the nodes with missing values, after which the size of the dataset is around 300 nodes. We also remove the corresponding edges of missing nodes. Similarly, Pokec-z and Pokec-n also contain a large number of nodes with missing attributes, which we remove, after which the number of nodes for Pokec-z and Pokec-n is around 6500 and 8000, respectively.

We first transform the datasets with Min-Max normalization. We use 70% of the data for training and validation purposes. The remaining 30% data is used for testing purposes. We stratify the train test split with respect to sensitive attributes to counter any imbalance. The missingness is artificially introduced in the training data according to Table I, the missing data is imputed, node classifiers are trained using imputed data, and results are computed. Experiments are performed on core i7, a 9th generation machine with 16 GB memory. All experiments are performed five times, and the mean results are reported.

For the implementation of FairWalk and Crosswalk we use their respective GitHub repository. For the implementation of Node2Vec and GCN we use Stellargraph library. We set the embedding dimension as 128 for CrossWalk, FairWalk, and Node2Vec. The remaining parameters are the same as stated in their papers [19], [24], [33]. Since node embeddings only use edge information, they do not depend upon node features. There is no point in missing data imputation in CrossWalk, FairWalk, and Node2Vec. To counter this issue, we concatenate node embeddings with respective node features. Logistic regression is used as a classifier in Node2vec, CrossWalk, and FairWalk. The architecture of GCN is already explained in Section IV. Binary cross entropy (BCE) is used as a loss function for GCN. An ADAM optimizer with a 0.01 learning rate is used to train GCN. For the implementation of Mean, KNN, SVD, and SI, we have used an open source library FancyImpute. The hyperparameters of KNN, SVD, and SI are the same as those given by FancyImpute. All our code and datasets used are available at Github.

Fig. 5: Results comparison of different classifiers and missing mechanisms for German Credit dataset. None represents data without missingness, remaining scenarios are explained in Table I. Figure best seen in color.

VI. RESULTS AND DISCUSSIONS

In this section, we present our findings regarding the impact of missing data imputation on the performance and fairness of graph node classifiers. We use statistical parity and equal opportunity as measures of fairness and accuracy to assess the classifiers’ performance. Specifically, we investigate the effect of (i) imputation techniques on fairness or accuracy, (ii) an increase in the probability of missingness, (iii) an increase in the number of attributes of missingness, and (iv) a fair oblivious versus fair classifier. Due to the page limit, we only present results for KNN imputation. Our main findings are reported in the form of a scatter plot in Figure 5 (German Credit), Figure 6 (Pokec-N), Figure 7 (Pokec-Z), Figure 8 (Credit Defaulter), Figure 9 (Recidivism), and Figure 10 (NBA).

1https://github.com/arielsinger/fairwalk
2https://github.com/ahmadkhajehnejad/CrossWalk
3https://github.com/stellargraph/stellargraph
4https://github.com/iskande/fancyimpute
5https://github.com/harisalimansoor/FairnessGraphDataImputation
which is also more than 100% with \( \Delta \) without missingness, remaining scenarios are explained in (1a) to (8) Table equal opportunity \( \Delta \) shows a similar behaviour (Figure 6). We find that as the missing probability increases, both \( \Delta_{SP} \) and \( \Delta_{EO} \) tend to increase for almost all datasets and all classifiers. This can be observed from 1a (MCAR) to 4a (MCAR) in Figure 5, 6, 7, 8, 9, and 10. The effect on German Credit dataset is quite worse and \( \Delta_{SP} \) increases from 7.08 (1a) to 18.92 (4a) which is more than 150% increase in \( \Delta_{SP} \) (Figure 5b). In the case of equal opportunity, the German dataset also performs worse with \( \Delta_{EO} \) increases from 2.25 (1a) to 15.12 (4a), which is more than 500% increase. The Credit Defaulter dataset also shows a similar behaviour (Figure 8), \( \Delta_{SP} \) increases from 4.36 (1a) to 8.77 (4a) which is about 100% increase. Similarly, the equal opportunity \( \Delta_{EO} \) increases from 2.84 (1a) to 5.79 (4a) which is also more than 100% increase. Recidivism, Pokec-z, Pokec-n, and NBA datasets also produce negative results on fairness as missing probability increases, but they are not as severe as the German and Credit Defaulter dataset.

A. Effect of MCAR

In this section, we show the effect of increasing values of missing probability and increasing value of \( K \).

1) Impact of Increasing Missing Probability: We find that as the missing probability increases, both \( \Delta_{SP} \) and \( \Delta_{EO} \) tend to increase for almost all datasets and all classifiers. This can be observed from 1a (MCAR) to 4a (MCAR) in Figure 5, 6, 7, 8, 9, and 10. The effect on German Credit dataset is quite worse and \( \Delta_{SP} \) increases from 7.08 (1a) to 18.92 (4a) which is more than 150% increase in \( \Delta_{SP} \) (Figure 5b). In the case of equal opportunity, the German dataset also performs worse with \( \Delta_{EO} \) increases from 2.25 (1a) to 15.12 (4a), which is more than 500% increase. The Credit Defaulter dataset also shows a similar behaviour (Figure 8), \( \Delta_{SP} \) increases from 4.36 (1a) to 8.77 (4a) which is about 100% increase. Similarly, the equal opportunity \( \Delta_{EO} \) increases from 2.84 (1a) to 5.79 (4a) which is also more than 100% increase. Recidivism, Pokec-z, Pokec-n, and NBA datasets also produce negative results on fairness as missing probability increases, but they are not as severe as the German and Credit Defaulter dataset. Another interesting observation is that equal opportunity difference, although increases with missing probability but remains more stable than parity difference. Increasing the missing probability also negatively affects accuracy but is not as adverse as fairness. German credit, NBA, and Pokec-n show a decrease in accuracy in the case of 4a(MCAR) for different classification algorithms (Figure 5.6.10).

2) Impact Of Increasing \( K \): As the number of attributes of missingness \( K \) increases, a general trend of increase in both \( \Delta_{SP} \) and \( \Delta_{EO} \) is observed for almost all datasets and all methods. This trend is present in almost all probabilities (0.1, 0.3, 0.5, 0.7), keeping probability constant and changing \( K \) from 10% to 40% increases the \( \Delta_{SP} \) and \( \Delta_{EO} \). We can see this by comparing 1a – 4a, 5a – 8a, and 9a – 12a in the result figures. The accuracy is also more severely affected by increasing \( K \), which can be observed from 8a and 12a in the results tables. The accuracy remains consistent for the Credit Defaulter dataset showing robustness (8). We have noticed some anomalies in the NBA dataset, which can be explained by the small sample size (300). Due to the small sample size, little changes in the confusion matrix cause the fairness and accuracy to fluctuate (Figure 10). By analyzing the results, we conclude that increasing the probability and \( K \) generally have a negative impact on fairness and accuracy. However, the specific effect varies greatly on the dataset and the algorithm used.

B. Effect of MAR

In MAR, 1b and 5b represent missingness based on the majority attribute, while 2b and 6b represent MAR from minority attributes. We can observe that both fairness measures are more affected in 5b and 6b as compared to 1b and 2b, concluding increasing \( K \) causes more bias in the case of MAR on the basis of the sensitive attribute.

In some datasets (German Credit, Credit Defaulter), 1b is more affected in terms of fairness as compared to 2b, while in the remaining datasets, we can observe the opposite behavior. This is due to the sensitive class imbalance. In German Credit and Credit Defaulter datasets, the samples with majority sensitive attributes are more than the minority; thus, MAR using majority attributes produces more imbalance. Another observation from comparing 3b, 4b to 1b, 2b (5b, 6b to 7b, 8b) is that MAR on the basis of sensitive attributes affects the results more as compared to MAR based on other attributes (3b, 4b). This also depicts the effect of sensitive attributes on fairness. In MAR, we see an effect of fairness accuracy trade-off in some cases, where a decrease in bias (increase in fairness) is associated with a decrease in accuracy. This behavior is also reported by [46]. We can see this behavior in 7b and 6b for the German credit dataset, 6b and 8b in Recidivism dataset, 6b in the credit defaulter dataset, 8b in the Pokec dataset, and 5b and 8b in the NBA dataset.

C. Effect of MNAR

We assume that node attributes follow a uniform distribution. Under this assumption, the probability of missingness increases from 1c to 4c and similarly from 5c to 8c (9c to 12c). Moreover, 1c, 5c, and 9c have the same probability of missing. Only the number of attributes of missingness increases. We notice that as the missing probability increases, both fairness measures tend to increase for almost all datasets and all classifiers for MNAR. This can be observed from 1c
Fig. 7: Results comparison of different classifiers and missing mechanisms for the Pokec-z dataset. None represents data without missingness, remaining scenarios are explained in Table I. Figure best seen in color.

This implies that GCN is least affected by any imputation method. We can observe this in both accuracy and fairness measures ($D_7, D_8, D_9$). Similarly, Node2Vec and CrossWalk have the worst results and are badly affected by the imputation method. In terms of the magnitude of difference, accuracy is least affected by the selection of a specific imputation method, while parity difference is most affected with $L_2$ norm difference as high as 25 ($D_{10}, D_{11}, D_{12}$).

E. Comparison of Fair Oblivious vs. Fair Classifiers

We use two fair oblivious classifiers (Node2Vec, GCN) and two fair classifiers (FairWalk, CrossWalk). As stated earlier, we concatenate node embeddings with node features for Node2Vec, FairWalk, and CrossWalk. It is worth noting that using FairWalk and CrossWalk with only node embedding does improve fairness, but at the cost of compromised accuracy Figure 3, 2. Another interesting observation comes from NBA data. Instead of improving fairness, Both FairWalk and CrossWalk reduce fairness, increasing bias. This can be explained in terms of the very small sample size of the NBA dataset.

Concatenating node attributes increases accuracy, but fairness is compromised. Also, the increase in accuracy is not as compared to Node2Vec or GCN. This implies that the notion of fairness in FairWalk and CrossWalk is restricted to edges, and a true fair node classifier is still an open task. A trade-
off between fairness and accuracy still exists in FairWalk and CrossWalk. We have explained earlier that such a trade-off further increases in MAR and MNAR. Another interesting observation comes from GCN (fair oblivious classifier), which performs best in terms of fairness without compromising accuracy. For the Recidivism dataset, GCN also performs best in terms of accuracy Figure 9. This is due to the robust design of GCN, which integrates both edges and features. Incorporating fairness in GCN can further improve fairness without compromising accuracy.

VII. CONCLUSION

In this work, we explore the effect of missing data imputation on the fairness of graph node classifiers. We performed a detailed experimental study using six datasets, different imputation techniques, fair oblivious, and fair node classifiers. Our results demonstrate that severe fairness issues exist in missing data imputation of graphs, describing the first known empirical research in this direction. Although fair node classifiers improve fairness, it comes at the cost of compromised accuracy. Moreover, fair node classifiers are also affected by data imputation. A similar trade-off between accuracy and fairness is widely observed in the results. Compared to fair node classifiers, GCN shows promising results with improved fairness and accuracy. Most fairness issues with missing data are associated with sensitive class imbalance.

Different imputation methods also affect fairness and accuracy. The exact effect on fairness and accuracy is subject to the used classifier, imputation technique, and the specific dataset. This research offers fertile observations for future work because there is limited progress in this direction.

REFERENCES

[1] Agarwal, C., Lakkaraju, H., Zitnik, M.: Towards a unified framework for fair and stable graph representation learning. In: Uncertainty in Artificial Intelligence. pp. 2114–2124. PMLR (2021)
[2] Ali, S.: Evaluating covid-19 sequence data using nearest-neighbors based network model. arXiv preprint arXiv:2211.10546 (2022)
[3] Ali, S., Ahmad, M., Beg, M.A., Khan, I., Zaman, A., Khan, M.A.: Ssag: Summarization and sparsification of attributed graphs. arXiv preprint arXiv:2109.15111 (2022)
[4] Ali, S., Shakeel, M.H., Khan, I., Faizullah, S., Khan, M.A.: Predicting attributes of nodes using network structure. ACM Transactions on Intelligent Systems and Technology (TIST) 12(2), 1–23 (2021)
[5] Bakker, M., Valdés, H.R., D Tu, P., Gummadi, K.P., Varshney, K.R., Weller, A., Pentland, A.S.: Fair enough: Improving fairness in budget-constrained decision making using confidence thresholds (2020)
[6] Beutel, A., Chen, J., Zhao, Z., Chi, E.H.: Data decisions and theoretical implications when adversarially learning fair representations. arXiv preprint arXiv:1707.00075 (2017)
[7] Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
[8] Buyyl, M., De Bie, T.: Debayes: a bayesian method for debiasing network embeddings. In: International Conference on Machine Learning. pp. 1220–1229. PMLR (2020)
[9] Chiang, W.L., Liu, X., Si, S., Li, Y., Bengio, S., Hsieh, C.J.: Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 257–266 (2019)

[10] Creager, E., Madras, D., Jacobsen, J.H., Weis, M., Swersky, K., Pittasi, T., Zemel, R.: Flexibly fair representation learning by disentanglement. In: International conference on machine learning. pp. 1436–1445. PMLR (2019)

[11] Dai, E., Wang, S.: Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In: International Conference on Web Search and Data Mining. pp. 680–688 (2021)

[12] Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems 29 (2016)

[13] Dong, Y., Liu, N., Jalilian, B., Li, J.: Edits: Modeling and mitigating data bias for graph neural networks. In: Proceedings of the ACM Web Conference 2022. pp. 1259–1268 (2022)

[14] Dwork, C., Hardt, M., Pittasi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd innovations in theoretical computer science conference. pp. 214–226 (2012)

[15] Edwards, H., Storkey, A.: Censoring representations with an adversary. arXiv preprint arXiv:1511.05897 (2015)

[16] Fan, W., Liu, K., Xie, R., Liu, H., Xiong, H., Fu, Y.: Fair graph autoencoder for unbiased graph representations with wasserstein distance. In: 2021 IEEE International Conference on Data Mining (ICDM). pp. 1054–1059. IEEE (2021)

[17] Garg, P., Villasenor, J., Foggio, V.: Fairness metrics: A comparative analysis. In: 2020 IEEE International Conference on Big Data (Big Data). pp. 3662–3666. IEEE (2020)

[18] Giles, C.L., Bollacker, K.D., Lawrence, S., Citeseer: An automatic citation indexing system. In: Proceedings of the third ACM conference on digital libraries. pp. 89–98 (1998)

[19] Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)

[20] Hamaguchi, T., Iiwa, H., Shimbó, M., Matsumoto, Y.: Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. arXiv preprint arXiv:1706.05674 (2017)

[21] Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems 30 (2017)

[22] Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. Advances in neural information processing systems 29 (2016)

[23] Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163 (2015)

[24] Khajehnejad, A., Khajehnejad, M., Babaei, M., Gummidi, K.P., Weller, A., Mirzasoleiman, B.: Crosswalk: fairness-enhanced node representation learning. In: AAAI Conference on Artificial Intelligence. vol. 36, pp. 7370–7377 (2019)

[25] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[26] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[27] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[28] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[29] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[30] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[31] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[32] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[33] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[34] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[35] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[36] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[37] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[38] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[39] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[40] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[41] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[42] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[43] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[44] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[45] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[46] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)

[47] Khosla, M., Setty, V., Anand, A.: A comparative study for unsupervised methods for dna microarrays. Bioinformatics 17(6), 866–872 (2001)