OFTEN: Graph Invariant Learning via Soft Environment Inference

Yang Liu[1✉], Zikun Zhang[1,2], Xiang Ao[1,2,3], Lingxiang Tian[1,2], and Qing He[1,2]

[1] Key Lab of AI Safety, Chinese Academy of Sciences (CAS), Key Lab of Intelligent Information Processing, Institute of Computing Technology, CAS, Beijing, China
{liuyang2023,aoxiang,heqing}@ict.ac.cn
[2] University of Chinese Academy of Sciences, CAS, Beijing, China
{zhangzikun20,tianlingxiang21}@mails.ucas.ac.cn
[3] Institute of Intelligent Computing Technology, CAS, Suzhou, China

Abstract. Graph invariant learning aims to acquire invariant node representations across various environments, achieving substantial success in addressing Out-of-Distribution (OOD) generalization for graph learning tasks. As obtaining environment splits on graphs is typically costly, most graph invariant learning methods heavily depend on inferring the underlying environments to learn invariant node representations. Due to the high heterogeneity of graph data without explicit source labels, existing environment inference methods cannot simultaneously satisfy the requirements of diversity and similarity. To address this challenge, we propose an approach called sOft environment inFerence with Test-timE adaptatioN, abbreviated as OFTEN, which enables us to perform graph invariant learning without any predefined environment split or partition information. The intuition is to enhance the diversity among environments while preserving the original graph topology. Extensive experiments on several graph OOD benchmarks demonstrate the consistent superiority of OFTEN across all settings.

Keywords: Graph Invariant Learning · Environment Inference

1 Introduction

The success of GNNs with empirical risk minimization (ERM) relies on the assumption that the testing and training data are identically drawn from the same distribution. However, distribution shifts [9] between testing and training data are usually inevitable due to data selection biases or unobserved confounders [4] that widely exist in real data. Under such circumstances, GNNs with ERM usually suffer from poor generalization performance due to the greedy exploitation of correlations among the training data, which are not stable under distribution shifts. Training a GNN algorithm with out-of-distribution (OOD) generalization ability is of paramount significance, especially in high-stake applications such as medical diagnosis [17] and financial analysis [15, 7].

✉ Corresponding author.

Graph invariant learning [10, 23] is one of the mainstream methods to solve the OOD generalization problem. The main idea is to exploit the causally invariant correlations(rather than varying spurious correlations) across multiple training environments, resulting in OOD optimal predictors. The effectiveness of such methods relies heavily on the quality of the environment splits. However, real-world graph data are frequently assembled by merging data from multiple sources without explicit source labels. The resultant unobserved heterogeneity makes it harder to determine which environment the sample belongs to.

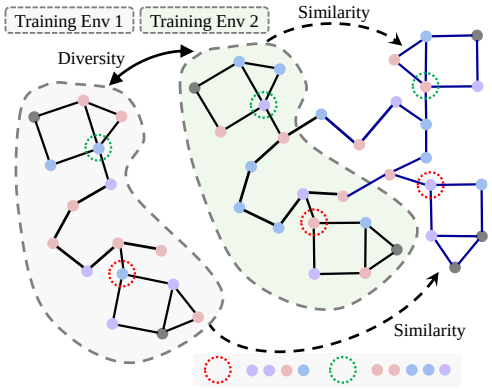


Fig. 1. Demonstration of Diversity & Similarity principles. The node label is based on the top/middle/bottom node of a house-like motif and the color is a spurious feature. The training environments 1 and 2 should be diverse in node colors but similar to the aggregated colors of the test set (nodes with blue edges). The dotted red circles include bottom nodes in a house connecting to a base node so they belong to the same class. Their aggregated colors remain invariant across the training and testing set.

Several environment inference methods [3, 13, 12] have been proposed to address the challenge of the unknown training environment. The primary idea behind these methods is to jointly optimize environment partition and invariant prediction. This approach can be easily applied to graph-level tasks like graph classification, where each graph is treated independently. In contrast, node-level OOD tasks cannot employ the same environment split solution since each node depends on its neighbors. For inferring node-level environments, EERM [20] is a pioneer work that uses multiple context explorers for graph structure editing, creating multiple virtual training environments.

The principles for designing environment splits and representation learning can be summarized as *Diversity & Similarity*, with *Diversity* pertaining to environment splits and *Similarity* to representation learning. Environmental splits should exhibit sufficient diversity to mitigate the adverse effects of spurious correlations. However, an exclusive pursuit of diversity may lead to deviations from the target distribution. Therefore, the latent representation space should be designed to be similar to the target embedding space for enhanced generalization.

As illustrated in Fig. 1, node color serves as a spurious feature, and node label is based on the location of a house-like motif[4]. The individual node colors vary across different environments to prevent spurious correlations, but the aggregated node colors for the same location remain consistent to promote better generalization. Generative OOD methods like EERM [20] achieve the *Diversity* principle by training multiple context generators to maximize risk variance but neglect the *Similarity* principle. In contrast, test-time adaptation methods like FLOOD [14] focus on realizing the *Similarity* goal by designing a flexible encoder updated in a self-supervised manner during the inference phase to align the latent feature space. However, the environments of FLOOD originate from random augmentation and thus do not satisfy the *Diversity* principle.

To simultaneously fulfill the aforementioned two principles, we propose a sOft environment inFerence approach with Test-timE adaptatioN (OFTEN) for graph OOD generalization. Different from existing environment inference solutions, we adopt a 'soft' approach to estimate the environment label for each node. Instead of assigning a scalar environment ID to each node, we train an assignment vector that reflects the node's involvement in each virtual environment. The node assignment weights are optimized to maximally violate the invariant principle for environment inference. Since nodes participate in all environments with varying weights, the neighbor set of a node remains unaffected by the environment split, overcoming the dependence challenge inherent in node-level graph learning tasks. In this way, the *Diversity* principle is achieved in environmental splits. For the *Similarity* principle, we regularize the model parameters during the test phase by penalizing the distance between the invariant representations of the training nodes and the target representations of the testing nodes.

2 Methodology

2.1 Problem Statement

We denote a graph as $\mathcal{G} = (\mathcal{V}, \mathbf{X}, \mathbf{A})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is the set of nodes, $\mathbf{X} \in \mathbb{R}^{N \times D}$ denotes node features, and $\mathbf{A} \in \{0,1\}^{N \times N}$ is an adjacency matrix representing the connections between nodes. Note that N and D represent the number of nodes and features, respectively. Besides, we denote the labels of the node as $\mathbf{Y} \in \{0,1\}^{N \times C}$ where C is the number of classes.

We focus on the node-level environment inference problem for the out-of-distribution node classification, where the target is to learn an environment assignment for each node in $\mathcal{V}$. Typical environment inference methods infer a scalar environment ID for each node. Different from them, our work treats the environment label as an M-dimensional vector, where the i-th element indicates the participation of the node in the i-th virtual environment. With these environment labels, the training nodes in $\mathcal{V}_{\text{train}}$ are re-weighted to construct M virtual environments that maximally violate the invariant learning principle.

[4] We adhere to the same labeling rule as in GOOD-CBAS [6].

We illustrate the pipeline of OFTEN on an example graph in Fig. 2, which consists of three parts: soft environment inference, graph invariant learning, and test-time adaptation.

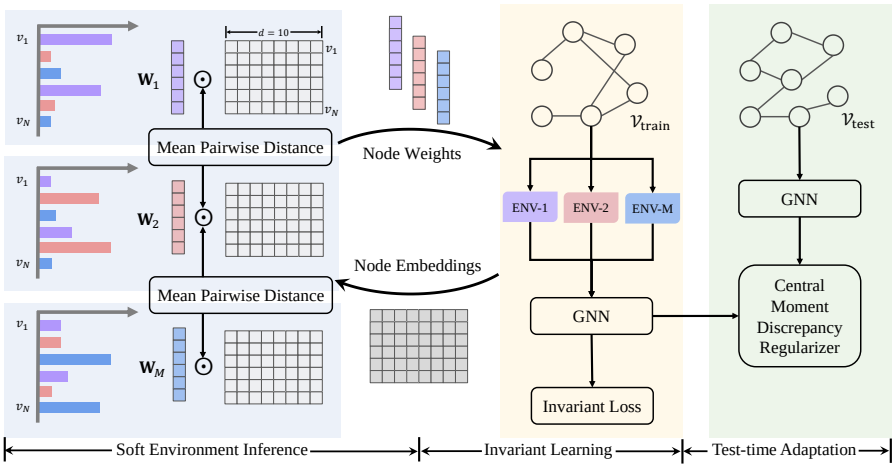


Fig. 2. The framework of OFTEN on an example graph.

2.2 Soft Environment Inference

The target of environment inference is to improve the diversity of training samples to facilitate invariant node representation learning and avoid spurious correlation. To achieve this, each node has a trainable assignment vector with each element $w_{v,e}$ being the weight of v in the environment e. The environment e is represented by normalized weighted node embeddings in Eq. (1).

$$\mathbf{H}_e = \frac{\sum_{v \in \mathcal{V}} w_{v,e} \mathbf{h}_v}{\sum_{v \in \mathcal{V}} w_{v,e}}, e \in \mathcal{E}, \tag{1}$$

where $w_{v,e}$ denotes the weight of node v in the environment e that satisfies $\sum_e w_{v,e} = 1$. $\mathbf{W}_e = [w_{1,e}, \ldots, w_{N,e}]^{\mathrm{T}}$ collects all the node weights in the virtual environment e.

The node weights $\mathbf{W} = [\mathbf{W}_1, \ldots, \mathbf{W}_M]$ are trained to maximize the variance[5] of all the environment representations and the Mean Pairwise Distance of all the possible environment pairs in Eq. (2).

$$\mathcal{L}_{\text{env}}(\mathbf{W}) = \text{Var}\left(\{\mathbf{H}_{e_1}, \ldots, \mathbf{H}_{e_M}\}\right) + \frac{2}{M(M-1)} \sum_{i<j} ||\mathbf{H}_{e_i} - \mathbf{H}_{e_j}||_2. \tag{2}$$

For each virtual environment e, the message-passing process is manipulated by $\mathbf{W}_e$. The aggregation step in the environment e is re-weighted by $\mathbf{W}_e$ and

[5] We compute the mean variance of each hidden dimension.

the update step for each node $v \in \mathcal{V}$ in layer l can be described as Eq.(3), where $\mathcal{N}(v)$ is the neighbor set of node v.

$$\mathbf{h}_{v,e}^{(l)} = \mathrm{UPD}^{(l)}\left(\mathbf{h}_{v,e}^{(l-1)}, \mathrm{AGG}^{(l)}\left(\left\{w_{u,e}\mathbf{h}_{u,e}^{(l-1)} : u \in \mathcal{N}(v)\right\}\right)\right). \tag{3}$$

The GNN with L layers are parameterized by θ and the obtained node embedding from the environment e is $\mathbf{h}_{v,e}^{(L)}$. In each virtual environment, θ is trained to minimize the weighted cross-entropy loss in Eq. (4).

$$\mathcal{R}_e(\theta) = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}} w_{v,e}\ell\left(\mathrm{MLP}(\mathbf{h}_{v,e}^{(L)}), \mathbf{Y}_v\right). \tag{4}$$

OFTEN employs the invariant loss from VREx as Eq. (5) to minimize the risk variance across all the environments. $\mathcal{L}_{\text{env}}$ and $\mathcal{L}_{\text{inv}}$ are optimized alternately as detailed in Section 2.4.

$$\mathcal{L}_{\text{inv}}(\theta) = \mathrm{Var}\left(\{\mathcal{R}_1(\theta), \ldots, \mathcal{R}_M(\theta)\}\right) + \sum_{e=1}^{M}\mathcal{R}_e(\theta). \tag{5}$$

2.3 Test Time Adaptation

After environment inference and invariant learning, we obtain invariant representations of the training nodes $\mathbf{Z}_{\text{train}} = f_\theta\left(\mathcal{V}_{\text{train}}, \mathbf{A}\right)$ and infer the representations of testing nodes $\mathbf{Z}_{\text{test}} = f_\theta\left(\mathcal{V}_{\text{test}}, \mathbf{A}\right)$. To mitigate the distribution shift between the training distribution and testing distribution, we adopt Central Moment Discrepancy as a regularizer during the test phase. The re-weighted node representations from each virtual environment $\mathrm{diag}(\mathbf{W}_e)\cdot\mathbf{Z}_{\text{train}}$ are refined to be close to the test distribution $\mathbf{Z}_{\text{test}}$ as shown in Eq. (6).

$$\mathcal{L}_{\text{reg}}(\theta) = \frac{1}{M}\sum_{e=1}^{M}\mathcal{D}\left(\mathrm{diag}(\mathbf{W}_e)\cdot\mathbf{Z}_{\text{train}}, \mathbf{Z}_{\text{test}}\right), \tag{6}$$

where the diag operator changes the N-dimensional column vector into an $N\times N$ diagonal matrix and the Central Moment Discrepancy $\mathcal{D}$ is used to measure the distance between two distributions $\mathbf{Z}_1$ and $\mathbf{Z}_2$ as defined in Eq.(7).

$$\mathcal{D}\left(\mathbf{Z}_1, \mathbf{Z}_2\right) = \frac{1}{|b-a|}\|\mathbb{E}(\mathbf{Z}_1) - \mathbb{E}(\mathbf{Z}_2)\|_2 + \sum_{k=2}^{K}\frac{1}{|b-a|^k}\|c_k\left(\mathbf{Z}_1\right) - c_k(\mathbf{Z}_2)\|_2, \tag{7}$$

where $c_k(\mathbf{Z}) = \mathbb{E}(\mathbf{Z} - \mathbb{E}(\mathbf{Z}))^k$ is k-th order moment and a, b denotes the joint distribution support of the distributions ($b - a = 1$ in our work). In practice, only a limited number of moments is usually included (K=5 in our work).

2.4 Overall Algorithm and Complexity Analysis

Given a set of training nodes $\mathcal{V}_{\text{train}}$ and a set of testing nodes $\mathcal{V}_{\text{test}}$, with the GNNs initialized, we first train the node weights w.r.t. Eq. (2) for N_{init} epochs.

Then in the invariant learning loop of N_{epoch} epochs, environment inference and invariant learning are optimized alternately. The node assignment weights are optimized for N_{split} times and the GNN parameter θ is updated for once. After that, the model parameter θ is refined for N_{test} iterations to minimize the Central Moment Discrepancy between training nodes and testing nodes. Finally, the predictions of testing nodes are returned.

Consider a graph with N nodes and E edges, the average degree is $\bar{d}$. GNN with L layers compute embeddings in time and space $O(NL\bar{d}^2)$. OFTEN estimates M virtual environments computations per update step (M for each training environment) plus a prediction step. As the update step during the test phase is similar to model training, the overall time complexity is linear to the scale of the graph $O(MNL\bar{d}^2)$.

3 Experiments

3.1 Experimental Setup

Datasets We adopt four node classification datasets from GOOD [6], a graph out-of-distribution benchmark, to verify the environment inference ability of OFTEN. The statistics of the datasets are shown in Table 1.

Table 1. Statistics of datasets.

Dataset	#Node	#Edge	#Class	#Feat	Domain
CBAS	700	3,962	4	4	Color
WebKB	617	1,138	5	1,703	University
Cora	19,793	126,842	70	8,710	Word/Degree
Arxiv	169,343	1,166,243	40	128	Time/Degree

Compared Methods We utilize OFTEN to generate virtual environments for graph invariant learning methods (IRM [1], VREx [8]) and denote them as IRM-OFTEN and VREx-OFTEN. We use EIIL [3] to generate environmental splits for IRM and VREx. Besides, OFTEN is also applied on SRGNN [24] and derives SRGNN-OFTEN to be compared with original SRGNN. Other compared methods include robust optimization GroupDRO [16], domain generalization DANN [5], DeepCoral [18], graph data augmentation Mixup [19], and graph OOD methods (EERM [20], IS-GIB [22], INL [11], FLOOD [14]).

Implementation Details Our work is implemented in Pytorch 1.10.1 with Python 3.8.17, and all the experiments are conducted on a Ubuntu 22.04.2 server with 72 cores and 512GB memory. IRM, VREx, GroupDRO, Mixup, SRGNN, EERM, DANN, and DeepCoral are based on the implementation of GOOD [6]. IS-GIB, INL, and FLOOD are based on the source code provided by the authors. We report the average value of 5 runs. For our work, we adopt GCN as the backbone, the same settings as GOOD does, for a fair comparison.

Hyper-parameter Settings The parameters of GCNs are optimized with Adam optimizer with learning rate lr=1e-3. Hyper-parameter tuning is conducted using grid search for most methods. In our work, $N_{\text{init}} = 15$, $N_{\text{split}} = 5$, $N_{\text{test}} = 5$, $d = 300$. For all situations unless specifically mentioned, $M = 8$ for covariate shift, $M = 2$ for concept shift, and $\alpha = 1e - 6$ for both shifts. For GOOD-CBAS and GOOD-WebKB, $\alpha = 1e - 4$ for both covariate and concept shifts. For GOOD-Cora with covariate shift on degree domain split, $M = 10$ and $\alpha = 1e - 4$. For GOOD-Cora with concept shift, $\alpha = 1e - 4$.

Table 2. Performance Comparison of Node Classification under Covariate Shift

Dataset	Cora		Arxiv		CBAS	WebKB
Domain	Word	Degree	Time	Degree	Color	University
ERM	64.58±0.27	55.94±1.05	70.27±0.47	58.43±0.28	67.14±7.65	14.55±5.88
Mixup	64.43±0.24	56.22±0.79	70.91±0.24	56.83±0.28	71.43±1.17	20.90±2.45
GroupDRO	64.56±0.11	56.07±0.18	70.15±0.59	58.35±0.24	72.86±1.17	12.70±0.00
DeepCoral	64.30±0.75	55.90±0.25	70.73±0.19	58.57±0.13	72.38±2.93	17.19±3.74
DANN	64.75±0.20	56.15±0.40	70.42±0.29	58.39±0.15	75.71±0.00	20.87±3.20
EERM	62.88±0.01	56.54±0.09	OOM	OOM	71.91±2.43	21.69±8.33
IS-GIB	65.23±0.46	54.16±0.48	70.45±0.31	57.64±0.18	74.29±0.90	12.70±1.01
INL	64.67±0.31	55.76±0.71	70.54±0.56	58.52±0.26	72.66±1.19	14.29±3.46
FLOOD	65.18±0.31	55.37±0.56	70.89±0.42	57.39±0.21	76.32±0.39	15.11±0.49
IRM	64.67±0.31	55.76±0.71	70.54±0.56	58.52±0.26	72.66±1.19	14.29±3.46
-OFTEN	64.87±0.44	**56.59±0.63**	**71.24±0.33**	**58.99±0.17**	77.62±0.67	19.21±2.81
VREx	64.34±0.25	56.36±0.14	70.38±0.02	58.33±0.33	61.43±2.02	9.26±0.75
-OFTEN	64.97±0.57	56.54±0.18	71.05±0.34	58.92±0.18	78.29±1.78	**22.86±4.67**
SRGNN	64.79±0.13	56.33±0.15	70.62±0.37	58.31±0.21	75.24±2.93	12.17±3.33
-OFTEN	**65.30±0.36**	56.51±0.59	71.15±0.22	58.91±0.17	**78.86±1.67**	16.98±1.79

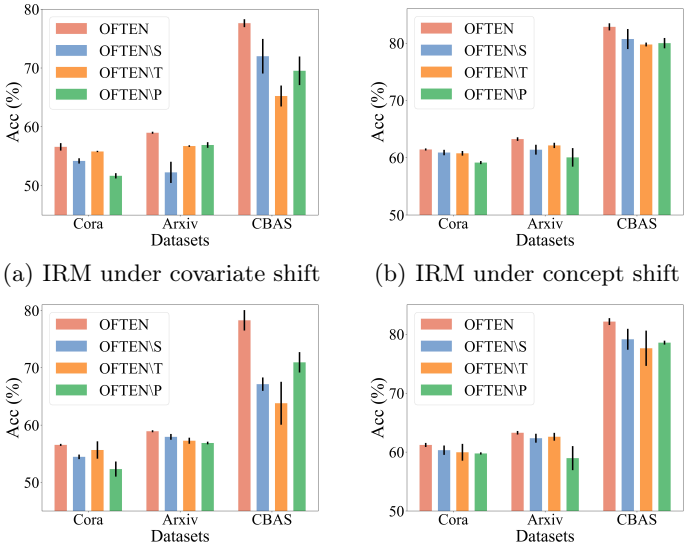
3.2 Performance Comparison

We evaluate the performance of OFTEN on OOD node classification tasks. The Accuracy scores are reported in Table 2. We have the following observations.

Firstly, OFTEN effectively conducts soft environment splits without any provided information. IRM-OFTEN, VREx-OFTEN, and SRGNN-OFTEN consistently achieve better or comparable performance compared with models trained with EIIL splits. In these soft splits, each node participates in various virtual environments with different weights, which are optimized to maximize the variance and pairwise distance. Instead of assigning one node to a specific environment, OFTEN avoids perturbing the original graph topology, resulting in improved generalization performance.

Secondly, OFTEN demonstrates superior OOD generalization ability compared to other graph OOD methods. EERM can construct multiple training environments and OFTEN outperforms EERM by aligning the latent space with

the test distribution in the test-time adaptation. In contrast, EERM deploys the invariant model without further adaptation. Notably, EERM achieves the second-best performance in WebKB because the graph of WebKB is small and sparse and context generators in EERM can provide more useful information for OOD generalization.



(a) IRM under covariate shift (b) IRM under concept shift

(c) VREx under covariate shift (d) VREx under concept shift

Fig. 3. Ablation study of OFTEN. OFTEN\S replaces the soft splits with hard splits. OFTEN\T removes the test-time adaptation. OFTEN\P randomizes the node weights.

3.3 Ablation Study

We evaluate the contributions of two key modules of OFTEN, namely soft environment inference and test-time adaptation, by replacing or removing each module respectively. We report the ablation results in Fig. 3. The full model OFTEN always achieves the best scores compared with the three variants, OFTEN\S, OFTEN\T, and OFTEN\P, indicating that each module is necessary for OOD generalization.

OFTEN\S replaces the soft splits with hard splits and each node is assigned to one virtual environment with the largest weight. The performance drop indicates that soft splits are more benefical to invariant learning since they re-weight the node features and do not change the topology. OFTEN\T removes the test-time adaptation module thus affects the generalization ability of the invariant GNN. OFTEN\P removes the Mean Pairwise Distance loss and randomizes the node weights. The performance of OFTEN\P are the worst in most cases and we can conclude that OFTEN learns meaningful soft environment splits from the node weights trained to maximize the variance and distance.

4 Related Work

Environment Inference for Invariant Learning. Out-of-Distribution (OOD) generalization addresses the challenging setting where the testing distribution is unknown and different from that of the training. Invariant learning methods, typified by invariant risk minimization(IRM, [1]), have shown promising results in tackling OOD generalization problem. However, they rely on existing environment splits. Recent works [3, 12] on environment inference learn invariance without environment indexes where the dataset is assembled by merging data from multiple environments. The methods mentioned above conduct environment partition for current invariant learning models for out-of-distribution generalization. Different from them, our major concern is the node-level environment inference problem on graphs. The principle is applicable to graph-level tasks since each graph is independent but the environment inference is more complicated for node-level tasks due to the complex dependence of the nodes.

Out-of-distribution Generalization on Graphs. Distribution shifts on graphs can appear in a variety of forms such as attributes and structures, making it difficult to identify the invariance. Moreover, environment construction or inference, which are often required by OOD methods on Euclidean data, can be highly expensive to obtain for graphs due to the structural relationship of the nodes. GIL [10], DIR [21], MoleOOD [23], and CIGA [2] investigate graph-level tasks for OOD generalization. Different from them, we consider the OOD problem of node-level tasks on graphs. EERM [20] and FLOOD [14] generate environments from context generators and data augmentation respectively. Differently, OFTEN learns an environment assignment vector for each node which increases the diversity in a "soft" way. In contrast, INL assigns each node to one environment in a "hard" way.

5 Conclusion

In this study, we investigated the issue of environment inference for graph invariant learning and proposed a new solution, OFTEN, which performs environment splits of graph-structured data for node classification tasks. Our framework aims to develop an automatic environment split for graph invariant learning. By using soft environment inference, OFTEN re-weighted node embeddings to maximize the pairwise distance among virtual environments. To further adapt the invariant model to the test distribution, OFTEN conducts test-time adaptation by minimizing the discrepancy between the observed training and testing distribution. Our work demonstrates the possibility of performing invariant learning on graph-structured data without any given environment partition information.

Acknowledgments. The research work is supported by National Key R&D Plan No. 2022YFC3303302, the National Natural Science Foundation of China under Grant No. 62406307, No. 62476263, the China Postdoctoral Science Foundation under Grant No. 2023M743567 and the Postdoctoral Fellowship Program of CPSF under Grant No. GZB20240761. Xiang Ao is also supported by Beijing Nova Program 20230484430.

References

1. Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint arXiv:1907.02893 (2019)
2. Chen, Y., Zhang, Y., Bian, Y., et al, Cheng, J.: Learning causally invariant representations for out-of-distribution generalization on graphs. In: NeurIPS (2022)
3. Creager, E., Jacobsen, J.H., Zemel, R.: Environment inference for invariant learning. In: ICML. pp. 2189–2200. PMLR (2021)
4. Cui, P., Athey, S.: Stable learning establishes some common ground between causal inference and machine learning. Nature Machine Intelligence **4**(2), 110–115 (2022)
5. Ganin, Y., Ustinova, E., et al, Lempitsky, V.: Domain-adversarial training of neural networks. The journal of machine learning research **17**(1), 2096–2030 (2016)
6. Gui, S., Li, X., Wang, L., Ji, S.: Good: A graph out-of-distribution benchmark. Advances in Neural Information Processing Systems (2022)
7. Huang, M., Liu, Y., Ao, X., Li, K., Chi, J., Feng, J., Yang, H., He, Q.: Auc-oriented graph neural network for fraud detection. In: WWW (2022)
8. Krueger, D., Caballero, E., et al, Courville, A.: Out-of-distribution generalization via risk extrapolation (rex). In: ICML. pp. 5815–5826. PMLR (2021)
9. Lee, Y., et al, Yao, H., Liang, P., Finn, C.: Surgical fine-tuning improves adaptation to distribution shifts. NeurIPS (2023)
10. Li, H., Zhang, Z., Wang, X., Zhu, W.: Learning invariant graph representations for out-of-distribution generalization. NeurIPS (2022)
11. Li, H., Zhang, Z., Wang, X., Zhu, W.: Invariant node representation learning under distribution shifts with multiple latent environments. ACM TOIS (2023)
12. Lin, Y., Zhu, S., Tan, L., Cui, P.: Zin: When and how to learn invariance without environment partition? NeurIPS **35**, 24529–24542 (2022)
13. Liu, J., Hu, Z., Cui, P., Li, B., Shen, Z.: Heterogeneous risk minimization. In: International Conference on Machine Learning. pp. 6804–6814. PMLR (2021)
14. Liu, Y., Ao, X., Feng, F., Ma, Y., Li, K., Chua, T.S., He, Q.: Flood: A flexible invariant learning framework for out-of-distribution generalization on graphs. In: KDD. pp. 1548–1558 (2023)
15. Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., He, Q.: Pick and choose: a gnn-based imbalanced learning approach for fraud detection. In: WWW (2021)
16. Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In: International Conference on Learning Representations (2020)
17. Shamsi, A., et al, Srinivasan, D.: An uncertainty-aware transfer learning-based framework for covid-19 diagnosis. IEEE TNNLS **32**(4), 1408–1417 (2021)
18. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: Computer Vision–ECCV 2016 Workshops. pp. 443–450. Springer (2016)
19. Wang, Y., Wang, W., Liang, Y., Cai, Y., Hooi, B.: Mixup for node and graph classification. In: Proceedings of the Web Conference 2021. pp. 3663–3674 (2021)
20. Wu, Q., Zhang, H., Yan, J., Wipf, D.: Handling distribution shifts on graphs: An invariance perspective. In: ICLR (2022)
21. Wu, Y.X., Wang, X., Zhang, A., He, X., Chua, T.S.: Discovering invariant rationales for graph neural networks. In: ICLR (2022)
22. Yang, L., Zheng, J., et al, Zhang, W., Cui, B.: Individual and structural graph information bottlenecks for out-of-distribution generalization. IEEE TKDE (2023)
23. Yang, N., Zeng, K., Wu, Q., Jia, X., Yan, J.: Learning substructure invariance for out-of-distribution molecular representations. In: NeurIPS (2022)
24. Zhu, Q., Ponomareva, N., Han, J., Perozzi, B.: Shift-robust gnns: Overcoming the limitations of localized graph training data. NeurIPS **34**, 27965–27977 (2021)