Learning to Reweight for Graph Neural Network

Zhengyu Chen[1,2], Teng Xiao[3], Kun Kuang[1,2], Zheqi Lv[1,2], Min Zhang[1,2], Jinluan Yang[1,2], Chengqiang Lu[4], Hongxia Yang[4], and Fei Wu[1,2]

[1] Institute of Artificial Intelligence, Zhejiang University
[2] Shanghai Institute for Advanced Study, Zhejiang University
[3] The Pennsylvania State University
[4] DAMA Academy, Alibaba Group.
{kunkuang, wufei}@zju.edu.cn, yang.yhx@alibaba-inc.com

Abstract

Graph Neural Networks (GNNs) show promising results for graph tasks. However, existing GNNs' generalization ability will degrade when there exist distribution shifts between testing and training graph data. The cardinal impetus underlying the severe degeneration is that the GNNs are architected predicated upon the I.I.D assumptions. In such a setting, GNNs are inclined to leverage imperceptible statistical correlations subsisting in the training set to predict, albeit it is a spurious correlation. In this paper, we study the problem of the generalization ability of GNNs in Out-Of-Distribution (OOD) settings. To solve this problem, we propose the Learning to Reweight for Generalizable Graph Neural Network (L2R-GNN) to enhance the generalization ability for achieving satisfactory performance on unseen testing graphs that have different distributions with training graphs. We propose a novel nonlinear graph decorrelation method, which can substantially improve the out-of-distribution generalization ability and compares favorably to previous methods in restraining the over-reduced sample size. The variables of the graph representation are clustered based on the stability of the correlation, and the graph decorrelation method learns weights to remove correlations between the variables of different clusters rather than any two variables. Besides, we interpose an efficacious stochastic algorithm upon bi-level optimization for the L2R-GNN framework, which facilitates simultaneously learning the optimal weights and GNN parameters, and avoids the overfitting problem. Experimental results show that L2R-GNN greatly outperforms baselines on various graph prediction benchmarks under distribution shifts.

Introduction

Graph Neural Networks (GNNs) have achieved state-of-the-art performances on various graph tasks (Kipf and Welling 2016; Veličković et al. 2017; Xu et al. 2018a), but they assume that the training and testing data are independent and identically distributed (i.e., i.i.d assumption), which is not always the case in real-world applications (Chen, Xu, and Wang 2021; Chen and Wang 2021; Chen, Gai, and Wang 2019; Chen, Wang, and Yin 2021; Gai et al. 2019). This leads to inadequate out-of-distribution (OOD) generalization ability, causing significant performance degradation under distribution shifts (Hu et al. 2020; Wu et al. 2018; Chen et al. 2021).

Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

GNNs' inadequate out-of-distribution generalization is caused by a spurious correlation between irrelevant features and category labels in training data (Xiao, Chen, and Wang 2022; Chen et al. 2023b). This correlation varies across distributions and is exploited by GNNs for inference. An example of spurious correlation is shown in the graph classification task of the "wheel" motif in Figure 1. In the biased training dataset, most positive graphs have only "star" motifs added, leading to a strong correlation between structural features of "wheel" motifs and "star" motifs. This unexpected correlation leads to a spurious correlation between the structural features of "star" motifs and the label "wheel". The GCN model exploits this spurious correlation and tends to use "star" motifs for prediction, making false predictions on negative graphs with a "star" motif.

To solve the spurious correlation problem caused by the discrepancy between training and testing distributions, earlier research attempts to train a model with stability guarantee through variable decorrelation with sample reweighting, taking model misspecification into consideration (Shen et al. 2020; Kuang et al. 2020; Lv et al. 2023b; Zhang et al. 2023; Lv et al. 2023a). However, the majority of these approaches are suggested in linear settings. GNNs combine heterogeneous data from node features and graph topological structures, resulting in the existence of intricate and unrecognized non-linear relationships across representations (Fan et al. 2021; Li et al. 2021). Non-linear dependencies on graph data cannot be removed using linear sample reweighting approaches. Recent studies propose non-linear decorrelation methods for graph tasks (Fan et al. 2021; Li et al. 2021). They attempt to eliminate the dependencies between all the variables of the graph representation through a set of learned sample weights. However, such a demanding aim might result in an excessively small sample size, which hampers the generalization ability of GNNs (Martino, Elvira, and Louzada 2017; Llorente et al. 2022; Zhang et al. 2022). Moreover, these non-linear decorrelation methodologies on graph data suffer from overfitting problems due to the additional hyperparameters, leading to difficulties in achieving convergence.

We suggest that not all correlations should be eliminated, in contrast to prior techniques (Fan et al. 2021; Li et al. 2021), which aggressively decorrelate all connections across graph representations. Such an aggressive objective may result in an issue with an overly-reduced sample size (Martino, Elvira,

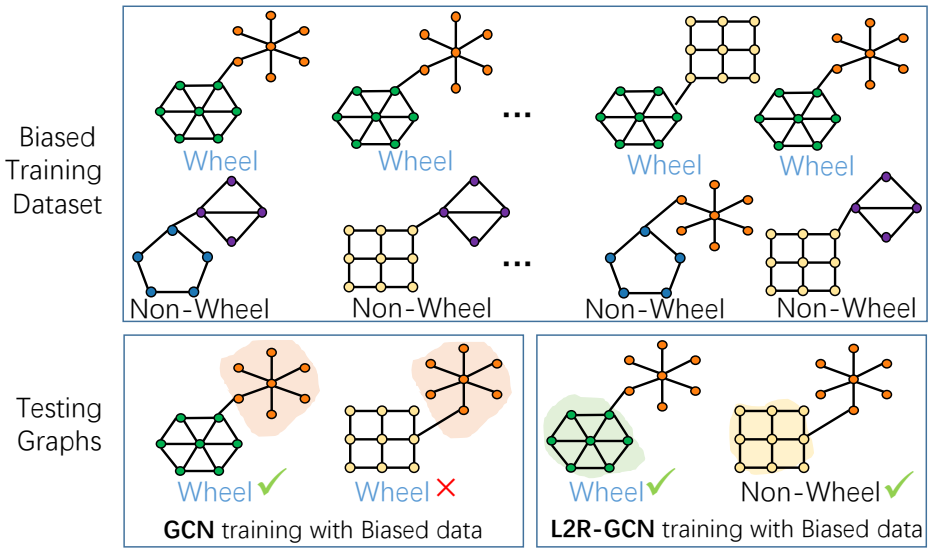


Figure 1: An illustration of a fictitious correlation in the "wheel" motif graph classification task.

and Louzada 2017; Llorente et al. 2022), which hampers the generalization ability of GNNs. Taking the graph classification task in Figure 1 as an example, although varied variables may be used to characterize the "wheel" motif's graph structure and the features of nodes; they function as an integrated whole, and these relationships remain stable across datasets with varied or unknown distribution shifts. The significant correlations between the variables in the "wheel" and the "star" due to the selection bias seen in the biased training dataset. Such "spurious" relationships, however, cannot be used in OOD datasets. Therefore, to obtain a precise graph model, we just need to eliminate the spurious correlations between two sets of variables (the "wheel" and the "star").

In this paper, we propose a framework called L2R-GNN to solve the problem of learning out-of-distribution graph representation. Our framework includes a nonlinear graph decorrelation method that reduces correlations between variables of various clusters. This method is more effective than previous approaches at controlling the overly-reduced sample size and can significantly increase the ability to generalize outside of the distribution. We group the variables of graph representations based on the stability of their correlations and learn a set of weights to remove spurious correlations. By doing so, graph neural networks can focus more on the real relationship between their ground-truth labels and the graph representations. We also introduce a stochastic approach based on bi-level optimization for the L2R-GNN framework. This approach allows for the simultaneous learning of the optimal GNN parameters and weights while avoiding the over-fitting problem. Our experimental results show that L2R-GNN outperforms baselines on different graph tasks subjected to distribution shifts. Our contributions are as follows: 1) We propose a novel framework that can learn effective graph representation under complex distribution shifts and achieve better performance simultaneously. - We propose a more effective graph decorrelation method than prior approaches at controlling the overly-reduced sample size and increasing the ability to generalize outside of the distribution. 2) We propose an effective stochastic algorithm based on bi-level optimization for the L2R-GNN framework, which enables simultaneously learning the optimal weights and GNN parameters and avoiding the over-fitting issue. 3) Our extensive empirical results on several graph benchmarks subjected to distribution shifts show that L2R-GNN greatly outperforms baselines in terms of performance.

Related Works

Generalizable Graph Neural Network. Most GNNs methods are proposed under the IID hypothesis, which states that the training and testing sets are independently sampled from the same distribution (Kipf and Welling 2016; Veličković et al. 2017). However, in practice, it might be challenging to satisfy this ideal hypothesis. Recent research (Fan et al. 2021; Li et al. 2021) studies how well GNNs generalize outside the training distribution. Several studies concentrate on size generalization ability to make GNNs function effectively on testing graphs whose size distribution is different from that of training graphs. SL-DSGCN (Tang et al. 2020) reduces the degree-related distribution shifts of GCNs for the OOD node classification task. ImGAGN (Qu et al. 2021) produces a set of synthetic minority nodes to balance the class distribution shifts. BA-GNN (Chen, Xiao, and Kuang 2022) is proposed to learn node representations that are invariant across various distributions for invariant prediction. EERM(Wu et al. 2022) helps GNNs take advantage of invariance principles for prediction on node-level problems. For OOD graph classification task, some works (Fan et al. 2021; Li et al. 2021) improve the generalization capability of GNNs via non-linear decorrelation methods. However, such an aggressive target might result in an issue with an excessively small sample size problem(Martino, Elvira, and Louzada 2017; Llorente

et al. 2022), which hampers the generalization ability of GNNs. Moreover, these non-linear decorrelation methods on graph data suffer from over-fitting issues due to the additional hyper-parameters and are hard to converge.

The Bi-level Optimization. Many works use bi-level optimization (Maclaurin, Duvenaud, and Adams 2015; Wang et al. 2020; Chen, Chen, and Wang 2021; Gai, Chen, and Wang 2021; Chen et al. 2023a) to improve the performance of GNNs. They optimize a higher-level learning subject for lower-level learning. To search the GNN architectures, several studies (Xiao et al. 2022; Xiao, Chen, and Wang 2023; Jiang et al. 2022) optimize a bi-level goal using reinforcement learning. Furthermore, (Xiao et al. 2021) presents bi-level programming with variational inference to provide a framework for learning propagation methods. The study (Liu et al. 2020) attempts to get a parameter initialization that can swiftly adapt to unfamiliar workloads utilizing gradient information from the bi-level optimization. Our study focuses on the capacity of GNNs to generalize in graph-level tasks, and we use bi-level programming to provide a framework for learning graph weights while avoiding over-fitting problems.

Method

Problem Formulation. Given the training graphs $\mathbf{G}_{train} = \{G_n, Y_n\}_{n=1}^N$, where G_n is the n-th graph and Y_n is the corresponding label. $\mathbf{G}_{test}$ is the testing graph which is unobserved in the training stage. The task is to learn a graph neural network $GNN(\theta) : \mathcal{G} \to \mathcal{Z}$ and classifier $\mathcal{R} : \mathcal{Z} \to \mathcal{Y}$ to predict the label of testing graphs $\mathbf{G}_{test}$, which is under distribution shifts $\mathrm{P}(\mathbf{G}_{train}) \neq \mathrm{P}(\mathbf{G}_{test})$. Denote graph representations $Z = GNN(\theta, G)$, $\mathbf{Z} \subset \mathbb{R}^{N \times d}$. $\mathbf{Z}_{i,j}$ represent the i-th row and j-th column in Z.

Graph Reweighting with RFF. Similar to previous works (Fan et al. 2021; Li et al. 2021), we decorrelate graph representations, removing statistical relationships between relevant and irrelevant graph representations. Relevant graph representation is invariant across many unknown testing graphs, while irrelevant representation varies. We remove statistical dependency of all dimensions in representation $\mathbf{Z}$, defined as: $\mathbf{Z}_{:,i} \perp\!\!\!\perp \mathbf{Z}_{:,j}, \forall i, j \in [1, d], i \neq j$. Hypothesis testing statistics evaluate independence between random variables. We use the Hilbert-Schmidt Independence Criterion (HSIC) to supervise feature decorrelation. If the product $k_{\mathbf{Z}_{:,i}} k_{\mathbf{Z}_{:,j}}$ is a characteristic kernel, we have $\mathrm{HSIC}(\mathbf{Z}_{:,i}, \mathbf{Z}_{:,j}) = 0 \Leftrightarrow \mathbf{Z}_{:,i} \perp\!\!\!\perp \mathbf{Z}_{:,j}$. However, HSIC is not suitable for training deep models on large datasets due to high computational cost. We use the Frobenius norm as the independent testing statistic for the graph representation space. The partial cross-covariance matrix is:$\hat{\Sigma}_{\mathbf{Z}_{:,i},\mathbf{Z}_{:,j}} = \frac{1}{N-1} \sum_{n=1}^{N} \left[\left(\mathbf{u}(\mathbf{Z}_{n,i}) - \frac{1}{N} \sum_{m=1}^{N} \mathbf{u}(\mathbf{Z}_{m,j}) \right)^T \cdot \left(\mathbf{v}(\mathbf{Z}_{n,j}) - \frac{1}{N} \sum_{m=1}^{N} \mathbf{v}(\mathbf{Z}_{m,j}) \right) \right]$ where $\mathcal{H}_{\mathrm{RFF}}$ represents the space of a random Fourier function. Using $n_u, n_v = 5$ is reliable enough to assess the independence of random variables in real-world situations.

Using the independence criterion, we apply graph reweighting to remove inter-variable dependencies in graph representation, and RFF to assess overall independence. The learnable graph weight $\mathbf{W} = \{w_n\}_{n=1}^N$ for the n-th graph G_n in the training set is $w_n \in \mathbb{R}$. The partial cross-covariance matrix after reweighting is:

$$\widehat{\Sigma}^{\mathbf{W}}_{\mathbf{Z}_{:,i},\mathbf{Z}_{:,j}} = \frac{1}{N-1} \sum_{n=1}^{N} \left[\left(w_n u(\mathbf{Z}_{ni}) - \frac{1}{N} \sum_{m=1}^{N} w_m u(\mathbf{Z}_{mi}) \right)^\top \cdot \left(w_n v(\mathbf{Z}_{nj}) - \frac{1}{N} \sum_{m=1}^{N} w_m v(\mathbf{Z}_{mj}) \right) \right]. \quad (1)$$

By reducing the squared Frobenius norm of the partial cross-covariance matrix $\|\widehat{\Sigma}^{\mathbf{W}}_{\mathbf{Z}_{:,i},\mathbf{Z}_{:,j}}\|_{\mathrm{F}}^2$, the optimal graph weight $\mathbf{W}^*$ reduces inter-variable dependencies in graph representation:

$$\mathbf{W}^* = \mathrm{argmin}_{\mathbf{W}} \sum_{1 \leq i < j \leq d} \|\widehat{\Sigma}^{\mathbf{W}}_{\mathbf{Z}_{:,i},\mathbf{Z}_{:,j}}\|_{\mathrm{F}}^2, \quad (2)$$

Reducing Eq. (1) directly eliminates correlations between any two variables of graph representation.

These methods are widely used in recent works (Fan et al. 2021; Li et al. 2021). However, this aggressive target in Eq. (1) hampers GNNs' generalization capacity due to an excessively decreased sample size problem.

Graph Decorrelation. We contend that *not all correlations* need to be eliminated, in contrast to prior approaches (Fan et al. 2021; Li et al. 2021), which aggressively decorrelate all dependencies between graph representations. As an example, consider the graph classification problem given in Figure 1. Although the features of the nodes and the graph structure of the "wheel" motif may be represented by several variables, they function as a single unit and exhibit consistent correlations across various unknown testing graphs. We can see the significant correlations between the variables in the "wheel" and the "star" due to the selection bias observed in the biased training dataset. Such "spurious" correlations, however, cannot be used for many unknown testing graphs. In order to get a correct graph model for such a situation, we just need to eliminate the erroneous connection between two sets of data (the "wheel" and the "star").

Specifically, we propose a novel nonlinear graph decorrelation method, which is more effective than previous approaches at limiting the overly-reduced sample size and may significantly increase the out-of-distribution generalization ability. The graph decorrelation approach learns a set of weights to reduce correlations between the variables of various clusters rather than any two variables. The variables of graph representation are grouped depending on the stability of their correlations. By eliminating erroneous correlations, the learnt weights enable graph neural networks to focus more on the real relationship between learned discriminative graph representations and their ground-truth labels.

We define the dissimilarity of two variables of graph representation Z as follows in order to express the invariant property of two variables through the variance of their correlation:

$$\text{Dis}\left(Z_{:,i}, Z_{:,j}\right) = \sqrt{\frac{1}{N-1}\sum_{l=1}^{N}\left(\text{Corr}\left(Z_{l,i}, Z_{l,j}\right) - Ave_{-}\ \text{Corr}\left(\hat{Z}_{:,i}, \hat{Z}_{:,j}\right)\right)^2} \tag{3}$$

where $Ave_{-}\ \text{Corr}\left(\hat{Z}_{:,i}, \hat{Z}_{:,j}\right)$ represents the average correlation across whole datasets and $\text{Corr}\left(Z_{l,i}, Z_{l,j}\right)$ represents the pearson correlation of $Z_{l,i}, Z_{l,j}$ in the lth graph. We load the full dataset in order to obtain $\hat{Z}_{:,i}$ and $\hat{Z}_{:,j}$. However, because of the high computational cost and enormous storage usage, it is practically impossible to apply in huge datasets. Thus, we propose a scalable method with momentum update.

It makes sense to cluster the variables with lower dissimilarity into the same cluster since they are more likely to retain a stable joint distribution over many graphs. Therefore, we choose the cluster with the closest mean as determined by the least squared Euclidean distance for each variable in the graph representation Z.

We could have K clusters of variables in graph representation Z, where jth cluster is S_j and cluster center is μ_j. Thus, we could learn μ and S by minimizing

$$\mu, S = \text{argmin}_{\mu,S}\sum_{j}^{k}\sum_{Z_{:,i}\in S_j} Dis(Z_{:,i}, \mu_j) \tag{4}$$

We could eliminate the correlation between the variables of distinct clusters rather than any two variables by combining the variable clustering of the graph representation. We could reformulate Eq. 2 as:

$$W^* = \text{argmin}_{\mathbf{W}}\sum_{1\le i<j\le d}\mathbb{I}(i,j)\|\widehat{\Sigma}^{\mathbf{W}}_{Z_{:,i},Z_{:,j}}\|_{\text{F}}^2, \tag{5}$$

where indicator variable $\mathbb{I}(i,j)$ returns 0 if the clusters of $Z_{:,i}$ and $Z_{:,j}$ are different and 1 if the clusters are the same.

To get optimal graph weights $\mathbf{W}$, graph neural network $GNN(\theta)$, and classifier $\mathcal{R}$, we have:

$$\theta^*, \mathcal{R}^* = \text{argmin}_{\theta,\mathcal{R}}\sum_{n=1}^{N} w_n \ell\left(\mathcal{R}\circ GNN(G,\theta), \mathbf{Y}_n\right), \tag{6}$$

$$\mathbf{W}^* = \text{argmin}_{\mathbf{W}}\sum_{1\le i<j\le d}\mathbb{I}(i,j)\|\widehat{\Sigma}^{\mathbf{W}}_{\mathbf{Z}_{:,i},\mathbf{Z}_{:,j}}\|_{\text{F}}^2, \tag{7}$$

where ℓ is the loss function. The statistical dependence between different clusters rather than all variables could be eliminated by jointly optimizing the graph neural network $GNN(\theta)$, classifier $\mathcal{R}$, and graph weights $\mathbf{W}$.

However, such non-linear decorrelation methods on graph data suffer from over-fitting issues due to the additional hyperparameters and are hard to converge. To overcome such over-fitting problem, we consider a bi-level optimization method for the model framework that allows for the simultaneous learning of the optimal GNN parameters and weights.

Bi-level Training Algorithm. We introduce our L2R-GNN framework to learn effective graph representation. However, as suggested by previous works (Ren et al. 2018; Xiao et al. 2021), sample reweighting algorithms suffer from over-fitting due to the additional hyperparameters and are hard to converge. In this section, we propose the bi-level training algorithm to alleviate the over-fitting problem.

For our proposed L2R-GNN framework, the introduced decorrelation method for joint learning sample reweights also increases the risk of over-fitting as shown in experiments. Inspired by gradient-based meta-learning (learning to learn), we use bi-level optimization to solve the over-fitting issue. Thus, the objective can be formulated as the following bi-level optimization problem:

$$\begin{aligned}\min_{W}\mathcal{L}_{\text{val}}\left(\theta^*(W), W\right) &= \sum_{1\le i<j\le d}\mathbb{I}(i,j)\|\Sigma^{\mathbf{W}}_{GNN(G_{val},\theta^*(W))}\|_{\text{F}}^2\\ \text{s.t. } \theta^*(W) &= \arg\min_{\theta}\mathcal{L}_{\text{train}}(\theta, W)\\ &= W\ell\left(\mathcal{R}\circ GNN\left(G_{train},\theta\right), \mathbf{Y}_{train}\right),\end{aligned} \tag{8}$$

This bi-level update aims to optimize the graph weights based on its validation for avoiding the over-fitting issues, where $\mathcal{L}_{\text{train}}(\theta, W)$ and $\mathcal{L}_{\text{val}}\left(\theta^*(W), W\right)$ are lower-level and higher-level objectives on the training and validation sets, respectively.

Since there is no closed-form expression for $theta$, it is hard to directly optimize the higher-level objective function in Eq. (8). We provide an alternating approximation approach to solve such problems.

Updating θ in outer loop. Different from previous works (Fan et al. 2021; Li et al. 2021), we do not solve the lower level problem for each outer loop. At the i-th iteration, we fix W and only perform the gradient steps for parameter θ with the learning rate η_θ that are listed below:

$$\theta^{(i)} = \theta^{(i-1)} - \eta_\theta\nabla_\theta\mathcal{L}_{\text{train}}(\theta^{(i-1)}, W^{(i-1)}), \tag{9}$$

Updating W in inner loop. We compute the higher-level objective in the inner loop after getting the parameter $\theta^{(i)}$, which is an estimate of $\theta^{(*)}(W)$):

$$W^{(i)} = W^{(i-1)} - \eta_W\nabla_W\mathcal{L}_{\text{val}}(\theta^{(i)}, W^{(i-1)}). \tag{10}$$

We could have the gradient of W, since the W is part of function $\theta^{(i)}$ due to Eq. (9), and the function $\nabla_W\mathcal{L}_{\text{val}}(\theta^{(i)}, W^{(i-1)})$ could be represented by:

$$\begin{aligned}&\nabla_W\mathcal{L}_{\text{val}}(\theta^{(i)}, W^{(i-1)}) = \nabla_W\mathcal{L}_{\text{val}}(\bar{\theta}^{(i)}, W^{(i-1)})\\ &- \eta_\theta\frac{1}{\epsilon}(\nabla_W\mathcal{L}_{\text{train}}(\theta^{(i-1)} + \epsilon\nabla_\theta\mathcal{L}_{\text{val}}(\theta^{(i)}, \bar{W}^{(i-1)}), W^{(i-1)})\\ &- \nabla_W\mathcal{L}_{\text{train}}(\theta^{(i-1)}, W^{(i-1)})),\end{aligned} \tag{11}$$

where $\bar{\theta}^{(i)}$ and $\bar{W}^{(i-1)}$ denote stopping the gradient. Set η_θ to 0 in Eq. (11) to obtain a first-order approximation as followed:

$$\nabla_W\mathcal{L}_{\text{val}}(\theta^{(i)}, W^{(i-1)}) = \nabla_W\mathcal{L}_{\text{val}}(\bar{\theta}^{(i)}, W^{(i-1)}). \tag{12}$$

By alternating the update procedures in Eqs. (9) and (10), we can derive the whole model algorithm from the above gradient derivations. Moreover, we investigate the impact of bi-level optimization as well as the first- and second-order

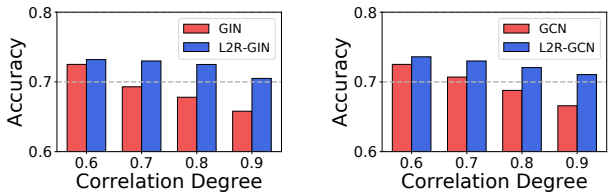


Figure 2: Results of GCN and GIN backbones under different correlation degree settings. Comparing with GCN and GIN method, our L2R-GNN methods (by applying our L2R-GNN framework on GCN and GIN backbone) improves the accuracy of graph classification across different spurious correlation degree settings.

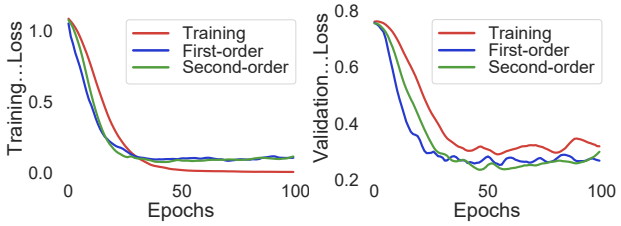


Figure 3: The training and validation loss curves on D&D.

approximations in experiments. Results show that the best performance is obtained with a first-order approximation.

Momentum Graph Weight Estimator. For each graph, a specific weight should be learned as shown in Equation 8. However, simultaneously loading the entire dataset for optimization is impractical due to high computational cost and excessive storage consumption, especially for large datasets. We employ weight queues with a K dimension to balance optimization performance and weight consistency. We have graph representation queue $\mathbf{Z}^{(q)} = [\mathbf{Z}^{(q_1)}, \cdots, \mathbf{Z}^{(q_K)}]$ and the corresponding weight queue $\mathbf{W}^{(q)} = [\mathbf{W}^{(q_1)}, \cdots, \mathbf{W}^{(q_K)}]$. During training, they act as a memory bank from earlier mini-batches.

The graph representations and weights used for optimization are constructed as follows for each mini-batch of input graphs G_n: $\widehat{\mathbf{Z}} = Concat(\mathbf{Z}^{(q_1)}, \cdots, \mathbf{Z}^{(q_K)}, \mathbf{Z}^{(l)})$, $\widehat{\mathbf{W}} = Concat(\mathbf{W}^{(q_1)}, \cdots, \mathbf{W}^{(q_K)}\mathbf{W}^{(l)})$. Using graph representations queues, we reformulate Eq.3 as:

$$\text{Dis}\left(Z_{:,i}, Z_{:,j}\right) = \sqrt{\frac{1}{N-1}\sum_{l=1}^{N}\left(\text{Corr}\left(Z_{:,i}^{(l)}, Z_{:,j}^{(l)}\right) - Ave\_\ \text{Corr}\left(\hat{Z}_{:,i}^{(q)}, \hat{Z}_{:,j}^{(q)}\right)\right)^2} \tag{13}$$

where the pearson correlation of $Z_{:,i}, Z_{:,j}$ in the mini-batch is represented by $\text{Corr}\left(Z_{:,i}^{(l)}, Z_{:,j}^{(l)}\right)$, and their average correlation across all graph representation queues is represented by $Ave\_\ \text{Corr}\left(\hat{Z}_{:,i}^{(q)}, \hat{Z}_{:,j}^{(q)}\right)$.

Our L2R-GNN reduce the computational cost via weight queues. If the batch size is B, then $\widehat{\mathbf{Z}}$ is a matrix with dimensions of $((k+1)B) \times m_Z$, and $\widehat{\mathbf{W}}$ is a vector with dimensions of $(K+1)B$. The computational cost is lowered from $O(N)$ to $O(kB)$ in this manner.

To dynamically update the representations $\mathbf{Z}^{(q)}$ and weights $\mathbf{W}^{(q)}$ in queues, we use a momentum coefficient $\alpha_i \in [0, 1)$: $\mathbf{Z}^{(q_i)*} = \alpha_i \mathbf{Z}^{(q_i)} + (1-\alpha_i)\,\mathbf{Z}^{(l)}$, $\mathbf{W}^{(q_i)*} = \alpha_i \mathbf{W}^{(q_i)} + (1-\alpha_i)\,\mathbf{W}^{(l)}$. We replace all $\mathbf{Z}^{(q_i)}$, $\mathbf{W}^{(q_i)}$ with $\mathbf{Z}^{(q_i)*}$, $\mathbf{W}^{(q_i)*}$ for the next batch.

Experiments

In this section, we describe the experimental setup used to evaluate the effectiveness of our proposed method. Experimental results demonstrate the effectiveness of our framework in comparison with different GNN backbones and datasets. We specifically aim to answer the following questions: **(RQ 1)** How effective is the proposed L2R-GNN framework for the graph classification task? **(RQ 2)** Could the proposed L2R-GNN alleviate different distribution shifts? **(RQ 3)** Could the proposed Bi-Level Training Algorithm alleviate the over-fitting issue? **(RQ 4)** Does the proposed reweighting mechanism work as designed and give some useful insights? **(RQ 5)** What are the effects of our proposed different components?

Baselines We compare our L2R-GNN with several representative state-of-the-art methods: GCN (Kipf and Welling 2016), GIN (Xu et al. 2018a), SGC (Wu et al. 2019), JKNet (Xu et al. 2018b), FactorGCN (Yang et al. 2020), PNA (Corso et al. 2020), TopKPool (Gao and Ji 2019), SAGPool (Lee, Lee, and Kang 2019), OOD-GNN (Li et al. 2021) and StableGNN (Fan et al. 2021).

Datasets We evaluate our method and baselines on synthetic and real-world datasets for complex and realistic graph distribution shifts: **Synthetic Datasets.** To validate L2R-GNN effectiveness with various distribution shifts, we generate synthetic datasets, allowing biased degree creation. Following GNN explanation works (Ying et al. 2019; Lin, Lan, and Li 2021), we focus on graph classification tasks with distribution shifts from training to testing datasets. We create a base subgraph for each graph, where each positive graph has a "wheel"-structured network motif and each negative graph has a motif chosen from four candidates: "star", "circle", "grid", and "diamond". The "wheel" motif is the causal structure determining the label.

Real-world Datasets. (1) Molecule and social datasets. Similar to previous works (Knyazev, Taylor, and Amer 2019), we consider three graph classification benchmarks: COLLAB, PROTEINS, and D&D. These datasets are split based on graph size. Methods are trained on smaller graphs and tested on unseen larger graphs. Specifically, COLLAB is a social dataset with 3 public datasets: High Energy Physics, Condensed Matter Physics, and Astro Physics. We train on graphs with nodes from 32 to 35 and test on graphs with nodes from 32 to 492. PROTEINS is a protein dataset. We train on graphs with nodes from 4 to 25 and test on graphs with nodes from 6 to 620. D&D is also a protein dataset.

	TOX21	BACE	BBBP	CLINTOX	HIV	ESOL
Metric	ROC-AUC (↑)					RMSE (↓)
GIN	70.4±0.9	73.8±3.1	67.9±1.4	87.4±2.8	75.8±1.2	1.14±0.08
GCN	72.7±0.6	77.6±1.5	66.8±1.2	88.6±2.2	76.2±1.2	1.12±0.04
SGC	71.8±1.2	70.7±1.4	62.7±1.8	76.4 ±2.0	67.5±1.3	1.59±0.06
PNA	69.1±0.8	74.9±1.9	64.5±1.3	80.6±2.3	76.2±1.8	0.98±0.07
JKNet	69.8±1.2	77.5±1.2	63.4±1.1	82.4±2.2	73.7±1.2	1.29±0.09
SAGPool	72.6±2.5	75.8±1.2	68.4±2.0	86.2±1.3	76.4±1.2	1.13±0.08
TopKPool	72.1±1.5	76.5±2.3	67.8±1.8	86.2±1.2	75.1±1.2	1.10±0.06
FactorGCN	56.2 ±2.8	68.9±1.5	55.1±1.6	65.7±2.6	57.5±1.9	3.12±0.17
OOD-GNN	76.2±1.3	78.3±1.8	68.4±2.8	89.1±1.6	78.2±1.2	0.94±0.06
StableGNN	74.8±1.9	79.2±2.4	68.1±2.6	88.4±1.9	76.5±1.8	0.96±0.04
L2R-GNN	**78.6±1.2**	**81.9±1.0**	**70.6±1.3**	**91.9±1.5**	**79.7±1.0**	**0.84±0.07**

Table 1: Performance of six Open Graph Benchmark (OGB) graph datasets.

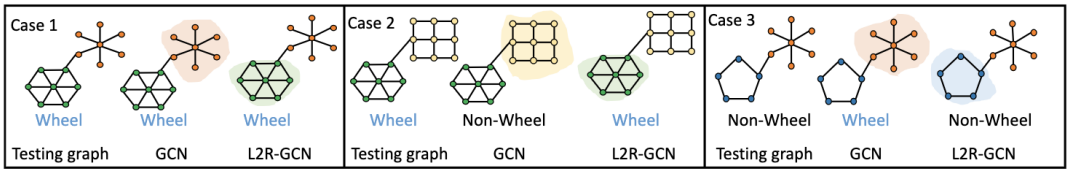
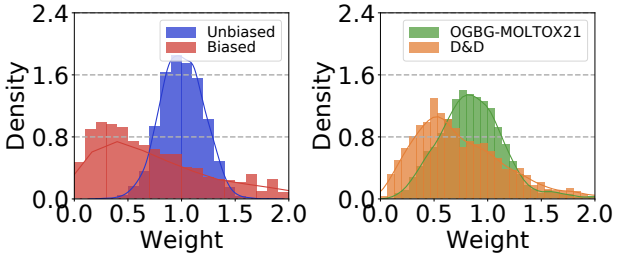


Figure 4: Case study of GCN and L2R-GNN in biased synthetic dataset. Shadow areas are the important subgraph calculated by the GNNExplainer.

	COLLAB	PROTEINS	D&D
GIN	56.3±3.5	74.4±2.5	68.9±4.1
GCN	64.0±2.8	74.8±2.7	71.7±3.3
SGC	54.9±4.1	72.6±2.1	64.2±3.8
PNA	58.7±4.3	71.9±2.9	70.5±2.4
JKNet	57.9±3.8	74.2±2.6	69.5±3.6
SAGPool	66.8±1.3	75.9±0.6	77.1±1.6
TopKPool	54.7±1.4	65.7±2.8	68.2±3.2
FactorGCN	52.3±1.8	62.4±4.3	55.2±2.4
OOD-GNN	66.9±1.5	77.1±1.1	79.1±1.3
StableGNN	67.3±1.4	76.5±0.9	78.7±1.6
L2R-GNN	**68.2±1.4**	**78.9±0.7**	**80.8±1.2**

Table 2: Performance of graph classification accuracy (%) with graph size distribution shifts, where the training and testing graphs are split by graph sizes. All methods are trained on small graphs and tested on larger graphs. Best results are indicated in bold.



(a) Synthetic dataset. (b) OGBG and D&D datasets.

Figure 5: The distribution of the learned graph weights on (a) unbiased and biased synthetic dataset and (b) two real-world datasets.

We train on graphs with nodes from 30 to 300 and test on graphs with nodes from 30 to 5, 748. (2)Open Graph Benchmark (OGB) (Hu et al. 2020). We consider OGBG-MOL∗ TOX21, BACE, BBBP, CLINTOX, HIV, and ESOL as six graph property prediction datasets from OGB with distribution shifts. Predicting target molecule properties is the graph classification task. We use scaffold splitting technique to separate graphs based on two-dimensional structural frameworks. This technique divides structurally diverse molecules into subsets, creating a more realistic and hard out-of-distribution generalization situation.

RQ1. Performance Comparison. Results on 6 OGB datasets are in Table 1. Datasets use scaffold splitting (Wu et al. 2018), dividing molecules by 2D structural frameworks, creating distribution shifts between train and test graphs. We could find that L2R-GNN outperforms other GNN models in all cases, effectively alleviating distribution shifts. L2R-GNN surpasses StableGNN and OOD-GNN, showing effectiveness of graph decorrelation method and bi-level optimization. L2R-GNN performs well on various tasks and dataset scales, indicating generality. L2R-GNN excels in out-of-distribution generalization, esp. for large-scale real-world graphs. Size generalization problem considered on real-world molecule and social datasets (COLLAB, PROTEINS, D&D), with train and test graphs split by size. Results in Table 2. L2R-GNN outperforms baselines, demonstrating best out-of-distribution generalization under size distribution shifts.

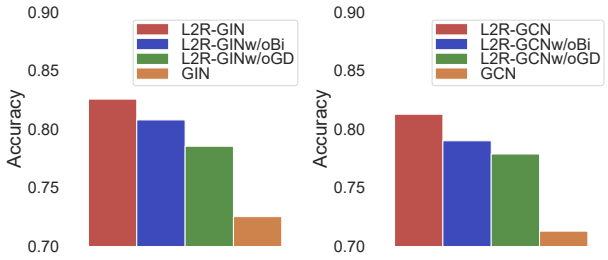


Figure 6: Ablation study of our L2R-GNN with (a) GIN and (b) GCN backbones.

RQ2. Different Distribution Shifts. To inject spurious correlation, $\mu * 100\%$ of "wheel" graphs have "star" motif added, and the remaining graphs have a non-causal motif chosen from 4 candidate motifs. For all nodes, node features are taken from the same uniform distribution. To create four spurious correlations for the training set, we set μ as $\{0.6, 0.7, 0.8, 0.9\}$. To answer RQ2, we conduct experiments on synthetic datasets with different distribution shifts. The results of GCN and GIN backbones under different correlation degree settings are shown in Figure 2. As spurious correlation degree increases, both GIN and GCN experience significant performance decline, suggesting spurious correlation greatly impacts GNNs' generalization performance, and larger distribution changes result in greater performance decline. Our L2R-GNN methods, implementing our framework on GCN and GIN backbones, increase graph categorization accuracy across various spurious correlation degree settings compared to GCN and GIN methods. The "wheel" motif, as described earlier, represents the label; thus, using this causal subgraph is the only way to improve performance. This demonstrates how our models can significantly reduce the impact of erroneous subgraph correlation. In all biased scenarios, our L2R-GNN outperforms backbones, proving it is a universal framework capable of fitting different GNN architectures.

RQ3. Bi-Level Optimization. To answer RQ3, we conduct experiments to analyze the model loss during training. We use *training* as a baseline, where we optimize W simultaneously with θ on training data without validation. We compare training with *first-order* and *second-order* approximates. Figures 3 shows the learning curves of training loss and validation loss on the D&D dataset of L2R-GNN . We can observe that the *training* gets stuck in the over-fitting issue attaining low training loss but high validation loss. For first-order and second-order, the difference between training and validation losses is significantly lower. It proves that the first-order approximation is adequate to prevent over-fitting and the bi-level optimization increases generalization ability.

RQ4. Reweighting Mechanism. We study the reweighting mechanism's contribution to robust graph representation learning via experiments on synthetic and real-world datasets. In synthetic datasets, $\mu * 100\%$ positive graphs have a "star" motif added, and the remaining positive and negative graphs have a non-causal motif chosen from 4 candidates. We collect graph weights in unbiased ($\mu = 0.25$) and biased ($\mu = 0.8$) datasets, as shown in Figure 5(a). Median weight in biased data is lower than in unbiased data, indicating L2R-GNN can identify noise graphs with spurious correlation. Weight variance in biased data is higher than in unbiased data, showing L2R-GNN 's reliable detection of noise graphs with spurious correlation. On real-world datasets D&D and OGBG-MOLTOX21, Figure 5(b) displays learned graph weight distribution, demonstrating non-trivial weights and varying distribution across datasets.

Case study. Using GNNExplainer (Ying et al. 2019), we visualize important subgraphs for GNN's prediction as shadow areas and compare GCN with L2R-GNN in Figure 4. Three cases demonstrate L2R-GNN effectiveness: *Case 1.* GNNExplainer (Ying et al. 2019) shows GCN assigns higher weights to "star" motif, while L2R-GNN focuses on "wheel" motif. GCN's accurate but unstable prediction may rely on spurious correlation, which is undesirable. *Case 2.* GCN ignores "wheel" motif due to spurious correlation, leading to incorrect prediction. L2R-GNN focuses on "wheel" motif, determining the true label. *Case 3.* Spurious correlation causes GCN to focus on "star" motif and make incorrect predictions. L2R-GNN 's decorrelation of subgraphs attributes more to prediction with "circle" motif.

RQ5. Component Effects We conduct an ablation study and hyper-parameter sensitivity analysis to understand component effects on performance. We compare L2R-GNN with: **L2R-GNNw/oBi**: L2R-GNN without bi-level optimization, optimizing W and θ simultaneously on training data without validation. **L2R-GNNw/oGD**: L2R-GNN without graph decorrelation module. Results in Figure 6 show L2R-GNN achieves the best performance, indicating each component contributes to effectiveness and robustness. Both components contribute to performance gain, complementing each other.

Conclusions

GNNs achieve state-of-the-art performance in tasks like molecular graph prediction, scene graph classification, and social network classification. We propose Learning to Reweight for Generalizable Graph Neural Network for OOD generalization of GNNs. Our novel nonlinear graph decorrelation method improves OOD generalization and outperforms previous methods in preventing over-reduced sample size. We propose a bi-level optimization-based stochastic algorithm for the L2R-GNN framework, enabling simultaneous learning of optimal example weights and GNN parameters, and avoiding over-fitting. Empirical results on synthetic and real-world datasets demonstrate L2R-GNN 's effectiveness.

Acknowledgements

This work was supported in part by Young Elite Scientists Sponsorship Program by CAST (2021QNRC001), National Natural Science Foundation of China (No. 62376243, U20A20387), the StarryNight Science Fund of Zhejiang University Shanghai Institute for Advanced Study (SN-ZJU-SIAS-0010), Project by Shanghai AI Laboratory (P22KS00111) and Program of Zhejiang Province Science and Technology (2022C01044).

References

Chen, S.; Chen, Z.; and Wang, D. 2021. Adaptive adversarial training for meta reinforcement learning. In *2021 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Chen, Y.; Wen, Z.; Fan, G.; Chen, Z.; Wu, W.; Liu, D.; Li, Z.; Liu, B.; and Xiao, Y. 2023a. MAPO: Boosting Large Language Model Performance with Model-Adaptive Prompt Optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 3279–3304.

Chen, Z.; Gai, S.; and Wang, D. 2019. Deep tensor factorization for multi-criteria recommender systems. In *2019 IEEE International Conference on Big Data (Big Data)*, 1046–1051. IEEE.

Chen, Z.; Ge, J.; Zhan, H.; Huang, S.; and Wang, D. 2021. Pareto Self-Supervised Training for Few-Shot Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13663–13672.

Chen, Z.; Gong, Y.; Yang, L.; Zhang, J.; Zhang, W.; He, S.; and Zhang, X. 2023b. Invariant Graph Neural Network for Out-of-Distribution Nodes. In *Proceedings of the 2023 15th International Conference on Machine Learning and Computing*, 192–196.

Chen, Z.; and Wang, D. 2021. Multi-Initialization Meta-Learning with Domain Adaptation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1390–1394. IEEE.

Chen, Z.; Wang, D.; and Yin, S. 2021. Improving cold-start recommendation via multi-prior meta-learning. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*, 249–256. Springer.

Chen, Z.; Xiao, T.; and Kuang, K. 2022. BA-GNN: On Learning Bias-Aware Graph Neural Network. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 3012–3024. IEEE.

Chen, Z.; Xu, Z.; and Wang, D. 2021. Deep transfer tensor decomposition with orthogonal constraint for recommender systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 4010–4018.

Corso, G.; Cavalleri, L.; Beaini, D.; Liò, P.; and Veličković, P. 2020. Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems*, 33: 13260–13271.

Fan, S.; Wang, X.; Shi, C.; Cui, P.; and Wang, B. 2021. Generalizing Graph Neural Networks on Out-Of-Distribution Graphs. In *arXiv preprint arXiv:2111.10657*.

Gai, S.; Chen, Z.; and Wang, D. 2021. Multi-modal meta continual learning. In *2021 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Gai, S.; Zhao, F.; Kang, Y.; Chen, Z.; Wang, D.; and Tang, A. 2019. Deep transfer collaborative filtering for recommender systems. In *PRICAI 2019: Trends in Artificial Intelligence: 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26-30, 2019, Proceedings, Part III 16*, 515–528. Springer.

Gao, H.; and Ji, S. 2019. Graph u-nets. In *international conference on machine learning*, 2083–2092. PMLR.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*.

Jiang, Y.; Chen, Z.; Kuang, K.; Yuan, L.; Ye, X.; Wang, Z.; Wu, F.; and Wei, Y. 2022. The Role of Deconfounding in Meta-learning. In *International Conference on Machine Learning*, 10161–10176. PMLR.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Knyazev, B.; Taylor, G. W.; and Amer, M. 2019. Understanding attention and generalization in graph neural networks. In *NeurIPS*.

Kuang, K.; Xiong, R.; Cui, P.; Athey, S.; and Li, B. 2020. Stable Prediction with Model Misspecification and Agnostic Distribution Shift. In *AAAI*.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. In *ICML*, 3734–3743. PMLR.

Li, H.; Wang, X.; Zhang, Z.; and Zhu, W. 2021. Ood-gnn: Out-of-distribution generalized graph neural network. *arXiv preprint arXiv:2112.03806*.

Lin, W.; Lan, H.; and Li, B. 2021. Generative Causal Explanations for Graph Neural Networks. In *ICML*.

Liu, Z.; Zhang, W.; Fang, Y.; Zhang, X.; and Hoi, S. C. 2020. Towards locality-aware meta-learning of tail node embeddings on networks. In *CIKM*.

Llorente, F.; Martino, L.; Read, J.; and Delgado, D. 2022. Optimality in Noisy Importance Sampling. *Signal Processing*, 108455.

Lv, Z.; Chen, Z.; Zhang, S.; Kuang, K.; Zhang, W.; Li, M.; Ooi, B. C.; and Wu, F. 2023a. Ideal: Toward high-efficiency device-cloud collaborative and dynamic recommendation system. *arXiv preprint arXiv:2302.07335*.

Lv, Z.; Zhang, W.; Zhang, S.; Kuang, K.; Wang, F.; Wang, Y.; Chen, Z.; Shen, T.; Yang, H.; Ooi, B. C.; et al. 2023b. DUET: A Tuning-Free Device-Cloud Collaborative Parameters Generation Framework for Efficient Device Model Generalization. In *Proceedings of the ACM Web Conference 2023*, 3077–3085.

Maclaurin, D.; Duvenaud, D.; and Adams, R. 2015. Gradient-based hyperparameter optimization through reversible learning. In *ICML*.

Martino, L.; Elvira, V.; and Louzada, F. 2017. Effective sample size for importance sampling based on discrepancy measures. *Signal Processing*, 131: 386–401.

Qu, L.; Zhu, H.; Zheng, R.; Shi, Y.; and Yin, H. 2021. Imgagn: Imbalanced network embedding via generative adversarial graph networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1390–1398.

Ren, M.; Zeng, W.; Yang, B.; and Urtasun, R. 2018. Learning to reweight examples for robust deep learning. In *ICML*.

Shen, Z.; Cui, P.; Liu, J.; Zhang, T.; Li, B.; and Chen, Z. 2020. Stable Learning via Differentiated Variable Decorrelation. In

Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2185–2193.

Tang, X.; Yao, H.; Sun, Y.; Wang, Y.; Tang, J.; Aggarwal, C.; Mitra, P.; and Wang, S. 2020. Investigating and Mitigating Degree-Related Biases in Graph Convoltuional Networks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1435–1444.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. In *ICLR*.

Wang, S.; Yang, J.; Chen, Z.; Yuan, H.; Geng, J.; and Hai, Z. 2020. Global and local tensor factorization for multi-criteria recommender system. *Patterns*, 1(2).

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying Graph Convolutional Networks. In *ICML*, 6861–6871. PMLR.

Wu, Q.; Zhang, H.; Yan, J.; and Wipf, D. 2022. Handling Distribution Shifts on Graphs: An Invariance Perspective. In *ICLR*.

Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; and Pande, V. 2018. MoleculeNet: a benchmark for molecular machine learning. *Chemical science*, 9(2): 513–530.

Xiao, T.; Chen, Z.; Guo, Z.; Zhuang, Z.; and Wang, S. 2022. Decoupled self-supervised learning for graphs. *Advances in Neural Information Processing Systems*, 35: 620–634.

Xiao, T.; Chen, Z.; Wang, D.; and Wang, S. 2021. Learning how to propagate messages in graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1894–1903.

Xiao, T.; Chen, Z.; and Wang, S. 2022. Representation matters when learning from biased feedback in recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2220–2229.

Xiao, T.; Chen, Z.; and Wang, S. 2023. Reconsidering Learning Objectives in Unbiased Recommendation: A Distribution Shift Perspective. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2764–2775.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018a. How Powerful are Graph Neural Networks? In *ICLR*.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.-i.; and Jegelka, S. 2018b. Representation learning on graphs with jumping knowledge networks. In *ICML*, 5453–5462. PMLR.

Yang, Y.; Feng, Z.; Song, M.; and Wang, X. 2020. Factorizable graph convolutional networks. In *NeurIPS*, volume 33, 20286–20296.

Ying, R.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *NeurIPS*.

Zhang, M.; Huang, S.; Li, W.; and Wang, D. 2022. Tree structure-aware few-shot image classification via hierarchical aggregation. In *European Conference on Computer Vision*, 453–470. Springer.

Zhang, M.; Yuan, J.; He, Y.; Li, W.; Chen, Z.; and Kuang, K. 2023. MAP: Towards Balanced Generalization of IID and OOD through Model-Agnostic Adapters. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11921–11931.