Subgroup Generalization and Fairness of Graph Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite enormous successful applications of graph neural networks (GNNs) recently, theoretical understandings of their generalization ability, especially for node-level tasks where data are not independent and identically-distributed (IID), have been sparse. The theoretical investigation of the generalization performance is beneficial for understanding fundamental issues (such as fairness) of GNN models and designing better learning methods. In this paper, we present a novel PAC-Bayesian analysis for GNNs under a non-IID semi-supervised learning setup. Moreover, we analyze the generalization performances on different subgroups of unlabeled nodes, which allows us to further study an accuracy-(dis)parity-style (un)fairness of GNNs from a theoretical perspective. Under reasonable assumptions, we demonstrate that the distance between a test subgroup and the training set can be a key factor affecting the GNN performance on that subgroup, which calls special attention to the training node selection for fair learning. Experiments across multiple GNN models and datasets support our theoretical results.

1 Introduction

Graph Neural Networks (GNNs) [11, 30, 16] are a family of machine learning models that can be used to model non-Euclidean data as well as inter-related samples in a flexible way. Recent years have witnessed enormous successful applications of GNNs in various areas, such as drug discovery [14], computer vision [24], transportation forecasting [42], recommender systems [41], etc. Depending on the type of prediction target, the application tasks can be roughly categorized into node-level, edge-level, subgraph-level, and graph-level tasks [39].

In contrast to the huge empirical success in practice, theoretical understandings of the generalization ability of GNNs have been rather limited. Among the existing literature, some studies [9, 10, 21] focus on the analysis of graph-level tasks where each sample is an entire graph and the training data are IID samples of graphs. A very limited number of studies [31, 36] explore GNN generalization for node-level tasks but they assume the training nodes (and their associated neighborhoods) are IID samples, which does not align with the commonly seen graph-based semi-supervised learning setups. Baranwal et al. [3] investigate GNN generalization under a specific data generating mechanism.

In this work, our first contribution is to provide a novel PAC-Bayesian analysis for the generalization ability of GNNs on node-level tasks with non-IID assumptions about training nodes. In particular, we assume the node features are fixed and the node labels are independently sampled from distributions conditioned on the node features. We also assume the training set and the test set can be chosen as arbitrary subsets of nodes on the graph. We first prove two general PAC-Bayesian generalization bounds (Theorem 1 and Theorem 2) under this non-IID setup, and then derive a generalization bound for GNN (Theorem 3) in terms of characteristics of the GNN models and the node features.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Notably, the generalization error is influenced by the distance of the aggregated node features between the test nodes and the training nodes. This implies that, given a fixed training set, test nodes that are "far away" from all the training nodes may suffer from larger generalization errors, which leads to an accuracy-disparity unfairness. In reality, these nodes may reside in small isolated clusters, or they are on the boundaries of large communities. We conduct empirical experiments using multiple benchmark datasets and investigate the test accuracy of four popular GNN models on different subgroups. Results indicate there is indeed a significant disparity in test accuracy among these subgroups.

We summarize the contributions of this work as follows. (1) We establish a novel PAC-Bayesian analysis for graph-based semi-supervised learning with non-IID training nodes. (2) We provide a subgroup generalization bound for GNNs under this setup. (3) As an implication of the derived generalization bound, we predict that there would be an accuracy disparity across different subgroups of test nodes. (4) We empirically verify the existence of accuracy-disparity unfairness of GNNs.

2 Related Work

2.1 Generalization of Graph Neural Networks

The majority of existing literature that aim to develop theoretical understandings of GNNs have focused on the expressive power of GNNs (see Sato [29] for a survey along this line), while the number of studies trying to understand the generalizability of GNNs is rather limited. Among them, some [9, 10, 21] focus on graph-level tasks, the analyses of which cannot be easily applied to node-level tasks. As far as we know, Scarselli et al. [31], Verma and Zhang [36], Baranwal et al. [3] are the only existing studies investigating the generalization of GNNs on node-level tasks, even though node-level tasks are more common in reality. Scarselli et al. [31] present an upper bound of the VC-dimension of GNNs; Verma and Zhang [36] derive a stability-based generalization bound for a single-layer GCN [16] model. Yet, both Scarselli et al. [31] and Verma and Zhang [36] (implicitly) assume that the training nodes are IID samples from a certain distribution, which does not align with the common practice of node-level semi-supervised learning. Baranwal et al. [3] investigate the generalization of graph convolution under a specific data generating mechanism, i.e., the contextual stochastic block model [8]. Our work presents the first generalization analysis of GNNs for non-IID node-level tasks without strong assumptions on the data generating mechanism.

2.2 Fairness of Machine Learning on Graphs

The fairness issues of machine learning on graphs start to receive research attention recently. Following conventional machine learning fairness literature, the majority of previous work along this line [1, 5–7, 18, 27, 33, 43] concerns about fairness with respect to a given sensitive attribute, such as gender or race, which defines protected groups. In practice, the fairness issues of learning on graphs are much more complicated due to the asymmetric nature of the graph-structured data. However, only a few studies [15] investigate the unfairness caused by the graph structure without knowing a sensitive feature. Moreover, in a node-level semi-supervised learning task, the non-IID sampling of training nodes brings additional uncertainty to the fairness of the learned models. This work is the first to present a learning theoretic analysis under this setup, which in turn suggests how the graph structure and the selection of training nodes may influence the fairness of machine learning on graphs.

2.3 PAC-Bayesian Analysis

PAC-Bayesian analysis [22] has become one of the most powerful theoretical framework to analyze the generalization ability of machine learning models. We will briefly introduce the background in Section 3.2, and refer the readers to a recent tutorial [12] for a systematic overview of PAC-Bayesian analysis. We note that Liao et al. [21] recently present a PAC-Bayesian generalization bound for GNNs on IID graph-level tasks. Both Liao et al. [21] and this work utilize results from Neyshabur et al. [25], a PAC-Bayesian analysis for ReLU-activated neural networks, in part of our proofs. Compared to Neyshabur et al. [25], the key contribution of Liao et al. [21] is the derivation of perturbation bounds of two types of GNN architectures; while the key contribution of this work is the novel analysis under the setup of non-IID node-level tasks. There is also an existing work of PAC-Bayesian analysis for transductive semi-supervised learning [4]. But it is different from our problem setup and, in particular, it cannot be used to analyze the generalization on subgroups.

3 Preliminaries

In this section, we first formulate the problem of node-level semi-supervised learning. We also provide a brief introduction of the PAC-Bayesian framework.

3.1 The Problem Formulation and Notations

Semi-supervised node classification. Let $G = (V, E) \in \mathcal{G}_N$ be an undirected graph, with $V = \{1, 2, \ldots, N\}$ being the set of N nodes and $E \subseteq V \times V$ being the set of edges. And $\mathcal{G}_N$ is the space of all undirected graphs with N nodes. The nodes are associated with node features $X \in \mathbb{R}^{N \times D}$ and node labels $y \in \{1, 2, \ldots, K\}^N$.

In this work, we focus on the transductive node classification setting [40], where the node features X and the graph G are observed prior to learning, and every quantity of interest in the analysis will be conditioned on X and G. Without loss of generality, we treat X and G as fixed throughout our analysis and the randomness comes from the labels y. In particular, we assume that for each node $i \in V$, its label y_i is generated from an unknown conditional distribution $\Pr(y_i \mid Z_i)$, where $Z = g(X, G)$ and $g : \mathbb{R}^{N \times D} \times \mathcal{G}_N \to \mathbb{R}^{N \times D'}$ is an aggregation function that aggregates the features over (multi-hop) local neighborhoods[1]. We also assume that the node labels are generated independently conditional on their respective aggregated features Z_i's.

Given a small set of the labeled nodes, $V_0 \subseteq V$, the task of node-level semi-supervised learning is to learn a classifier $h : \mathbb{R}^{N \times D} \times \mathcal{G}_N \to \mathbb{R}^{N \times K}$ from a function family $\mathcal{H}$ and perform it on the remaining unlabeled nodes. Given a classifier h, the classification for a node i is obtained by

$$\hat{y}_i = \underset{k \in \{1, \ldots, K\}}{\operatorname{argmax}} \; h_i(X, G)[k],$$

where $h_i(X, G)$ is the i-th row of the output of $h(X, G)$ and $h_i(X, G)[k]$ refers to the k-th element of $h_i(X, G)$.

Subgroups. In Section 4, we will present an analysis of the GNN generalization performance on any subgroup of the set of unlabeled nodes, $V \setminus V_0$. Note that the analysis on any subgroup is a stronger result than that on the entire unlabeled set, as the entire set is a subset. Later we will show that the analysis on subgroups (rather than on the entire set) further allows us to investigate the accuracy disparity across subgroups. We denote a collection of subgroups of interest as $V_1, V_2, \ldots, V_M \subseteq V \setminus V_0$. In practice, a subgroup can be defined based on an attribute of the nodes (e.g., a gender group), certain graph-based properties, or an arbitrary partition of the nodes. We also define the size of each subgroup as $N_m := |V_m|, m = 0, \ldots, M$.

Margin loss on each subgroup. Now we can define the *empirical* and *expected margin loss* of any classifier $h \in \mathcal{H}$ on each subgroup $V_m, m = 0, 1, \ldots, M$. Given a sample of observed node labels y_i's, the empirical margin loss of h on V_m for a margin $\gamma \geq 0$ is defined as

$$\widehat{\mathcal{L}}_m^{\gamma}(h) := \frac{1}{N_m} \sum_{i \in V_m} \mathbb{1}\left[h_i(X, G)[y_i] \leq \gamma + \max_{k \neq y_i} h_i(X, G)[k] \right], \tag{1}$$

where $\mathbb{1}[\cdot]$ is the indicator function. The expected margin loss is the expectation of Eq. (1), i.e.,

$$\mathcal{L}_m^{\gamma}(h) := \mathbb{E}_{y_i \sim \Pr(y \mid Z_i), i \in V_m} \widehat{\mathcal{L}}_m^{\gamma}(h). \tag{2}$$

To simplify the notation, we define $y^m := \{y_i\}_{i \in V_m}, \forall m = 0, \ldots, M$, so that Eq. (2) can be written as $\mathcal{L}_m^{\gamma}(h) = \mathbb{E}_{y^m} \widehat{\mathcal{L}}_m^{\gamma}(h)$. We note that the classification *risk* and *empirical risk* of h on V_m are respectively equal to $\mathcal{L}_m^0(h)$ and $\widehat{\mathcal{L}}_m^0(h)$.

3.2 The PAC-Bayesian Framework

The PAC-Bayesian framework [22] is an approach to analyze the generalization ability of a stochastic predictor drawn from a distribution Q over the predictor family $\mathcal{H}$ that is learned from the training

[1] An example is $g_i(X, G) = \frac{1}{|\mathcal{N}(i)|+1} \left(X_i + \sum_{j \in \mathcal{N}(i)} X_j \right)$, where $g_i(X, G)$ is the i-th row of the output of $g(X, G)$ and $\mathcal{N}(i) := \{j \mid (i, j) \in E\}$ is the set of 1-hop neighbors of node i. The aggregation function g can also be defined to aggregate over multiple-hop neighbors.

data. For any stochastic classifier distribution Q and $m = 0, \ldots, M$, slightly overloading the notation, we denote the empirical margin loss of Q on V_m as $\widehat{\mathcal{L}}_m^\gamma(Q)$, and the corresponding expected margin loss as $\mathcal{L}_m^\gamma(Q)$. And they are given by

$$\widehat{\mathcal{L}}_m^\gamma(Q) := \mathbb{E}_{h\sim Q}\widehat{\mathcal{L}}_m^\gamma(h), \quad \mathcal{L}_m^\gamma(Q) := \mathbb{E}_{h\sim Q}\mathcal{L}_m^\gamma(h).$$

In general, a PAC-Bayesian analysis aims to bound the generalization gap between $\mathcal{L}_m^\gamma(Q)$ and $\widehat{\mathcal{L}}_m^\gamma(Q)$. The analysis is usually done by first proving that, for any "prior" distribution[2] P over $\mathcal{H}$ that is independent of the training data, the generalization gap can be controlled by the discrepancy between P and Q; the analysis is then followed by careful choices of P to get concrete upper bounds of the generalization gap. While the PAC-Bayesian framework is built on top of stochastic predictors, there exist standard techniques [19] that can be used to derive generalization bounds for deterministic predictors from PAC-Bayesian bounds.

Finally, we introduce two divergence of distributions that will be used in the analysis. We denote the *total variation (TV) divergence* between two distributions Q and P as $D_{\mathrm{TV}}(Q\|P) := \frac{1}{2}\int |\frac{dQ}{dP} - 1| dP$, and the *Kullback-Leibler (KL) divergence* as $D_{\mathrm{KL}}(Q\|P) := \int \ln \frac{dQ}{dP} dQ$.

4 Analysis

As we mentioned in Section 2.3, existing PAC-Bayesian analyses cannot be directly applied to the non-IID semi-supervised learning setup where we care about the generalization (disparity) across different subgroups of the unlabeled samples. In this section, we first present general PAC-Bayesian theorems for subgroup generalization under our problem setup; then we derive a generalization bound for GNNs and discuss implications of the bounds.

4.1 General PAC-Bayesian Theorems for Subgroup Generalization

Stochastic classifier bound. We first present the general PAC-Bayesian theorem (Theorem 1) for subgroup generalization of stochastic classifiers. The generalization bound depends on a notion of *expected loss discrepancy* between two subgroups as defined below.

Definition 1 (Expected Loss Discrepancy)**.** *Given a distribution P over $\mathcal{H}$, for any $\lambda > 0$ and $\gamma \geq 0$, for any two subgroups V_m and $V_{m'}$ ($0 \leq m, m' \leq M$), define the expected loss discrepancy between V_m and V_0 with respect to (P, γ, λ) as*

$$D_{m,m'}^\gamma(P;\lambda) := \ln \mathbb{E}_{h\sim P} e^{\lambda\phi\left(\mathcal{L}_m^{\gamma/2}(h) - \mathcal{L}_{m'}^\gamma(h)\right)},$$

where $\mathcal{L}_m^{\gamma/2}(h)$ and $\mathcal{L}_{m'}^\gamma(h)$ follow the definition of Eq. (2), and we define $\phi(x) := \max(0, x)$.

Intuitively, $D_{m,m'}^\gamma(P;\lambda)$ captures the difference of the expected loss between V_m and $V_{m'}$ in an average sense (over P). Note that $D_{m,m'}^\gamma(P;\lambda)$ is asymmetric in terms of V_m and $V_{m'}$, and can be negative if the loss on V_m is mostly smaller than that on $V_{m'}$.

For stochastic classifiers, we have the following Theorem 1. Proof can be found in Appendix A.1.

Theorem 1 (Subgroup Generalization of Stochastic Classifiers)**.** *For any $0 < m \leq M$, for any $\lambda > 0$ and $\gamma \geq 0$, for any "prior" distribution P on $\mathcal{H}$ that is independent of the training data on V_0, with probability at least $1 - \delta$ over the sample of y^m, for any Q on $\mathcal{H}$, we have*[3]

$$\mathcal{L}_m^{\gamma/2}(Q) \leq \widehat{\mathcal{L}}_0^\gamma(Q) + \frac{1}{\lambda}\left(D_{\mathrm{TV}}(Q\|P) + \ln\frac{2}{\delta} + \frac{\lambda^2}{4N_0} + D_{m,0}^\gamma(P;\lambda)\right). \tag{3}$$

Theorem 1 can be viewed as an adaptation of a result by Alquier et al. [2] from the IID supervised setting to our non-IID semi-supervised setting. The terms $D_{\mathrm{TV}}(Q\|P), \ln\frac{2}{\delta}$, and $\frac{\lambda^2}{4N_0}$ are

[2]The distribution is called "prior" in the sense that it doesn't depend on training data. "Prior" and "posterior" in PAC-Bayesian are different with those in conventional Bayesian statistics. See Guedj [12] for details.

[3]Theorem 1 also holds when we substitute $\mathcal{L}_m^{\gamma/2}(h)$ and $\mathcal{L}_m^{\gamma/2}(Q)$ as $\mathcal{L}_m^\gamma(h)$ and $\mathcal{L}_m^\gamma(Q)$ respectively. But we state the theorem in this form to ease the development of the later analysis.

commonly seen in PAC-Bayesian analysis for IID supervised setting. In particular, when setting $\lambda = \Theta(\sqrt{N_0})$, $\frac{1}{\lambda}\left(\ln\frac{2}{\delta} + \frac{\lambda^2}{4N_0}\right)$ vanishes as the training size N_0 grows. The divergence between Q and P, $D_{\mathrm{TV}}(Q\|P)$, is usually considered as a measurement of the model complexity [12]. And there will be a trade-off between the training loss, $\widehat{\mathcal{L}}_0^{\gamma}(Q)$, and the complexity (how far can the learned "posterior" Q go from the "prior" P).

Uniquely for the non-IID semi-supervised setting, there is an extra term $D_{m,0}^{\gamma}(P;\lambda)$, which is the expected loss discrepancy between the target test subgroup V_m and the training set V_0. Note that this quantity is independent of the training labels y^0. Not surprisingly, it is difficult to give generalization guarantees if the expected loss on V_m is much larger than that on V_0 for any stochastic classifier P independent of training data. We have to make some assumptions about the relationship between V_m and V_0 to obtain a meaningful bound on $\frac{1}{\lambda}D_{m,0}^{\gamma}(P;\lambda)$, which we will discuss in details in Section 4.2.

Deterministic classifier bound. Utilizing standard techniques in PAC-Bayesian analysis [19, 22, 25], we can convert the bound for stochastic classifiers in Theorem 1 to a bound for deterministic classifiers as stated in Theorem 2 below (see Appendix A.2 for the proof).

Theorem 2 (Subgroup Generalization of Deterministic Classifiers)**.** *Let $\tilde{h}$ be any classifier in $\mathcal{H}$. For any $0 < m \leq M$, for any $\lambda > 0$ and $\gamma \geq 0$, for any "prior" distribution P on $\mathcal{H}$ that is independent of the training data on V_0, with probability at least $1-\delta$ over the sample of y^m, with probability at least $1-\delta$ over the sample of y^m, for any Q on $\mathcal{H}$ such that $\Pr_{h\sim Q}\left(\max_{i\in V_0\cup V_m}|h_i(X,G) - \tilde{h}_i(X,G)|_\infty < \frac{\gamma}{8}\right) > \frac{1}{2}$, we have*

$$\mathcal{L}_m^0(\tilde{h}) \leq \widehat{\mathcal{L}}_0^{\gamma}(\tilde{h}) + \frac{1}{\lambda}\left(2\sqrt{D_{\mathrm{KL}}(Q\|P)+1} + \ln\frac{2}{\delta} + \frac{\lambda^2}{4N_0} + D_{m,0}^{\gamma/2}(P;\lambda)\right). \tag{4}$$

Theorem 1 and 2 are not specific to GNNs and hold for any (respectively stochastic and deterministic) classifier under the semi-supervised setup. In Section 4.2, we will apply Theorem 2 to obtain a subgroup generalization bound that explicitly depends on the characteristics of GNNs and the data.

4.2 Subgroup Generalization Bound for Graph Neural Networks

The GNN model. We consider GNNs where the node feature aggregation step and the prediction step are separate. In particular, we assume the GNN classifier takes the form of $h_i(X,G) = f(g_i(X,G);W_1,W_2,\ldots,W_L)$, where g is an aggregation function as we described in Section 3.1 and f is a ReLU-activated L-layer Multi-Layer Perceptron (MLP) with $W_1,\ldots,W_L$ as parameters for each layer[4]. Denote the largest width of all the hidden layers as b.

Upper-bounding $D_{m,0}^{\gamma}(P;\lambda)$. To derive the generalization guarantee, we first upper-bound the expected loss discrepancy $D_{m,0}^{\gamma}(P;\lambda)$ by making two assumptions on the data. We first assume that the label distributions conditional on aggregated features are smooth (Assumption 1).

Assumption 1 (Smoothness of Data Distribution)**.** *Assume there exist c-Lipschitz continuous functions $\eta_1,\eta_2,\ldots,\eta_K : \mathbb{R}^{D'} \to [0,1]$, such that, for any node $i \in V$,*

$$\Pr(y_i = k \mid g_i(X,G)) = \eta_k(g_i(X,G)), \forall k = 1,\ldots,K.$$

We also need to characterize the relationship between a target subgroup V_m and the training set V_0. For this purpose, we define the distance from V_m to V_0 and the concept of *near set* below.

Definition 2 (Distance To Training Set and Near Set)**.** *For each $0 < m \leq M$, define the distance from the subgroup V_m to the training set V_0 as*

$$\epsilon_m := \max_{j\in V_m}\min_{i\in V_0}\|g_i(X,G) - g_j(X,G)\|_2.$$

Further, for each $i \in V_0$, define the near set of i with respect to V_m as

$$V_m^{(i)} := \{j \in V_m \mid \|g_i(X,G) - g_j(X,G)\|_2 \leq \epsilon_m\}.$$

Clearly,

$$V_m = \cup_{i\in V_0} V_m^{(i)}.$$

[4]SGC [38] and APPNP [17] are special cases of GNNs in this form.

Then, with the Assumption 2 below, we can bound the expected loss discrepancy $D^{\gamma}_{m,0}(P;\lambda)$ with the following Lemma 1 (see the proof in Appendix A.3).

Assumption 2 (Equal-Sized and Disjoint Near Sets)**.** *For any $0 < m \leq M$, assume the near sets of each $i \in V_0$ with respect to V_m are disjoint and have the same size $s_m \in \mathbb{N}^+$.*

Lemma 1 (Bound for $D^{\gamma}_{m,0}(P;\lambda)$)**.** *Under Assumption 1 and 2, for any $0 < m \leq M$, any $\lambda > 0$ and $\gamma \geq 0$, assume the prior P on $\mathcal{H}$ is defined by sampling the vectorized MLP parameters from $\mathcal{N}(0, \sigma^2 I)$ for some $\sigma^2 \leq \frac{(\gamma/4\epsilon_m)^{2/L}}{2b(\ln 2bL + \lambda)}$. we have*

$$D^{\gamma}_{m,0}(P;\lambda) \leq \ln 2 + \lambda c K \epsilon_m. \tag{5}$$

Intuitively, what we need to bound $D^{\gamma}_{m,0}(P;\lambda)$ is that the training set V_0 is "representative" for V_m. This is reasonable in practice as it's natural to select the training samples according to the distribution of the population. Specifically, Assumption 2 assumes that V_m can be split into equal-sized partitions indexed by the training samples. The elements of each partition $V_m^{(i)}$ are close to the corresponding training sample i but not so close to training samples other than i. This assumption is stronger than needed to obtain a meaningful bound on $D^{\gamma}_{m,0}(P;\lambda)$, and we can relax it by only assuming that most samples in V_m have proportional "close representatives" in V_0. But we keep Assumption 2 in this work, as it is intuitively clear and significantly eases the analysis and notations.

The bound (5) suggests that the closer between V_m and V_0 (smaller ϵ_m), the smaller the expected loss discrepancy.

Bound for GNNs. Finally, with an additional assumption (Assumption 3) that the maximum L2 norm of aggregated node features does not grow too fast in terms of the number of training samples, we obtain a subgroup generalization bound for GNNs in Theorem 3. The proof of Theorem 3 can be found in Appendix A.4.

Assumption 3. *Define $B_m := \max_{i \in V_0 \cup V_m} \|g_i(X, G)\|_2$, and assume $B_m = o(N_0^{1/4})$.*

Theorem 3 (Subgroup Generalization Bound for GNNs)**.** *Let $\tilde{h}$ be any classifier in $\mathcal{H}$ with parameters $\{\widetilde{W}_l\}_{l=1}^L$. Under Assumptions 1, 2, and 3, for any $0 < m \leq M$, $\gamma \geq 0$ and large enough N_0, with probability at least $1-\delta$ over the sample of y^m, we have*

$$\mathcal{L}^0_m(\tilde{h}) \leq \widehat{\mathcal{L}}^{\gamma}_0(\tilde{h}) + O\left(cK\epsilon_m + \frac{2\sqrt{b\sum_{l=1}^L \|\widetilde{W}_l\|_F^2}}{N_0^{1/4}(\gamma/8)^{1/L}} (\epsilon_m)^{1/L} + \frac{1}{N_0^{1/2}} \ln \frac{N_0}{\delta} \right). \tag{6}$$

Next, we investigate the qualitative implications of our theoretical results.

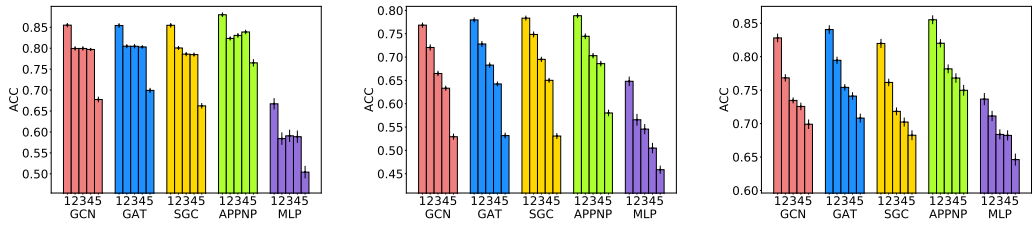
4.3 Implications for Fairness of Graph Neural Networks

Accuracy-disparity style of unfairness. One merit of our analysis is that we can apply Theorem 3 on different subgroups of the unlabeled nodes and compare the subgroup generalization bounds. This allows us to study the accuracy disparity across subgroups from a theoretical perspective.

A major factor that affects the generalization bound (6) is ϵ_m, the distance from the target subgroup V_m to the training set V_0. The generalization bound (6) suggests that there is a better generalization guarantee for subgroups that are closer to the training set. In other words, it is unfair for subgroups that are far away from the training set. While our theoretical analysis only provides an upper bound for the generalization error, in Section 5, we empirically verify that the test performances of GNN models do present accuracy disparity across subgroups with varying distances to the training set.

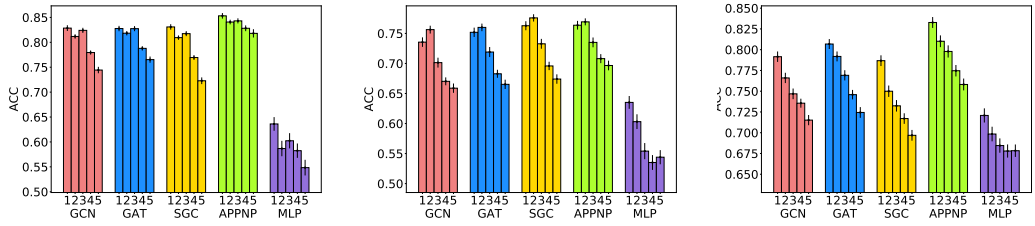
Moreover, when more domain knowledge about the particular learning task and data is available, we can further investigate the factors that affect ϵ_m and identify potential fairness issues. As an example, the geodesic distance (length of shortest-path on the graph) between two nodes may be a good indicator for the similarity of their aggregated features. Below we discuss two such scenarios.

Smoothing effect of feature aggregation in GNNs. Many existing GNN models are known to have a smoothing effect on the aggregated node features [20]. As a result, nodes with a shorter geodesic distance are likely to have more similar aggregated features.



(a) Cora. (b) Citeseer. (c) Pubmed.

Figure 1: Test accuracy disparity across subgroups by aggregated-feature distance. Each figure corresponds to a dataset, and each bar cluster corresponds to a model. Bars labeled 1 to 5 represent subgroups with increasing distance to training set. Results are averaged over 40 independent trials with different random splits of the data, and the error bar represents the standard error of the mean.



(a) Cora. (b) Citeseer. (c) Pubmed.

Figure 2: Test accuracy disparity across subgroups by geodesic distance. The experiment and plot settings are the same as Figure 1, except for the bars labeled from 1 to 5 here represent subgroups with increasing shortest-path distance to training set.

Homophily. Many real-world graph-structured data present a homophily property [23], i.e., connected nodes tend to share similar attributes. In this case, again, nodes with a shorter distance on the graph tend to have more similar aggregated features.

Impact of training data selection. Another implication of the theoretical results is that the selection of the training set plays an important role on the fairness of the learned GNN models. First, at a population level, if the training set of choice leaves part of the unlabeled set far away, there will likely be a large accuracy disparity. Second, a key ingredient in the proof of Lemma 1 is that the predictions of the model on two nodes are more likely to be the same if they are close in terms of the aggregated node features. This suggests that, when the shortest-path distance is a good indicator for the similarity of the aggregated features, training nodes with higher *closeness centrality*[5] may have a higher impact on the behaviour of the learned model. More generally, the influence of training nodes on the learned model may be relevant to their positions on the graph.

5 Experiments

In this section, we empirically verify the accuracy disparity suggested by our theoretical results.

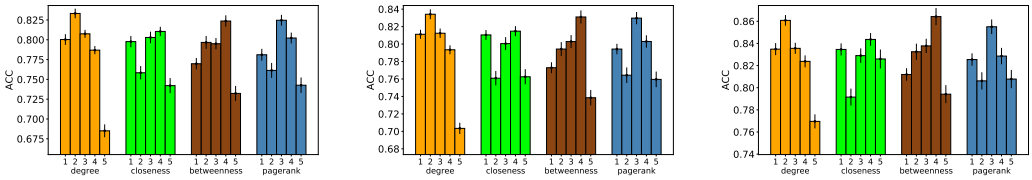
General setup. We experiment on 4 popular GNN models, GCN [16], GAT [35], SGC [38], and APPNP [17], as well as a MLP model for reference. For all the models, we use the implementations in Deep Graph Library [37]. 40 independent trails are carried out for each experiment.

5.1 Accuracy Disparity Across Subgroups

Subgroups. We examine the accuracy disparity with *three types of* subgroups as described below.

Subgroup by aggregated-feature distance. In order to directly investigate the effect of ϵ_m on the generalization bound (6), we first split the test nodes into subgroups by their distance to the training set in terms of the aggregated features. As the GCN and GAT models are all two-layer GNNs, we use the two-step aggregated features to calculate the distance. In particular, denote the adjacency matrix of the graph G as $A \in \{0,1\}^{N\times N}$ and the corresponding degree matrix as D, where D is a $N \times N$ diagonal

[5]Closeness centrality of node i is defined as $1/\sum_{j\in V\setminus\{i\}} d(i,j)$, where $d(\cdot,\cdot)$ is the shortest-path distance.



(a) GCN on Cora. (b) GAT on Cora. (c) APPNP on Cora.

Figure 3: Test accuracy disparity across subgroups by node centrality. Each figure corresponds to the results of a pair of model and dataset, and each bar cluster corresponds to the subgroups defined by a certain centrality metric. In each cluster, the bars labeled from 1 to 5 represent subgroups with decreasing node centrality. Other settings are the same as Figure 1.

matrix with $D_{ii} = \sum_{j=1}^{N} A_{ij}, \forall i = 1, \ldots, N$. Given the feature matrix $X \in \mathbb{R}^{N \times D}$, The two-step aggregated features are obtained by $Z = (D+I)^{-1}(A+I)(D+I)^{-1}(A+I)X$. For each test node i, we define its aggregated-feature distance to the training set V_0 as $d_i = \min_{j \in V_0} \|Z_i - Z_j\|_2$. Then we sort the test nodes according to this distance and split them into 5 equal-sized subgroups.

Subgroup by geodesic distance. As we discussed in Section 4.3, the geodesic distance on the graph may correlate with the aggregated-feature distance. So we also define subgroups based on the geodesic distance. We split the subgroups similarly by replacing S_i of each test node i as the minimum of the geodesic distances from i to each training node on the graph.

Subgroup by node centrality. Lastly, we also define subgroups based on 4 types of node centrality scores (degree, closeness, betweenness, and PageRank) of the test nodes. We split the subgroups by replacing S_i of each test node i as the centrality score of i. The purpose of this setup is to rule out a potential confounding factor that test nodes close to the training set may have high centrality scores.

Experiment setup. Following common GNN experiment setup [32], we randomly select 20 nodes in each class for training, 500 nodes for validation, and 1,000 nodes for testing. Once training is done, we report the test accuracy on subgroups defined by aggregated-feature distance, geodesic distance, and node centrality in Figure 1, 2, and 3 respectively[6].

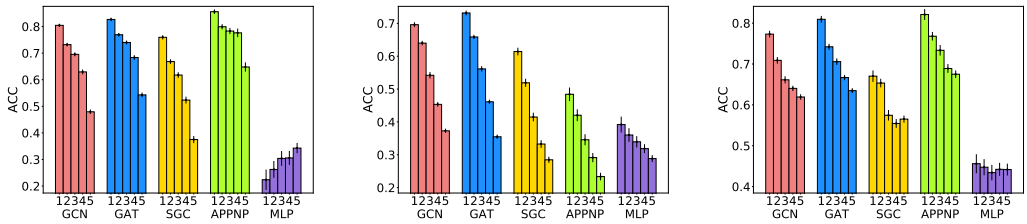
Experiment results. First, as shown in Figure 1, there is a clear trend that the accuracy of a test subgroup decreases as the aggregated-feature distance between the test subgroup and the training set increases. And the trend is consistent for all 4 GNN models on all the datasets we test on (except for APPNP on Cora). This result verifies the existence of accuracy disparity suggested by Theorem 3.

Second, we observe in Figure 2 that there is a similar trend when we split subgroups by the geodesic distance. This suggests that the geodesic distance on the graph can be used as a simpler indicator in practice for machine learning fairness on real-world graph-structured data. Using such a classical network metric as an indicator also helps us connect graph-based machine learning to network theory, especially to understandings about social networks, to better analyze fairness issues of machine learning on social networks, where high-stake decisions related to human subjects may be involved.

Furthermore, as in Figure 3, there is no monotonic trend for test accuracy when we split subgroups by node centrality. This suggests that it is indeed the distance between the test subgroup and the training nodes, rather than the centrality of the test nodes alone, that influences the generalization error.

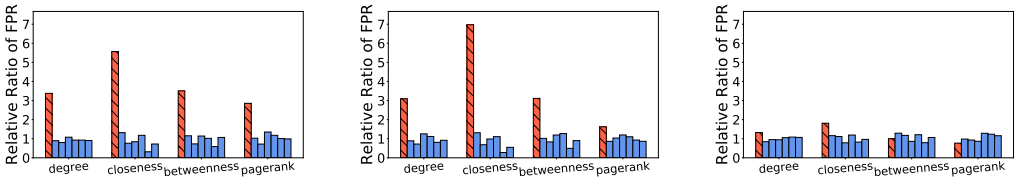
Finally, it is intriguing that, in both Figure 1 and 2, the test accuracy of MLP (which does not use the graph structure) also decreases as the distance of a subgroup to the training set increases. This result is perhaps not surprising if the subgroups were defined by distance on the original node features, as MLP can be viewed as a special GNN where the feature aggregation function is an identity mapping, so the "aggregated features" for MLP essentially equal to the original features. Our theoretical analysis can then be similarly applied to MLP. The question is why there is also an accuracy disparity w.r.t. the aggregated-feature distance and the geodesic distance. We suspect this is because these datasets present homophily, i.e., original (non-aggregated) features of geodesically closer nodes tend to be more similar. As a result, a subgroup with smaller geodesic distance may also have closer node features to the training set. To verify this hypothesis, we repeat the experiments in Figure 1, but with

[6]The main paper reports the results on selected datasets (Cora, Citeseer, and Pubmed for subgroups by aggregated-feature & geodesic distance, and Cora for node centrality. Results on more datasets are in Appendix C.



(a) Cora. (b) Citeseer. (c) Pubmed.

Figure 4: Test accuracy disparity across subgroups by aggregated-feature distance, experimented with noisy features. The experiment and plot settings are the same as Figure 1, except for the node features are perturbed by independent noises such that they are less homophilious.



(a) GCN on Cora. (b) GAT on Cora. (c) MLP on Cora.

Figure 5: Relative ratio between the FPR under biased training node selection and the FPR under uniform training node selection. Each bar in each cluster corresponds to a class (there are 7 classes in total). The red shaded bar indicates the class with high centrality training nodes under the biased setup. Each cluster corresponds to a centrality metric being used for the biased node selection.

independent noises added to node features such that they become less homophilious. As in Figure 4, the decreasing pattern of test accuracy across subgroups remains for the 4 GNNs on all datasets; while for MLP, the pattern disappears on Cora and Pubmed and becomes less sharp on Citeseer.

5.2 Impact of Biased Training Node Selection

In all the previous experiments, we follow the standard GNN training setup where 20 training nodes are uniformly sampled for each class. Next we investigate the impact if the selection of training nodes is biased, verifying our discussions in Section 4.3. We will demonstrate that the node centrality scores of the training nodes play an important role in the learned GNN model.

We choose a "dominant class" and construct a manipulated training set. For each class, we still sample 20 training nodes but in a biased way. For the dominant class, the sample is biased towards nodes of high centrality; while for other classes, the sample is biased towards nodes of low centrality. We evaluate the relative ratio of False Positive Rate (FPR) for each class between the setup using manipulated training set and the setup using uniformly sampled training set.

As shown in Figure 5, compared to MLP, the GNN models have significantly worse FPR for the dominant class when the training nodes are biased. This is because, after feature aggregation, there will be a larger proportion of test nodes that are closer to the training nodes of higher centrality. And the learned GNN model will be heavily biased towards the training labels of these nodes.

6 Discussion and Conclusion

We present a novel PAC-Bayesian analysis for the generalization ability of GNNs on node-level semi-supervised learning tasks. As far as we know, this is the first generalization bound for GNNs for non-IID node-level tasks without strong assumptions on the data generating mechanism. One advantage of our analysis is that it can be applied to arbitrary subgroups of the test nodes, which allows us to investigate an accuracy-disparity style of fairness for GNNs. Both the theoretical and empirical results suggest that there is an accuracy disparity across subgroups of test nodes that have varying distance to the training set, and nodes with larger geodesic distance to the training nodes suffer from a lower classification accuracy. In reality, these nodes are likely to reside in underrepresented marginalized communities or on the boundaries of large communities. In the future, we would like to utilize our theoretical results to analyze other potential factors of the fairness of GNNs.

References

[1] Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. Towards a unified framework for fair and stable graph representation learning. *CoRR*, abs/2102.13186, 2021. URL `https://arxiv.org/abs/2102.13186`.

[2] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of gibbs posteriors. *The Journal of Machine Learning Research*, 17(1):8374–8414, 2016.

[3] Aseem Baranwal, Kimon Fountoulakis, and Aukosh Jagannath. Graph convolution for semi-supervised classification: Improved linear separability and out-of-distribution generalization. *arXiv preprint arXiv:2102.06966*, 2021.

[4] Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. Pac-bayesian theory for transductive learning. In *Artificial Intelligence and Statistics*, pages 105–113. PMLR, 2014.

[5] Avishek Joey Bose and William L. Hamilton. Compositional fairness constraints for graph embeddings. *CoRR*, abs/1905.10674, 2019. URL `http://arxiv.org/abs/1905.10674`.

[6] Maarten Buyl and Tijl De Bie. The kl-divergence between a graph model and its fair i-projection as a fairness regularizer. *CoRR*, abs/2103.01846, 2021. URL `https://arxiv.org/abs/2103.01846`.

[7] Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 680–688, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382977. doi: 10.1145/3437963.3441752. URL `https://doi.org/10.1145/3437963.3441752`.

[8] Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual stochastic block models. In *NeurIPS*, 2018.

[9] Simon S Du, Kangcheng Hou, Barnabás Póczos, Ruslan Salakhutdinov, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. *arXiv preprint arXiv:1905.13192*, 2019.

[10] Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*, pages 3419–3430. PMLR, 2020.

[11] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[12] Benjamin Guedj. A primer on pac-bayesian learning. *arXiv preprint arXiv:1901.05353*, 2019.

[13] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

[14] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pages 2323–2332. PMLR, 2018.

[15] Ahmad Khajehnejad, Moein Khajehnejad, Mahmoudreza Babaei, Krishna P. Gummadi, Adrian Weller, and Baharan Mirzasoleiman. Crosswalk: Fairness-enhanced node representation learning, 2021.

[16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[17] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2019.

[18] Charlotte Laclau, Ievgen Redko, Manvi Choudhary, and Christine Largeron. All of the fairness for edge prediction with optimal transport. *CoRR*, abs/2010.16326, 2020. URL `https://arxiv.org/abs/2010.16326`.

[19] John Langford and John Shawe-Taylor. Pac-bayes & margins. *Advances in neural information processing systems*, pages 439–446, 2003.

[20] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[21] Renjie Liao, Raquel Urtasun, and Richard Zemel. A pac-bayesian approach to generalization bounds for graph neural networks. In *ICLR*, 2021.

[22] David McAllester. Simplified pac-bayesian margin bounds. In *Learning theory and Kernel machines*, pages 203–215. Springer, 2003.

[23] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

[24] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5115–5124, 2017.

[25] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. 2018.

[26] Yuki Ohnishi and Jean Honorio. Novel change of measure inequalities with applications to pac-bayesian bounds and monte carlo estimation. In *International Conference on Artificial Intelligence and Statistics*, pages 1711–1719. PMLR, 2021.

[27] Tahleen Rahman, Bartlomiej Surma, Michael Backes, and Yang Zhang. Fairwalk: Towards fair graph embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3289–3295. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/456. URL `https://doi.org/10.24963/ijcai.2019/456`.

[28] Omar Rivasplata. Subgaussian random variables: An expository note. *Internet publication, PDF*, 2012.

[29] Ryoma Sato. A survey on the expressive power of graph neural networks. *arXiv preprint arXiv:2003.04078*, 2020.

[30] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[31] Franco Scarselli, Ah Chung Tsoi, and Markus Hagenbuchner. The vapnik–chervonenkis dimension of graph and recursive neural networks. *Neural Networks*, 108:248–259, 2018.

[32] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *CoRR*, abs/1811.05868, 2018. URL `http://arxiv.org/abs/1811.05868`.

[33] Indro Spinelli, Simone Scardapane, Amir Hussain, and Aurelio Uncini. Biased edge dropout for enhancing fairness in graph representation learning, 2021.

[34] Joel A Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[36] Saurabh Verma and Zhi-Li Zhang. Stability and generalization of graph convolutional neural networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1539–1548, 2019.

[37] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

[38] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[39] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 2020.

[40] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016.

[41] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983, 2018.

[42] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3634–3640, 2018.

[43] Ziqian Zeng, Rashidul Islam, Kamrun Naher Keya, James R. Foulds, Yangqiu Song, and Shimei Pan. Fair representation learning for heterogeneous information networks. *CoRR*, abs/2104.08769, 2021. URL `https://arxiv.org/abs/2104.08769`.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] Proofs are provided in the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] In the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] The experiments in this study are not computationally expensive and the resources are less of interest.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We credited the use of Deep Graph Library as well as the benchmark datasets in our experiments.
   (b) Did you mention the license of the assets? [No] Both the Deep Graph Library and benchmark datasets are commonly used in the community.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We included our code in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We are using widely-used benchmark datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The benchmark datasets are already anonymized and widely used by the research community.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]