Debiasing Graph Neural Networks via Learning Disentangled Causal Substructure

Shaohua Fan[1,2]*, Xiao Wang[1], Yanhu Mo[1], Chuan Shi[1]†, Jian Tang[2,3,4]†
[1]Beijing University of Posts and Telecommunications, China
[2] Mila - Québec AI Institute, Canada
[3] HEC Montréal, Canada
[4] CIFAR AI Research Chair
{fanshaohua, xiaowang, moyanhu, shichuan}@bupt.edu.cn, jian.tang@hec.ca

Abstract

Most Graph Neural Networks (GNNs) predict the labels of unseen graphs by learning the correlation between the input graphs and labels. However, by presenting a graph classification investigation on the training graphs with severe bias, surprisingly, we discover that GNNs always tend to explore the spurious correlations to make decision, even if the causal correlation always exists. This implies that existing GNNs trained on such biased datasets will suffer from poor generalization capability. By analyzing this problem in a causal view, we find that disentangling and decorrelating the causal and bias latent variables from the biased graphs are both crucial for debiasing. Inspiring by this, we propose a general disentangled GNN framework to learn the causal substructure and bias substructure, respectively. Particularly, we design a parameterized edge mask generator to explicitly split the input graph into causal and bias subgraphs. Then two GNN modules supervised by causal/bias-aware loss functions respectively are trained to encode causal and bias subgraphs into their corresponding representations. With the disentangled representations, we synthesize the counterfactual unbiased training samples to further decorrelate causal and bias variables. Moreover, to better benchmark the severe bias problem, we construct three new graph datasets, which have controllable bias degrees and are easier to visualize and explain. Experimental results well demonstrate that our approach achieves superior generalization performance over existing baselines. Furthermore, owing to the learned edge mask, the proposed model has appealing interpretability and transferability.[3]

1 Introduction

Graph Neural Networks (GNNs) have exhibited powerful performance on graph data with various applications [19, 38, 14, 10, 9]. One major category of applications are the graph classification task, such as molecular graph property prediction [16, 22, 48], superpixel graph classification [15], and social network category classification [50, 48]. It is well known that graph classification is usually determined by a relevant substructure, but not the whole graph structure [47, 28, 49]. For example, for MNIST superpixel graph classification task, the digit subgraphs are causal ($i.e.$, deterministic) for labels [39]. The mutagenic property of a molecular graph depends on the functional groups ($i.e.$,

*This work was done when the first author was a visiting student at Mila.

†Corresponding authors.

[3]Code and data are available at: https://github.com/googlebaba/DisC.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

nitrogen dioxide (NO_2)), rather than the irrelevant patterns ($i.e.$, carbon rings) [29]. Therefore, it is a fundamental requirement for GNNs to identify causal substructures, so as to make correct prediction.

Ideally, when the graphs are unbiased, $i.e.$, only the causal substructures are related with the graph labels, the GNNs are able to utilize such substructure to predict the labels. However, due to the uncontrollable data collection process, the graphs are inevitably biased, $i.e.$, existing meaningless substructures spuriously correlates with labels. Taking a colored MNIST superpixel graph dataset in Sec. 3.1 as an example (illustrated in Fig. 1(a)), each category of digit subgraphs mainly correspond to one kind of color background subgraphs, $e.g.$, digit 0 subgraph is related with red background subgraph. Therefore, the color background subgraph will be treated as bias information, which highly correlates with labels but does not determines them in the training set. Under this situation, *will GNNs still stably utilize the causal substructure to make decision?*

To investigate the impact of bias on GNNs, we conduct an experimental investigation to demonstrate the impact of bias (especially in the severe bias scenarios) on the generalization capability of GNNs (Sec. 3.1). We find that GNNs actually utilize both bias and causal substructures to make prediction. However, with severer bias correlation, even bias substructure still could not exactly determine labels like causal substructure, GNNs majorly utilize bias substructure as shortcuts to make prediction, causing a large generalization performance degradation. Why this happens? We analyze the data-generating process and model prediction mechanism behind the graph classification using a causal graph (Sec. 3.2). The casual graph illustrates that the observed graphs are generated by the causal and bias latent variables and existing GNNs could not distinguish the causal substructure from entangled graphs. *How can we disentangle the causal and bias substructures from observed graphs, so that GNNs can only utilize the causal substructures to make stable prediction when severe bias appears?*

To address the question, two challenges need to be faced. 1) How to identify the causal substructure and bias substructure in the severe biased graphs? In the severe bias scenarios, bias substructure will be "easier to learn" for GNNs and finally dominate the prediction. Using the normal cross-entropy loss, like DIR [43], could not fully capture such aggressive property of bias. 2) How to extract the causal substructure from an entangled graph? The statistically causal substructure is usually determined by the global property of the entire graph population, rather than a single graph. When extracting causal substructure from a graph, we need to establish the relations among all the graphs.

In this paper, we propose a novel debiasing framework for GNNs via learning **Dis**entangled **C**ausal substructure, called **DisC**. Given an input biased graph, we propose to explicitly filter edges into causal and bias subgraphs by a parameterized edge mask generator, whose parameters are shared across entire graph population. As a result, the edge masker is naturally capable to specify the importance for each edge and extract causal and bias subgraphs from a global view of the entire observations. Then, a "casual"-aware (weighted cross-entropy) loss and a "bias"-aware (generalized cross-entropy) loss are respectively utilized to supervise two functional GNN modules. Based on the supervision, the edge mask generator could generate corresponding subgraphs and the GNNs could encode corresponding subgraphs into their disentangled embeddings. With the disentangled embeddings, we randomly permute the latent vectors extracted from different graphs to generate more unbiased counterfactual samples in embedding space. The new generated samples still contain both causal and bias information, while their correlation has been decorrelated. In this time, there is only correlation between causal variables with labels, so that the model could concentrate on the true correlation between the causal subgraphs and labels. Our major contributions are as follows:

- To our knowledge, we first study the generalization problem of GNNs in a more challenging yet practical scenario, i.e., the graphs are with severe bias. We systematically analyze the bias impact on GNNs from both experimental study and causal analysis. We find that the bias substructure, compared with causal substructure, is much easier to dominate the training of GNNs.

- To debias GNNs, we develop a novel GNN framework for disentangling causal substructure, which is flexible to build upon various GNNs for improving generalization ability while enjoying inherent interpretability, robustness and transferability.

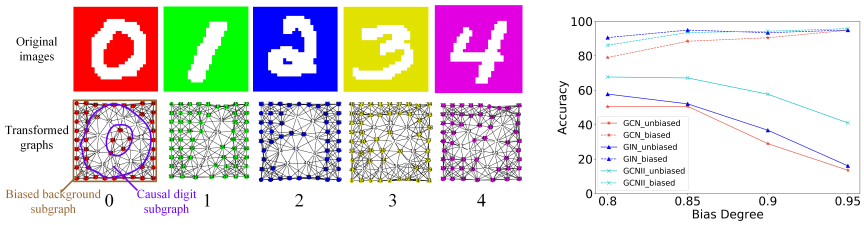
- We construct three new datasets with various properties and controllable bias degrees, which can better benchmark the new problem. Our model outperforms the corresponding base models with a large margin (from 4.47% to 169.17% average improvements). Various investigation studies demonstrate that our model could discover and leverage causal substructure for prediction.

2 Related Works

Generalization of GNNs in wild environments. Most existing GNN methods are proposed under the IID hypothesis, $i.e.$, training and testing set are independently sampled from the identical distribution [36, 19, 38, 14, 26]. However, in reality, thus ideal assumption is hard to be satisfied. Recently, several methods have been proposed to improve the generalization ability of GNNs in wild OOD environments. Several works [31, 8, 42] study the OOD problem of node classification. For OOD graph classification task, StableGNN [7] propose to learn the stable causal relationship in graphs. OOD-GNN [24] propose to constrain each dimension of learned embedding independent. DIR [43] discovers the invariant rationales for generalizing GNNs. Although they have achieved better OOD performance, they are not designed for the datasets with severe bias, which is more challenging for guaranteeing the generalization ability of GNNs.

Disentangled graph neural networks. Recently, there are a couple of methods that study the disentangled GNNs. DisenGCN [30] utilizes neighbourhood routing mechanism to divide the neighbours of the node into several mutually exclusive parts. IPGDN [27] promotes DisenGCN by constraining the different parts of the embedding feature independent. DisenGCN and IPGDN are node-level disentanglement, thus FactorGCN [46] considers the whole graph information and disentangles the target graph into several factorized graphs. Despite results of the previous works, they do not consider disentangling the causal and bias information for graphs.

General debiasing methods. Recently, debiasing problem has drawn much attention in machine learning community [17, 25, 35, 1, 2, 12]. One category of these methods is pre-defining a certain bias type explicitly to mitigate [17, 25, 35, 1, 40]. For example, Wang et al. [40] and Bahng et al. [1] design a texture- and color-guided model to adversarially train a debiased neural network against the biased one. Instead of defining certain types of bias, recent approaches [32, 6, 23] rely on the straightforward assumption that models are prone to exploit the bias as shortcuts to make prediction [11]. In the line with the recent studies, our study belongs to the second category. However, most of existing methods are designed for image datasets and could not effectively extract causal substructure from graph data. Distinctly, we first study the severe bias problem on graph data, and our method could effectively extract causal substructure from graph data.



(a) Examples of graphs in CMNIST-75sp. (b) Performance of GNNs.

Figure 1: Example graphs of CMNIST-75sp and the performance of GNNs on this dataset.

3 Preliminary Study and Analysis

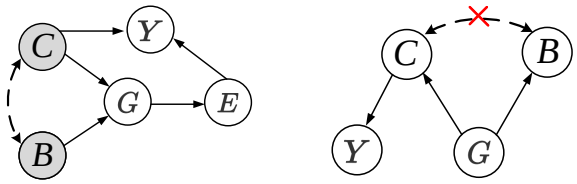
In this section, we first illustrate the existing GNNs tend to exploit the bias substructure as shortcuts for prediction through a motivating experiment. Then we analyze the prediction process of GNNs in causal view. Based on this causal view, it motivates our solution to relieve the impact of bias.

3.1 Motivating Example

To measure the generalization ability of GNNs with the effect of bias, we construct a graph classification dataset with controllable bias degrees, called CMNIST-75sp. We first construct a biased MNIST image dataset like [1], where each category of digit highly correlates with a pre-defined color in their background. For example, in the training set, 90% of 0 digits are with red background ($i.e.$, biased samples), and remaining 10% images are with random background color ($i.e.$, unbiased samples), whose the bias degree is 0.9 in this situation. We consider four bias degrees $\{0.8, 0.85, 0.9, 0.95\}$.

For the testing set, we construct both biased testing set and unbiased testing set. The biased testing set has the same bias degree with training set, aiming to measure the extent of models relying on bias. The unbiased testing set, where the digit labels uncorrelate with the background colors, aims to test whether the model could utilize the inherent digit signals for prediction. Note that training set and testing set have the same pre-defined color set. Then, we convert the biased MNIST images into superpixel graphs with at most 75 nodes each graph using [20], where the edges are constructed by the KNN method based on the 2D coordinates of superpixels and node features are the concatenation of coordinates and average color of superpixels. Each graph is labeled by its digit class, so that its digital subgraph is deterministic for label and background subgraph is spuriously correlated with labels but not deterministic. The examples of graphs are illustrated in Fig. 1(a).

We perform three popular GNN methods: GCN [19], GIN [45], and GCNII [3] on CMNIST-75sp and the results are shown in Fig. 1(b). The same color of dashed line and solid line represent the results of the corresponding methods on the biased testing set and the unbiased testing set respectively. Overall, the GNNs achieve much better performance on biased testing set than unbiased testing set. The phenomenon indicates that although GNNs could still learn some causal signals for prediction, the unexpected bias information is also being utilized for prediction. More specifically, with bias degree becoming larger, the performance of GNNs on biased testing set is increased and the value of accuracy is nearly in line with the bias degree, while the performance on unbiased testing drops dramatically. Hence, although causal substructure could determine labels perfectly, in severe bias scenarios, the GNNs lean to utilize the easier to learn bias information to make prediction rather than the inherent causal signals, and bias substructure will finally dominate the prediction.



(a) SCM of the union of the data generation and the existing GNNs' prediction process.

(b) SCM of our debiasing method.

Figure 2: SCMs. Grey and white variables represent unobserved and observed variables, respectively.

3.2 Problem Analysis

Debiasing GNNs for unbiased prediction requires understanding the natural mechanisms of graph classification task. We present a causal view of the union of the data-generating process and the model prediction process behind the task. Here we formalize the causal view as a Structure Causal Model (SCM) or causal graph [13, 33] by inspecting on the causalities among five variables: unobserved causal variable C, unobserved bias variable B, observed graph G, graph embedding E, and ground truth label / prediction Y[4]. Fig. 2(a) illustrates the SCM, where each link denotes a causal relationship.

- $C \rightarrow G \leftarrow B$. The observed graph data is generated by two unobserved latent variables: the causal variable C and the bias variable B, such as digit subgraphs and background subgraphs in the CMNIST-75sp dataset. And all bellow relations are illustrated by CMNIST-75sp.
- $C \rightarrow Y$. This link means that the causal variable C is the only endogenous parent to determine the generation of ground-truth label Y. For example, C is the oracle digit subgraph, which exactly explains why the label is labeled as Y.
- $C \dashleftarrow\dashrightarrow B$. This link indicates the spurious correlation between C and B. Such probabilistic dependencies is usually caused by the direct cause or unobserved confounder [34]. Here we do not distinguish these scenarios and only observe the spurious correlation between B and C, such as the spurious correlation between the color background subgraphs and digit subgraphs.

[4]We use variable Y for both the ground-truth labels and prediction, as they are optimized to be the same.

- $G \rightarrow E \rightarrow Y$. Existing GNNs usually learn the graph embedding E based on the observed graph G and make the prediction Y based on the learned embedding E.

According to the SCM, GNNs will utilize both information to make prediction. As bias substructure ($e.g.$, background subgraph) usually has simpler structure than meaningful causal substructure ($e.g.$, digit subgraph), if GNN utilizes such simple substructure, it could achieve low loss very fast. Hence, GNN inclines to utilizes bias information when most graphs are biased. Based on the SCM in Fig. 2(a), according to d-connection theory [33] (see App. A): two variables are dependent if they are connected by at least one unblocked path, we could find two paths that would induce the spurious correlation between the bias variable B and label Y: **(1)** $\mathbf{B} \rightarrow \mathbf{G} \rightarrow \mathbf{E} \rightarrow \mathbf{Y}$ and **(2)** $\mathbf{B} \leftrightarrow \mathbf{C} \rightarrow \mathbf{Y}$. To make the prediction Y being uncorrelated with the bias B, we need to intercept the two unblocked paths. For this purpose, we propose to debias GNNs in causal view, as in Fig. 2(b).

- $C \leftarrow G \rightarrow B$ and $C \rightarrow Y$. To intercept the path (1), we should disentangle the latent variables C and B from the observed graph G and make prediction only based on the causal variable C.
- $C \leftarrow \cancel{\rightarrow\leftarrow} \rightarrow B$. To intercept the path (2), as we cannot change the link between C and Y, one possible solution is to make C and B uncorrelated.

4 Methodology

Motivated by the above causal analysis, in this section, we present our proposed debiasing GNN framework **DisC**, to remove the spurious correlation. The overall framework is shown in Fig. 3. First, an edge mask generator is learnt to mask the edges of original input graphs into causal subgraphs and bias subgraphs. Second, two separate GNN modules with their corresponding masked subgraphs are trained to encode corresponding causal substructure and bias substructure into disentangled representations, respectively. Last, after the disentangled representations are well-trained, we permute the bias representations among the training graphs to generate counterfactual unbiased samples, so that the correlation between causal representations and bias representations is removed.

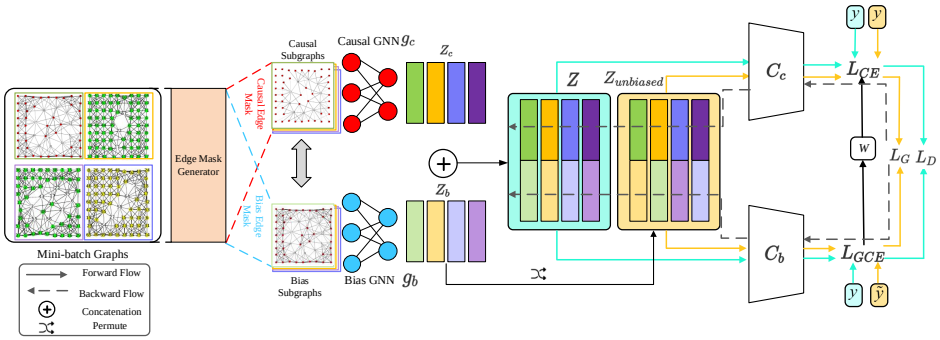


Figure 3: The overall framework of DisC.

4.1 Causal and Bias Substructure Generator

Given a mini-batch of biased graphs $\mathcal{G} = \{G_1, \cdots, G_n\}$, our idea is that: we take a collection of graph instances and design a generative probabilistic model to learn to mask the edges into causal subgraph or bias subgraph. Particularly, given a graph $G = \{\mathbf{A}, \mathbf{X}\}$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{X}$ is the node feature matrix, we utilize a multi-layer perceptron (MLP) upon the concatenation of node features $\mathbf{x}_i$ of node i and $\mathbf{x}_j$ of node j to measure the importance of edge (i, j) for causal subgraph:

$$\alpha_{ij} = \text{MLP}([\mathbf{x}_i, \mathbf{x}_j]). \tag{1}$$

Then a sigmoid function $\sigma(\cdot)$ is employed to project α_{ij} into the range of (0,1), which indicates the probability of edge (i, j) being the edge in the causal subgraph as follows:

$$c_{ij} = \sigma(\alpha_{ij}). \tag{2}$$

Naturally, we could get the probability of edge (i, j) being the edge in the bias subgraph by: $b_{ij} = 1 - c_{ij}$. Now we could construct the causal edge mask $\mathbf{M}_c = [c_{ij}]$ and bias edge mask $\mathbf{M}_b = [b_{ij}]$. Finally, we decompose the original graph G into causal subgraph $G_c = \{\mathbf{M}_c \odot \mathbf{A}, \mathbf{X}\}$ and bias subgraph $G_b = \{\mathbf{M}_b \odot \mathbf{A}, \mathbf{X}\}$. Intuitively, the edge mask could highlight different part of structure information of original graphs, thus GNNs built on the different subgraphs could encode different parts of graph information. Moreover, the mask generator has two advantages. (1) **Global view:** In individual graph level, the mask generator ($i.e.$, MLP), whose parameters are shared by all the edges in a graph, take a global view of all the edges in a graph, which enables us to identify community in graph. It is well known that the effect of an edge cannot be judged independently, because edges usually collaborate with each other, forming a community, to make prediction. Thus, it is critical to evaluate an edge in a global view. In whole graph population level, the mask generator takes a global view of all the graphs in the training set, which enables us to identify causal/bias subgraph. Particularly, as the causal/bias is the statistical information in the population level, it is necessary to view all the graphs to identify the causal/bias substructure. Considering both such coalition effects and population-level statistical information, the generator is able to measure the importance of edges more accurately. (2) **Generalization**: The mask generator can generalize the mechanism of mask generation to new graphs without retraining, so it is capable and efficient to prune unseen graphs.

4.2 Learning Disentangled Graph Representations

Given G_c and G_b, how to ensure they are causal subgraph and bias subgraph, respectively? Inspired by [23], our approach simultaneously trains a pair of GNNs (g_b, g_c) with linear classifiers (C_b, C_c) as follows: (1) Motivated by the observation in Sec. 3.1 that bias substructure is easier to learn, we utilize a bias-aware loss to train a bias GNN g_b and a bias classifier C_b and (2) in contrast, we train a causal GNN g_c and a causal classifier C_c on the training graphs that the bias GNN struggles to learn. Next, we would present each component in detail.

As shown in Fig. 3, GNN g_c and g_b embed the corresponding subgraphs into causal embedding $z_c = g_c(G_c; \gamma_c)$ and bias embedding $z_b = g_b(G_b; \gamma_b)$, respectively, where γ is the parameters of GNNs. Subsequently, concatenated vector $z = [z_c; z_b]$ is fed into linear classifiers C_c and C_b to predict the target label y. To train g_b and C_b as bias extractor, we utilize the generalized cross entropy (GCE) [51] loss to amplify the bias of the bias GNN and classifier:

$$GCE(C_b(z; \alpha_b), y) = \frac{1 - C_b^y(z; \alpha_b)^q}{q}, \tag{3}$$

where $C_b(z; \alpha_b)$ and $C_b^y(z; \alpha_b)$ are softmax output of the bias classifier and its probability belonging to the target category y, respectively, and α is the parameters of classifier. Here $q \in (0, 1]$ is a hyperparameter that controls the degree of amplifying bias. Given $\theta_b = [\gamma_b, \alpha_b]$, the gradient of the GCE loss up-weights the gradient of the standard cross entropy (CE) loss for the samples with a high confidence C_b^y of predicting the correct target category as follows:

$$\frac{\partial GCE(C_b(z; \alpha_b), y)}{\partial \theta_b} = (C_b^y)^q \frac{\partial CE(C_b(z; \alpha_b), y)}{\partial \theta_b}. \tag{4}$$

Therefore, compared with CE loss, GCE loss will amplify the gradients of θ_b on samples by the confidence score $(C_b^y)^q$. Based on our observation that the bias information is usually easier to be learned, so the biased graphs will have higher $(C_b^y)^q$ than unbiased graphs. Therefore, the model g_b and C_b trained by GCE loss will focus on bias information and finally get the bias subgraph. Note that, to ensure that C_b predicts target labels mainly based on this z_b, the loss from C_b is not backpropagated to g_c, $i.e.$, only update θ_b in Eq. (4), and vice versa.

Meanwhile, we also train a causal GNN simultaneously with the weighted CE loss. The graphs with high CE loss from C_b can be regarded as the unbiased samples compared with the samples with low CE loss. In this regard, we could obtain the unbias score of each graph as

$$W(z) = \frac{CE(C_b(z), y)}{CE(C_c(z), y) + CE(C_b(z), y)}. \tag{5}$$

Large value of W implies the graph is an unbiased sample, hence we could use these weights to reweight the loss of these graphs to train g_c and C_c, enforcing them to learn the unbiased information. Thus, the objective function for learning disentangled representation is:

$$L_D = W(z) CE(C_c(z), y) + GCE(C_b(z), y). \tag{6}$$

4.3 Counterfactual Unbiased Sample Generation

Until now, we have achieved the first goal analyzed in Sec. 3.2 that is the disentanglement of causal and bias substructures. Next, we will show how to achieve the second goal that makes the causal variable z_c and bias variable z_b uncorrelated. Although we have disentangled causal and bias information, they are disentangled from the biased observed graphs. Hence, there will exist statistical correlation between causal and bias variables inheriting from the biased observed graphs. To further decorrelate z_c and z_b, according to the causal relation of data-generating process: $C \rightarrow G \leftarrow B$, we propose to generate the counterfactual unbiased samples in embedding space by swapping z_b. More specifically, we randomly permute bias vectors in each mini-batch and obtain $z_{unbiased} = [z_c; \hat{z}_b]$, where $\hat{z}_b$ represents the randomly permuted bias vectors of z_b. As z_c and $\hat{z}_b$ in $z_{unbiased}$ are randomly combined from different graphs, they will have much less correlation than $z = [z_c; z_b]$ where both are from the same graph. To make g_b and C_b still focus on the bias information, we also swap label y as $\hat{y}$ along with $\hat{z}_b$, so that the spurious correlation between $\hat{z}_b$ and $\hat{y}$ still exists. With the generated unbiased samples, we utilize the following loss function to train two GNN modules:

$$L_G = W(z)CE(C_c(z_{unbiased}), y) + GCE(C_b(z_{unbiased}), \hat{y}), \tag{7}$$

Together with the disentanglement loss, total loss function is defined as:

$$L = L_D + \lambda_G L_G, \tag{8}$$

where λ_G is a hyperparameter for weighting the importance of generation component. Moreover, training with more diverse samples would also benefit with better generalization on unseen testing scenarios. Our approach is summarized in App. B. Note that, as we need well-disentangled representations to generate the high-quality unbiased samples, in the early stage of training, we only train the model with L_D. After certain epochs, we train the model with L.

5 Experiment

Datasets. We construct three datasets with various properties and bias ratios to benchmark this new problem, where the datasets have clear causal subgraphs making the results explainable. Following CMNIST-75sp introduced in Sec. 3.1, we use the similar way to construct CFashion-75sp and CKuzushiji-75sp datasets based on the Fashion-MNIST [44] and Kuzushiji-MNIST [4] datasets. As the causal subgraphs of these two datasets are more complicated (fashion product and hiragana characters), they are more challenging. Due to the page limits, here we set bias degrees as $\{0.8, 0.9, 0.95\}$. We report the main results on unbiased test sets. Details are in App. C.1.

Baselines and experimental setup. As DisC is a general framework which could be built on various base GNN models, we select three popular GNNs: GCN [19], GIN [45], and GCNII [3]. The corresponding models are termed as DisC$_{GCN}$, DisC$_{GIN}$ and DisC$_{GCNII}$, respectively. Hence, base models are the most straight baselines. Another kind fo baselines are the causal-inspired GNN method DIR [43] and StableGNN [7]. We also compare against a general debiasing method LDD [23] by replacing its encoder with GNNs. Graph Pooling method DiffPool [48] and graph disentangling method FactorGCN [46] are also compared. To keep fair comparison, our model uses the same GNN architecture and hyperparameters with the corresponding base model. All the experiments are run 4 times with different random seeds and we report the accuracy and the standard error. More details are in App. C.2.

5.1 Quantitative Evaluation

Main results. The overall results are summarized in Table 1, and we have following observations:

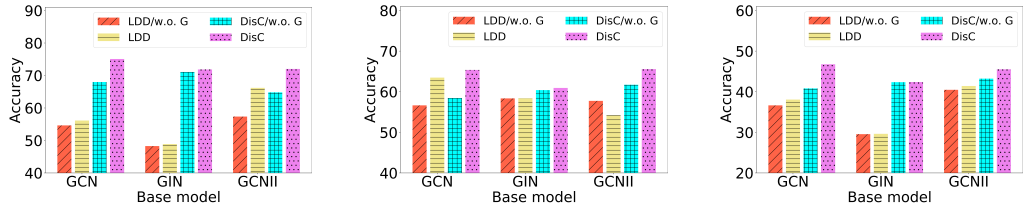
(1) DisC has much better generalization ability than base models. DisC outperforms the corresponding base model consistently with a large margin. With heavier biases, our model achieves larger improvements over base models. Specifically, for CMNIST-75sp, CFashion-75sp and CKuzushiji-75sp with smaller bias degree ($i.e.$, 0.8), our model achieves 40.02%, 4.47% and 29.82% average improvements over corresponding base models, respectively. Surprisingly, with severer biases (0.9 and 0.95), DisC achieves 169.17%, 14.67% and 49.35% average improvements over base models on three datasets, respectively. It indicates that the proposed method is a general framework helping existing GNNs against the negative impact of bias.

(2) DisC significantly outperforms existing debiasing methods. We notice that DIR could not achieve satisfying results. The reason is that DIR utilizes CE loss to extract bias information, which could not fully capture the property of bias in severe bias scenarios. And DIR sets one fixed threshold to spilt subgraphs, which is suboptimal. StableGNN outperforms their base model DiffPool and achieve competitive results, indicating the effectiveness of their proposed causal variable distinguishing regularizer. However, their framework adjusts data distribution based on the original dataset, it is hard to generate unbiased distribution when the unbiased samples are scarce. DisC could generate more unbiased samples based on the disentangled representations. Moreover, LDD is a general debiasing method which is not designed for graph data. DisC outperforms corresponding LDD variants with average 23.15%, indicating that the seamless joint of global-population-aware edge masker with debiasing disentangle framework is very effective.

Table 1: Graph classification accuracy evaluated on unbiased testing sets, which have same color (bias) set with training set. The best performance within each base model variant is in bold.

Dataset	CMNIST-75sp			CFashion-75sp			CKuzushiji-75sp		
Bias	0.8	0.9	0.95	0.8	0.9	0.95	0.8	0.9	0.95
FactorGCN [46]	$72.30_{\pm 1.18}$	$62.35_{\pm 5.07}$	$42.50_{\pm 4.91}$	$61.23_{\pm 1.11}$	$53.50_{\pm 1.29}$	$45.78_{\pm 2.40}$	$42.87_{\pm 1.19}$	$32.35_{\pm 2.79}$	$23.87_{\pm 0.12}$
DiffPool [48]	$73.79_{\pm 0.02}$	$66.45_{\pm 0.78}$	$47.12_{\pm 1.04}$	$62.82_{\pm 0.53}$	$57.50_{\pm 0.39}$	$50.86_{\pm 0.20}$	$45.46_{\pm 0.65}$	$36.18_{\pm 0.19}$	$27.45_{\pm 0.26}$
DIR [43]	$9.98_{\pm 0.33}$	$9.96_{\pm 0.23}$	$10.03_{\pm 0.27}$	$13.02_{\pm 1.92}$	$12.80_{\pm 1.67}$	$11.98_{\pm 1.41}$	$10.35_{\pm 0.32}$	$10.72_{\pm 0.27}$	$10.59_{\pm 0.46}$
StableGNN [7]	$77.65_{\pm 1.64}$	$68.87_{\pm 1.74}$	$51.33_{\pm 0.87}$	$64.03_{\pm 0.29}$	$58.26_{\pm 0.09}$	$51.46_{\pm 0.39}$	$49.41_{\pm 0.09}$	$39.30_{\pm 0.12}$	$28.26_{\pm 0.14}$
LDD_{GCN} [23]	$64.95_{\pm 1.22}$	$56.65_{\pm 2.18}$	$46.83_{\pm 2.88}$	$63.85_{\pm 1.17}$	$64.30_{\pm 0.89}$	$62.28_{\pm 0.48}$	$42.38_{\pm 0.33}$	$38.75_{\pm 0.49}$	$33.08_{\pm 0.59}$
LDD_{GIN} [23]	$64.88_{\pm 1.45}$	$50.59_{\pm 1.07}$	$31.23_{\pm 2.48}$	$64.65_{\pm 0.63}$	$57.10_{\pm 0.43}$	$53.38_{\pm 0.47}$	$37.83_{\pm 0.54}$	$28.97_{\pm 0.18}$	$22.13_{\pm 0.34}$
LDD_{GCNII} [23]	$78.03_{\pm 0.66}$	$69.53_{\pm 0.96}$	$51.05_{\pm 3.87}$	$50.63_{\pm 1.79}$	$54.09_{\pm 2.54}$	$57.93_{\pm 0.88}$	$48.70_{\pm 1.98}$	$41.59_{\pm 1.07}$	$33.93_{\pm 0.71}$
GCN [19]	$50.43_{\pm 4.13}$	$28.97_{\pm 4.4}$	$13.50_{\pm 1.38}$	$63.60_{\pm 0.53}$	$57.22_{\pm 0.93}$	$47.69_{\pm 0.42}$	$38.45_{\pm 1.1}$	$28.35_{\pm 0.79}$	$20.70_{\pm 0.88}$
DisC_{GCN}	$\mathbf{82.60_{\pm 0.93}}$	$\mathbf{78.14_{\pm 2.14}}$	$\mathbf{63.47_{\pm 5.65}}$	$\mathbf{66.85_{\pm 1.11}}$	$\mathbf{65.33_{\pm 4.70}}$	$\mathbf{63.93_{\pm 1.50}}$	$\mathbf{55.53_{\pm 2.29}}$	$\mathbf{48.13_{\pm 2.59}}$	$\mathbf{36.63_{\pm 1.73}}$
GIN [45]	$57.75_{\pm 0.78}$	$36.78_{\pm 5.55}$	$16.04_{\pm 1.14}$	$64.25_{\pm 0.46}$	$58.03_{\pm 0.40}$	$49.74_{\pm 0.60}$	$41.83_{\pm 0.78}$	$30.09_{\pm 0.87}$	$21.18_{\pm 1.63}$
DisC_{GIN}	$\mathbf{82.10_{\pm 1.50}}$	$\mathbf{74.90_{\pm 1.81}}$	$\mathbf{58.58_{\pm 4.24}}$	$\mathbf{67.10_{\pm 1.07}}$	$\mathbf{59.90_{\pm 1.31}}$	$\mathbf{55.80_{\pm 0.36}}$	$\mathbf{55.18_{\pm 1.00}}$	$\mathbf{41.75_{\pm 0.81}}$	$\mathbf{30.25_{\pm 1.63}}$
GCNII [3]	$69.70_{\pm 1.73}$	$57.68_{\pm 1.68}$	$41.00_{\pm 3.75}$	$\mathbf{66.68_{\pm 0.59}}$	$60.58_{\pm 0.28}$	$53.18_{\pm 0.08}$	$48.53_{\pm 0.25}$	$36.23_{\pm 0.20}$	$25.60_{\pm 0.76}$
DisC_{GCNII}	$\mathbf{79.50_{\pm 2.48}}$	$\mathbf{76.00_{\pm 1.90}}$	$\mathbf{60.54_{\pm 5.33}}$	$66.47_{\pm 1.77}$	$\mathbf{65.48_{\pm 0.70}}$	$\mathbf{61.75_{\pm 0.27}}$	$\mathbf{54.90_{\pm 1.30}}$	$\mathbf{44.73_{\pm 1.55}}$	$\mathbf{36.95_{\pm 0.70}}$

Ablation studies. To validate the importance of each module in our method, in Fig. 4, we conduct ablation studies on our variants (w.o. G means without the sample generation module) and the related variants of LDD. The major difference between DisC/w.o. G with LDD /w.o. G is the edge mask module. In most cases, DisC/w.o. G significantly outperforms LDD /w.o. G, indicating the necessity of learning edge mask for graph data. And DisC which has counterfactual sample generation module could further boost the performances based on the disentangled embeddings of DisC/w.o. G. However, LDD seldomly outperforms LDD /w.o. G or even achieves worse performances. That is, generating high-quality counterfactual samples needs well-disentangled causal and bias embeddings. If embeddings are not well-disentangled, counterfactual samples may act as noisy samples, which would prevent models from achieving further improvement. The edge masker could help the model generate well-disentangled embeddings, which is crucial for overall performance.



(a) CMNIST-75sp (b) CFashion-75sp (c) CKuzushiji-75sp

Figure 4: Ablation studies of the DisC vs. LDD average over three bias degrees of each dataset.

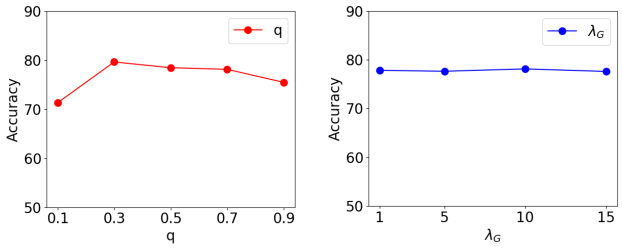
Robustness on unseen bias. Table 2 reports the results of DisC compared with its corresponding base models on testing set with unseen bias, $i.e.$, the pre-defined color (bias) sets of training set and testing set are disjoint. The performances of base models further drop compared with the results on seen bias scenario in Table 1. However, our model still achieves very stable performances, fully demonstrating the generalization ability of our model on agnostic bias scenario.

Hyperparameter experiments Fig. 5 is the hyperparameter experiments of the degree of amplifying bias q in GCE loss and the importance of generation component λ_G. For q, we fix $\lambda_G = 10$ and vary q from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. For λ_G, we fix $q = 0.7$ and vary λ_G from $\{1, 5, 10, 15\}$. From the results, we can see that our model achieves stable performance across different values of q and

Table 2: The results on unseen unbiased testing sets, $i.e.$, the color has not been seen in training set.

Dataset	CMNIST-75sp			CFashion-75sp			CKuzushiji-75sp		
Bias	0.8	0.9	0.95	0.8	0.9	0.95	0.8	0.9	0.95
DIR	$10.38_{\pm 0.28}$	$10.14_{\pm 0.40}$	$9.77_{\pm 0.18}$	$16.77_{\pm 1.71}$	$16.51_{\pm 3.20}$	$12.59_{\pm 1.61}$	$10.48_{\pm 0.34}$	$10.33_{\pm 0.75}$	$10.59_{\pm 0.95}$
GCN	$36.88_{\pm 5.16}$	$23.07_{\pm 4.07}$	$11.88_{\pm 0.33}$	$59.33_{\pm 0.55}$	$53.65_{\pm 0.47}$	$45.60_{\pm 1.06}$	$36.35_{\pm 0.48}$	$27.88_{\pm 0.94}$	$19.95_{\pm 0.67}$
DisC$_{GCN}$	$\mathbf{82.73_{\pm 1.31}}$	$\mathbf{77.70_{\pm 0.87}}$	$\mathbf{65.48_{\pm 0.76}}$	$\mathbf{67.9_{\pm 1.45}}$	$\mathbf{68.28_{\pm 0.18}}$	$\mathbf{63.77_{\pm 1.37}}$	$\mathbf{57.80_{\pm 2.38}}$	$\mathbf{51.60_{\pm 0.41}}$	$\mathbf{41.60_{\pm 3.94}}$
GIN	$48.93_{\pm 2.99}$	$34.95_{\pm 0.86}$	$14.53_{\pm 0.97}$	$58.88_{\pm 0.57}$	$53.80_{\pm 0.52}$	$48.43_{\pm 0.69}$	$39.25_{\pm 0.57}$	$30.75_{\pm 1.45}$	$22.35_{\pm 0.86}$
DisC$_{GIN}$	$\mathbf{77.80_{\pm 1.33}}$	$\mathbf{73.00_{\pm 0.61}}$	$\mathbf{58.80_{\pm 1.66}}$	$\mathbf{67.15_{\pm 0.79}}$	$\mathbf{59.98_{\pm 0.62}}$	$\mathbf{51.70_{\pm 0.34}}$	$\mathbf{55.47_{\pm 0.98}}$	$\mathbf{43.20 \pm 1.36}$	$\mathbf{31.33_{\pm 1.71}}$
GCNII	$53.50_{\pm 6.23}$	$45.52_{\pm 2.26}$	$32.6_{\pm 5.66}$	$58.85_{\pm 1.89}$	$53.98_{\pm 0.85}$	$46.97_{\pm 1.38}$	$39.93_{\pm 0.88}$	$30.33_{\pm 1.17}$	$23.09_{\pm 1.83}$
DisC$_{GCNII}$	$\mathbf{79.65_{\pm 2.13}}$	$\mathbf{76.63_{\pm 1.38}}$	$\mathbf{60.00_{\pm 5.66}}$	$\mathbf{60.50_{\pm 2.77}}$	$\mathbf{63.05_{\pm 2.25}}$	$\mathbf{61.78_{\pm 1.60}}$	$\mathbf{56.23_{\pm 3.45}}$	$\mathbf{49.10_{\pm 2.05}}$	$\mathbf{41.05_{\pm 0.11}}$

λ_G. When $q = 0.1$, it means the GCE loss will nearly reduce to normal CE loss. We can see the performance of DisC$_{GCN}$ is worse than other scenarios, demonstrating the effectiveness of utilizing GCE loss.

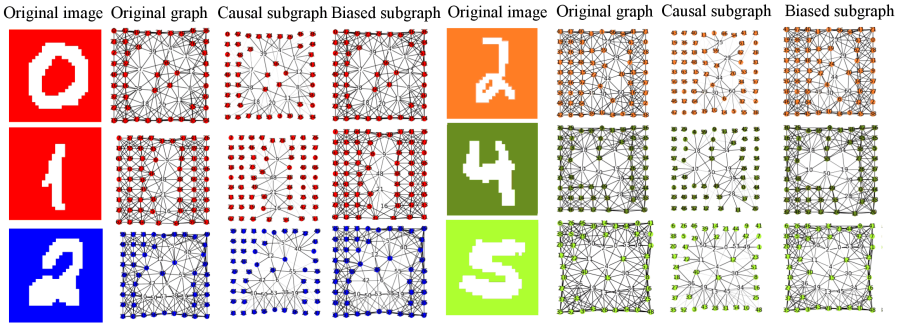


(a) q (b) λ_G

Figure 5: The hyperparameter experiments of q and λ_G

5.2 Qualitative Evaluation

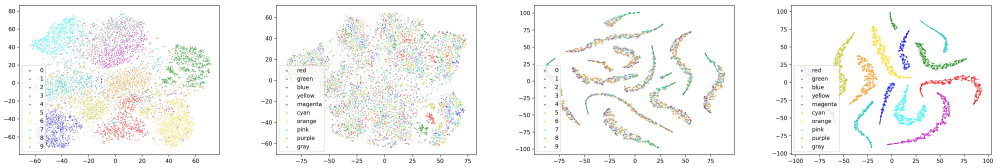
Visualization of edge mask. To better illustrate the significant causal and bias subgraphs extracted by DisC$_{GCN}$, we visualize the original images, original graph, and corresponding causal subgraph and bias subgraph of CMNIST-75sp with 0.9 bias degree in Fig. 6, where the width of edge represents the value of learned weight c_{ij} or b_{ij}. Fig. 6(a) shows the visualization results of testing graphs with the bias (color) that has been seen in the training set. As we can see, our model could discover the causal subgraphs where the most salient edges are in the digital subgraphs. With these causal subgraphs that highlight the structure information of digital, the GNNs will more easily extract this causal information. Fig. 6(b) shows the visualization results of testing graphs with unseen bias. According to the visualization, our model could still discover the causal subgraph outline, indicating our model could recognize causal subgraphs, whether the bias is seen or unseen. The visualization results of CFashion-75sp and CKuzushiji-75sp are shown in App. D.



(a) Testing graphs with seen bias in training set (b) Testing graphs with unseen bias in training set

Figure 6: Visualization of subgraphs extracted by DisC. The width of edge is edge weight c_{ij} or b_{ij}.

Projection of disentangled representation. Fig. 7 shows the projection of latent vectors z_c and z_b extracted from the causal GNN g_c and bias GNN g_b of DisC$_{GCN}$, respectively, using t-SNE [21] on



(a) z_c labeled by digit. (b) z_c labeled by color. (c) z_b labeled by digit. (d) z_b labeled by color.

Figure 7: Visualization of z_c and z_b with colors labeled by the digit and bias (color) labels. We observe that z_c and z_b are well clustered according to the groundtruth labels and bias labels, respectively.

CMNIST-75sp. Fig. 7 (a-b) are the projections of z_c labeled by the target labels (digit) and bias labels (color), respectively. Fig. 7 (c-d) are the projections of z_b labeled by the target labels and bias labels, respectively. We observe that z_c are clustered according to the target labels while z_b are clustered with the bias labels. And z_c are mixed with bias labels and z_b are mixed with target labels. The results indicate that DisC successfully learns the disentangled causal and bias representations.

Transferability of the learned mask. As our model could extract GNN-independent subgraphs, the learning edge weights can be used to purify original biased graphs. These sparse subgraphs represent significant semantic information and can be universally transferred to any GNNs. To validate this point, we learn the edge mask by DisC$_{GCN}$ and prune the edges with least $\{0\%, 20\%, 40\%, 60\%\}$ weights while keeping the remaining edge weights. Then we train vanilla GIN and GCNII on these weighted pruned datasets. Fig. 8 is the comparison of the results, where the dashed lines represent the results of base model on original biased graphs and the solid lines represent the performance of GNNs on weighted pruned datasets. The results show that the GNNs trained on the pruned datasets achieve better performances, indicating our learned edge mask has considerable transferability.

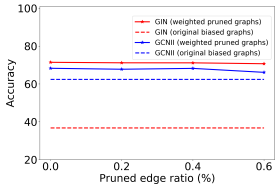


Figure 8: Performance of GIN and GCNII on the weighted pruned graphs found by DisC$_{GCN}$.

6 Conclusion

In this paper, we are first to study the generalization problem of GNNs on severe bias datasets, which is crucial to study the transparently knowledge learning mechanism of GNNs. We analyze the problem in a causal view that the generalization of GNNs will be hindered by entangled representations as well as the correlation between causal and bias variables. To remove the impact from these two aspects, we propose a general disentangling framework, DisC, which extracts causal substructure and bias substructure by two different functional GNNs, respectively. After the representations are well-disentangled, we proliferate the counterfactual unbiased samples by randomly swapping the disentangled vectors. With the new constructed benchmarks, we clearly validate the effectiveness, robustness, interpretability, and transferability of our method.

Acknowledgments and Disclosure of Funding

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, 62172052, 62002029, 62172052, U1936014). This work is also partially supported by the Natural Sciences and Engineering Research Council (NSERC) Discovery Grant, the Canada CIFAR AI Chair Program, collaboration grants between Microsoft Research and Mila, Samsung Electronics Co., Ltd., Amazon Faculty Research Award, Tencent AI Lab Rhino-Bird Gift Fund and a NRC Collaborative R&D Project (AI4D-CORE-06). This project was also partially funded by IVADO Fundamental Research Project grant PRF-2019- 3583139727. The work of Shaohua Fan is supported by the China Scholarship Council (No.202006470078). The computation resource of this project is supported by Compute Canada[5].

[5]https://www.computecanada.ca/

References

[1] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *ICML*, 2020.

[2] Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. In *NeurIPS*, 2019.

[3] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *ICML*, 2020.

[4] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

[5] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4883–4894, 2019.

[6] Luke Darlow, Stanisław Jastrzębski, and Amos Storkey. Latent adversarial debiasing: Mitigating collider bias in deep neural networks. *arXiv preprint arXiv:2011.11486*, 2020.

[7] Shaohua Fan, Xiao Wang, Chuan Shi, Peng Cui, and Bai Wang. Generalizing graph neural networks on out-of-distribution graphs. In *arXiv preprint arXiv:2111.10657*, 2021.

[8] Shaohua Fan, Xiao Wang, Chuan Shi, Kun Kuang, Nian Liu, and Bai Wang. Debiased graph neural networks with agnostic label selection bias. *TNNLS*, 2022.

[9] Shaohua Fan, Xiao Wang, Chuan Shi, Emiao Lu, Ken Lin, and Bai Wang. One2multi graph autoencoder for multi-view graph clustering. In *WWW*, pages 3070–3076, 2020.

[10] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. Metapath-guided heterogeneous graph neural network for intent recommendation. In *SIGKDD*, pages 2478–2486, 2019.

[11] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[12] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[13] Madelyn Glymour, Judea Pearl, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[15] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[16] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020.

[17] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *CVPR*, 2019.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2016.

[20] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. In *NeurIPS*, 2019.

[21] G.E. Kvan der Maaten, L.J.P.; Hinton. Understanding attention and generalization in graph neural networks. *Journal of Machine Learning Research*, 2008.

[22] John Boaz Lee, Ryan Rossi, and Xiangnan Kong. Graph classification using structural attention. In *SIGKDD*, 2018.

[23] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. In *NeurIPS*, 2021.

[24] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. Ood-gnn: Out-of-distribution generalized graph neural network. *arXiv preprint arXiv:2112.03806*, 2021.

[25] Yi Li and Nuno Vasconcelos. Repair: Removing representation bias by dataset resampling. In *CVPR*, 2019.

[26] Renjie Liao, Raquel Urtasun, and Richard Zemel. A pac-bayesian approach to generalization bounds for graph neural networks. In *ICLR*, 2020.

[27] Yanbei Liu, Xiao Wang, Shu Wu, and Zhitao Xiao. Independence promoted graph disentangled networks. In *AAAI*, 2020.

[28] Ana Lucic, Maartje ter Hoeve, Gabriele Tolomei, Maarten de Rijke, and Fabrizio Silvestri. Cf-gnnexplainer: Counterfactual explanations for graph neural networks. In *AISTATS*, 2022.

[29] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. In *NeurIPS*, 2020.

[30] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. Disentangled graph convolutional networks. In *ICML*, pages 4212–4221. PMLR, 2019.

[31] Jiaqi Ma, Junwei Deng, and Qiaozhu Mei. Subgroup generalization and fairness of graph neural networks. In *NeurIPS*, 2021.

[32] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. In *NeurIPS*, 2020.

[33] Judea Pearl. *Causality*. Cambridge university press, 2009.

[34] Hans Reichenbach. *The direction of time*, volume 65. Univ of California Press, 1991.

[35] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *ICLR*, 2019.

[36] 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.

[37] Zhenchao Sun, Hongzhi Yin, Hongxu Chen, Tong Chen, Lizhen Cui, and Fan Yang. Disease prediction via graph neural networks. *IEEE Journal of Biomedical and Health Informatics*, 25(3):818–826, 2020.

[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2017.

[39] Minh Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. In *NeurIPS*, 2020.

[40] Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. Learning robust representations by projecting superficial statistics out. *arXiv preprint arXiv:1903.06256*, 2019.

[41] Jianian Wang, Sheng Zhang, Yanghua Xiao, and Rui Song. A review on graph neural network methods in financial applications. *arXiv preprint arXiv:2111.15367*, 2021.

[42] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. Handling distribution shifts on graphs: An invariance perspective. In *ICLR*, 2022.

[43] Ying-Xin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat-Seng Chua. Discovering invariant rationales for graph neural networks. In *ICLR*, 2022.

[44] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[45] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[46] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. Factorizable graph convolutional networks. In *NeurIPS*, volume 33, pages 20286–20296, 2020.

[47] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *NeurIPS*, 2019.

[48] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *NeurIPS*, volume 31, 2018.

[49] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In *ICML*, 2021.

[50] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *AAAI*, 2018.

[51] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *NeurIPS*, 2018.

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] See App. E.
   (c) Did you discuss any potential negative societal impacts of your work? [No] We could not foresee any potential negative societal impacts of our work.
   (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] See Sec. 3.2.
   (b) Did you include complete proofs of all theoretical results? [Yes]
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] We provide the URL of code and data for reproducing the main results.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 5 and App. C.2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Sec. 5.
   (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)? [Yes] See Sec. 5.
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 construct new assets based on existing assets. We have cited them in Sec. 5.

(b) Did you mention the license of the assets? [Yes] See App. C.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We release new assets through a URL in Sec. 5.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The source data for generating our data is publicly available.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We do not have personally identifiable information or offensive content.

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]

A Preliminaries of Causal Inference

A.1 Structural Causal Models

In order to rigorously formalize our causal assumption behind the dataset, we resort to the Structural Causal Models, or SCM. SCM is a way of describing the relevant features (variables) of a particular problem and how they interact with each other. In particular, an SCM describes how the system assigns values to variables of interest.

Formally, an SCM consists of a set of *exogenous variables* U and a set of *endogenous variables* V, and a set of functions f that determines the values of variables in V based on the other variables in the model. Casually, a variable X is a *direct cause* of a variables Y if X exists in the function that determines the value of Y. If X is a direct cause of Y or of any cause of Y, X is a *cause* of Y. Exogenous variables U roughly means that they are external to the model, hence, in most scenarios, we choose not to explain how they are caused. Every endogenous variable is a descendant of at least one exogenous variable. Exogenous variables can only be the root variables. If we know the value of every exogenous variable, with the functions in f , we can perfectly determine the value of every endogenous variable. In many cases, we usually assume that all exogenous variables are unobserved variables like noise and are independently distributed with an expected value zero, so we only interest with the interaction with endogenous variables. Every SCM is associated with a *graphical causal model* or simply referred to "casual graph". Causal graph consists of nodes representing the variables in U and V, and the direct edges between the nodes representing the functions in f. Note the in our SCM in Section 3.2, we only show the endogenous variables we are interested in.

A.2 d-separation/connection

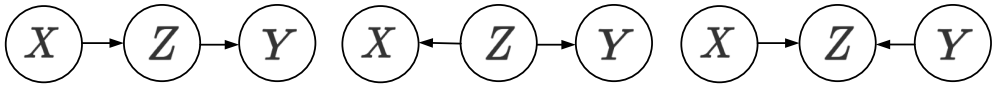
Given an SCM, we are particularly interested in (conditional) dependence information that is embedded in the model. There are three basic relationships of variables in an SCM, $i.e.$, chains, forks and colliders, as shown in Fig. 9. For chains and forks, X and Y would be dependent if Z is not in the conditional set, $i.e.$, the path is unblocked, and vice versa. And for colliders, X and Y would be independent if Z is not in the conditional set, $i.e.$, the path is blocked. Built upon these rules, d-separation is a criterion that can be applied in causal graphs of any complexity in order to predict dependencies that are shared by all datasets generated by the graph [13]. Two nodes X and Y are d-separated if every path between them is blocked. If even one path between X and Y is unblocked, X and Y are d-connected. Formally, we have following definition of d-separation:

Definition 1 (d-separation [13]) *A path p is blocked by a set of nodes Z if and only if*

1. p contains a chain of nodes $A \rightarrow B \rightarrow C$ or a fork $A \leftarrow B \rightarrow C$ such that the middle node B is in Z ($i.e.$, B is conditioned on), or

2. p contains a collider $A \rightarrow B \leftarrow C$ such that the collision node B is not in Z, and no descendant of B is in Z.

With this principle, we could find that the paths **(1)** $\mathbf{B \rightarrow G \rightarrow E \rightarrow Y}$ and **(2)** $\mathbf{B \leftrightarrow C \rightarrow Y}$ in Section 3.2 are unblocked paths, which would induce unexpected correlation between bias variable B and prediction Y.



(a) Chain (b) Fork (c) Collider

Figure 9: Three basic relations in causal graph.

More knowledge of causal inference, please refer to [13, 33].

B Algorithm

Algorithm 1: Disentangled Casual Substructure Learning

Input : graph G, label y, max iteration T, generation iteration t_{gen}
Output : Learned edge mask generator MLP, two GNN networks (g_c, g_b), and two classifiers (C_c, C_b)
Initialization : iteration $t = 0$; Initialize MLP, (g_c, g_b) and (C_c, C_b)

while *not converged* or $t < T$ **do**
  Extract subgraphs G_c and G_b based on edge mask generator;
  Encode subgraphs G_c and G_b into z_c and z_b via $g_c(G_c)$ and $g_b(G_b)$;
  Concatenate $z = [z_c; z_b]$;
  Update (g_c, g_b) and (C_c, C_b) via L_D in Eq. (6);
  if $t > t_{gen}$:
    Randomly swap $z = [z_c; z_b]$ into $z_{unbiased} = [z_c; \hat{z_b}]$;
    Update (g_c, g_b) and (C_c, C_b) via L in Eq. (8);
  $t = t + 1$;
end

C Experimental Details

C.1 Datasets details

Table 3: Statistics of Biased Graph Classification Datasets.

Dataset	Causal subgraph type	Bias subgraph type	#Graphs(train/val/test)	#Classes	#Avg. Nodes	#Avg. Edges	Node feat (dim.)	Bias degree	Difficulties
CMNIST-75sp	Digit subgraph	Color background subgraph	10K/5K/10K	10	61.09	488.78	Pixel+Coord (5)	0.8/0.9/0.95	Easy
CFashion-75sp	Fashion product subgraph	Color background subgraph	10K/5K/10K	10	61.03	488.26	Pixel+Coord (5)	0.8/0.9/0.95	Medium
CKuzushiji-75sp	Hiragana subgraph	Color background subgraph	10K/5K/10K	10	52.87	423.0	Pixel+Coord (5)	0.8/0.9/0.95	Hard

We summarize statistics of datasets constructed in this paper in Table 3. Note that the bias degree of validation set is 0.5, we use it to adjust the learning rate during training process. Without loss of any generality, here we subsample original 60K training samples into 10K training samples to make the training process more efficient. One could easily construct full dataset with our method. Each graph of CFashion-75sp is labeled by the category of fashion product it belongs to and each graph of CKuzushiji-75sp is labeled by one of 10 Hiragana characters. Moreover, we would like to list the map between label and predefined correlated color for all datasets in Table 4. The links for source image datasets are as follows:

1. MNIST: http://yann.lecun.com/exdb/mnist/.
2. Fashion-MNIST: https://github.com/zalandoresearch/fashion-mnist. MIT License.
3. Kuzushiji-MNIST: https://github.com/rois-codh/kmnist. CC BY-SA 4.0 License.

Table 4: Mapping between label and color.

Label	Color (RGB)	Label	Color (RGB)
0	(255, 0, 0)	5	(0, 255, 255)
1	(0, 255, 0)	6	(255, 128, 0)
2	(0, 0, 255)	7	(255, 0, 128)
3	(225, 225, 0)	8	(128, 0, 255)
4	(225, 0, 225)	9	(128, 128, 128)

For unbiased testing dataset with unseen bias used in Table 2, the RGB value of predefined color set is {(199, 21, 133), (255, 140, 105), (255, 127, 36), (139, 71, 38), (107, 142, 35), (173, 255, 47), (60, 179, 113), (0, 255, 255), (64, 224, 208), (0, 191, 255)}.

C.2 Experimental setup

For GCN and GIN, we use the same model architectures as [15][6], which have 4 layers, and 146 hidden dimension for GCN and 110 hidden dimension for GIN. And GIN utilizes its GIN0 variant. For GCNII, it has 4 layers and 146 hidden dimension. DIR[7] utilizes the default parameters in original paper for MNIST-75sp dataset. For causal GNN or bias GNN in our model, it has the same architecture with base model. We optimize all models with the Adam [18] optimizer and 0.01 learning rate with for all experiments. The batch-size for all the methods is 256. We train all the models with 200 epochs and set the generation iteration of our method t_{gen} as 100. For our model, we set q of GCE loss as 0.7 and λ_G as 10 for all experiments. Our substructure generator is a two-layer MLP, whose activation function is sigmoid function. For StableGNN, we use their GraphSAGE variant. For other baselines, we use their default hyperparameters. LDD[8] has same hyparameters with our model. To better reflect the performance of unbiased sample generation, we take the performances of last step as final results. All the experiments are conducted on the single NVIDIA V100 GPU.

D Visualization of CFashion-75sp and CKuzushiji-75sp

Figure 10 and Figure 11 are visualization results of CFashion-75sp and CKuzushiji-75sp dataset. As we can see, our model could also discover reasonable causal subgraphs for these challenging datasets.

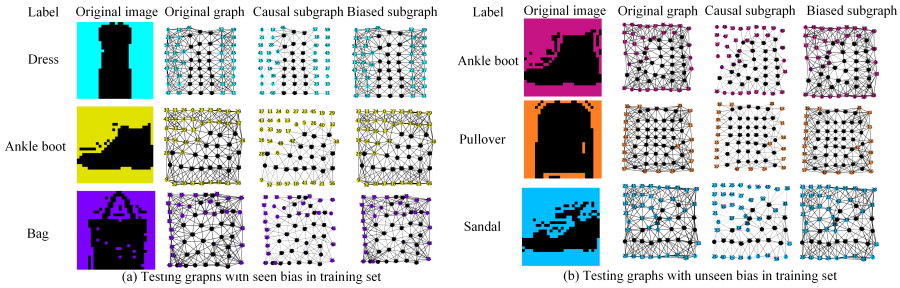


Figure 10: Visualization of subgraphs extracted by the mask generator from CFashion-75sp.

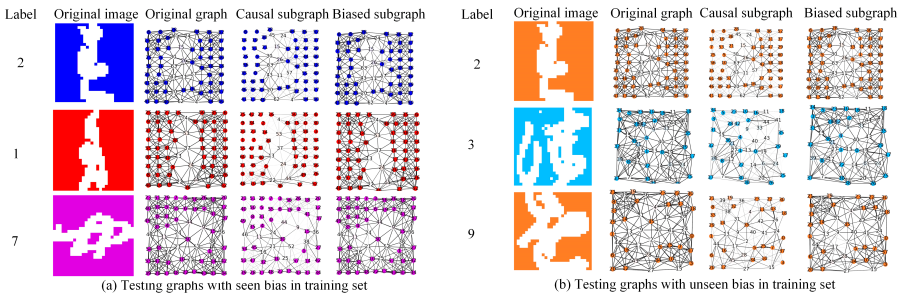


Figure 11: Visualization of subgraphs extracted by the mask generator from CKuzushiji-75sp.

E Limitations and societal impacts

Our method assumes that the graph consists of causal subgraph and bias subgraph. In reality, it may also exist non-informative subgraphs, which is neither causal nor biased for label. We would

[6]https://github.com/graphdeeplearning/benchmarking-gnns.

[7]https://github.com/Wuyxin/DIR-GNN.

[8]https://github.com/kakaoenterprise/Learning-Debiased-Disentangled

like to consider more fine-grained splitting of graphs in the future. When deploying GNNs to real-world applications, especially safety-critical fields, whether the results of GNNs are stable is an important factor. The demands for a stable model are universal and extensive such as in the field of disease prediction [37], traffic states prediction [5], and financial applications [41], where utilizing human-understandable causal knowledge for prediction is necessary.