Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments

HAOYANG LI and ZIWEI ZHANG, Department of Computer Science and Technology, Tsinghua University, China
XIN WANG and WENWU ZHU, Department of Computer Science and Technology, BNRist, Tsinghua University, China

Node representation learning methods, such as graph neural networks, show promising results when testing and training graph data come from the same distribution. However, the existing approaches fail to generalize under distribution shifts when the nodes reside in multiple latent environments. How to learn invariant node representations to handle distribution shifts with multiple latent environments remains unexplored. In this article, we propose a novel Invariant Node representation Learning (INL) approach capable of generating invariant node representations based on the invariant patterns under distribution shifts with multiple latent environments by leveraging the invariance principle. Specifically, we define invariant and variant patterns as ego-subgraphs of each node and identify the invariant ego-subgraphs through jointly accounting for node features and graph structures. To infer the latent environments of nodes, we propose a contrastive modularity-based graph clustering method based on the variant patterns. We further propose an invariant learning module to learn node representations that can generalize to distribution shifts. We theoretically show that our proposed method can achieve guaranteed performance under distribution shifts. Extensive experiments on both synthetic and real-world node classification benchmarks demonstrate that our method greatly outperforms state-of-the-art baselines under distribution shifts.

CCS Concepts: • **Computing methodologies** → **Neural networks**; **Learning latent representations**; • **Mathematics of computing** → **Graph algorithms;**

Additional Key Words and Phrases: Graph neural networks, node representation learning, distribution shift

ACM Reference format:
Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu. 2023. Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. *ACM Trans. Inf. Syst.* 42, 1, Article 26 (August 2023), 30 pages.
https://doi.org/10.1145/3604427

This work was supported in part by the National Key Research and Development Program of China No. 2020AAA0106300, National Natural Science Foundation of China (Nos. 62250008, 62222209, 62102222, 61936011), Beijing National Research Center for Information Science and Technology (BNRist) under Grant No. BNR2023RC01003, and Beijing Key Lab of Networked Multimedia, China National Postdoctoral Program for Innovative Talents No. BX20220185, China Postdoctoral Science Foundation No. 2022M711813.
Authors' addresses: H. Li and Z. Zhang, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China; emails: lihy18@mails.tsinghua.edu.cn, zwzhang@tsinghua.edu.cn; X. Wang (corresponding author) and W. Zhu (corresponding author), Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing 100084, China; emails: {xin_wang, wwzhu}@tsinghua.edu.cn.

1046-8188/2023/08-ART26 $15.00
https://doi.org/10.1145/3604427

1 INTRODUCTION

Graph-structured data is ubiquitous in the real world, e.g., social networks [22], knowledge graphs [61], biology networks [5], chemical molecules [80], and so on. Learning node representation is critical for various graph analytical tasks such as node classification [38] and link prediction [67]. Especially, **graph neural networks (GNNs)** [38, 75, 81] have shown great successes in learning effective node representations and handling applications from various fields [14, 35, 55, 70, 84, 94, 97, 100].

Despite their successes, the existing node representation learning approaches typically assume that the testing and training graph data are drawn from the same distribution, namely the node features and graph structures of labeled training nodes and testing nodes follow similar patterns. Under this assumption, the node representation learning methods can naturally generalize to unseen testing nodes. However, this assumption can be easily violated in real-world graphs, since nodes always reside in multiple latent environments where distribution shifts widely exist between multiple latent environments of training and testing data induced by complex underlying data generation mechanism [6]. For example, in protein-protein interaction graphs, the distributions of protein features/interactions (i.e., input data) and their functions (i.e., labels) exist significant changes between different species that the proteins come from (i.e., environments) [15]. In citation networks, the papers' citations (i.e., input data) and their subject topics (i.e., labels) are strongly affected by the publication time (i.e., environments) [33]. There exist increasing evidences suggesting that most node representation learning approaches are vulnerable to distribution shifts [33, 78, 79] and fail to achieve **out-of-distribution (OOD)** generalization. If the models capture the variant correlations across different environments rather than focus on invariant patterns of the truly predictive properties in multiple latent environments, they will inevitably fail under distribution shifts, hindering their applications in real-world graphs, especially for high-stake applications such as molecular prediction [80], financial analysis [85], medical diagnosis [47], drug repurposing [32], and so on.

In this work, we study learning invariant node representations to handle distribution shifts with multiple latent environments, which remains unexplored and poses great challenges as follows:

- First, nodes in the graph are connected by structures and cannot be modeled as independent samples for predictions. Distribution shifts can happen on both node features and graph structures, leading to complex invariant and variant patterns. How to define and identify these patterns to capture sufficiently predictive information is non-trivial.
- Second, environment labels for nodes are usually unavailable or prohibitively expensive to collect. How to infer the environment labels, which is critical for designing invariant learning methods, is also challenging, since the environments of different nodes are also highly entangled.
- Last but not least, even with the inferred environment labels of nodes, it requires tailored designs to learn invariant node representations capable of generalization under distribution shifts with theoretical guarantees.

To tackle these challenges, we propose **Invariant Node representation Learning** (**INL**) approach capable of learning invariant node representations under distribution shifts with multiple latent environments and achieve theoretically grounded generalization performance. The framework of **INL** is shown in Figure 1. In particular, we take a local view and define invariant patterns as ego-subgraphs, i.e., subgraphs of the L-order ego-graph of each node, and identify these ego-subgraphs through jointly considering node features and graph structures. Then, we use the variant ego-subgraphs, i.e., the complement of invariant ego-subgraphs, to infer environment labels by proposing a contrastive modularity-based graph clustering method. The variant ego-subgraphs capture correlative but not truly predictive patterns with node labels under

distribution shifts and therefore contain discriminative information to infer environment labels of nodes. Finally, we propose to optimize the maximal invariant pattern criterion given the identified invariant ego-subgraphs and inferred environments to produce invariant node representations. We theoretically show that **INL** can achieve guaranteed generalization performance by finding a maximal invariant pattern. We conduct extensive experiments on both synthetic datasets and real-world benchmarks for the node classification task. The results show that **INL** achieves substantial performance gains on the unseen testing nodes compared with various state-of-the-art baselines. Our contributions are summarized as follows:

- We propose a novel Invariant Node representation Learning (**INL**) approach to learn invariant node representations capable of OOD generalization under distribution shifts. To the best of our knowledge, we are the first to study invariant node representation learning with multiple latent environments.
- We design a contrastive modularity-based graph clustering method to infer the environment labels of nodes for the graph with complex multiple latent environments.
- We propose a maximal invariant pattern criterion to learn node representations. We theoretically show that by finding maximal invariant ego-subgraphs, **INL** can achieve guaranteed OOD generalization performance under distribution shifts.
- Extensive experimental results demonstrate the effectiveness of **INL** on various synthetic and benchmark datasets for the node classification task under distribution shifts.

We introduce the notations and preliminaries in Section 2. In Section 3, we describe the problem formulation and the details of our proposed **INL**. We present the experimental results in Section 4, including quantitative comparisons on both synthetic and real-world datasets, complexity analysis, ablation studies, hyper-parameter sensitivity, and so on. Subsequently, some related works are reviewed in Section 5. We conclude this work in Section 6.

2 NOTATIONS AND PRELIMINARIES

2.1 Notations

Consider a graph $G = (V, E)$, the node feature matrix $X = \{x_v | v \in V\} \in \mathbb{R}^{|V| \times F}$ (where F denotes the node feature dimension), and labels $Y = \{y_v | v \in V\}$. The adjacency matrix is denoted as $A = \{a_{v,v'} | v, v' \in V\} \in \mathbb{R}^{|V| \times |V|}$, where $a_{v,v'} = 1$ means there exists an edge connecting node v and v', and $a_{v,v'} = 0$ otherwise. We assume the nodes V are collected from multiple environments, i.e., $V = \{V^e\}_{e \in \text{supp}(\mathcal{E}_{tr})}$, where V^e denotes the nodes from environment e, $\text{supp}(\mathcal{E}_{tr})$ is the support of the environmental variable. We use $\mathbf{v}$ and $\mathbf{y}$ to denote the random variables of node and label, respectively. We summarize the key notations of this article and the corresponding descriptions in Table 1.

2.2 Preliminaries

Recently, invariant learning has received surging attention to enable generalizing to distribution shifts, i.e., ***out-of-distribution (OOD)*** *generalization*. It aims to exploit the invariant relationships between the input data and labels across distribution shifts while filtering out the variant spurious correlations.[1] Following the invariant learning literature [2, 4, 11, 40, 42, 64], we formulate the

[1]Although the variant spurious correlations can be potentially useful for predictions in some environments, such correlations are not stable and can change across different environments. It is infeasible to judge whether the variant spurious correlations are still correct or not when the model is deployed in unknown testing environments with distribution shifts. Therefore, for achieving good OOD generalization rather than trivially overfitting the training data, the key idea of invariant learning is to learn invariant models for guaranteed generalization under distribution shifts.

Table 1. Notations

Notation	Description
$G = (V, E)$	The input graph G with node set V and edge set E
X, A, Y	The node feature matrix, the adjacency matrix, and the label vector
$G_v, \mathbf{G_v}$	An instance and the random variable of node v's ego-graph
$G_v^I = \Psi(G_v)$	An instance of the invariant ego-subgraph and the invariant ego-subgraph generator
Ψ^*	The optimal invariant ego-subgraph generator
X_v, A_v	The local node feature matrix and the adjacency matrix of ego-graph G_v
$G_v^S = G_v \backslash G_v^I$	An instance of the variant ego-subgraph
$\mathbf{G_v}, \mathbf{v}, \mathbf{Y}, \mathbf{y}$	The random variable of ego-graph, node, label vector, node label
X_v^I / X_v^S	The local node feature matrix of the invariant/variant ego-subgraph G_v
A_v^I / A_v^S	The local adjacency matrix of the invariant/variant ego-subgraph G_v
$\mathbf{Z}^I$	The invariant node representations
$\mathcal{N}_v$	The node v's L-hop neighbors
K	The number of the ground-truth environments
$\mathcal{E} / \mathcal{E}_{tr}$	A random variable on indices of all/training environments
$\mathcal{E}_{infer}$	A random variable on indices of the inferred environments
$\lvert\mathcal{E}_{infer}\rvert$	The number of the inferred environments
C	The cluster assignment matrix
C_v	The one-hot vector indicating the environment of node v with dimensionality $\lvert\mathcal{E}_{infer}\rvert$
e	An instance of environment
$\mathbb{G}, \mathbb{Y}$	The graph space and label space
f	The predictor from $\mathbb{G}$ to $\mathbb{Y}$
w	The classifier from $\mathbb{R}^d$ to $\mathbb{Y}$
h	The representation learning function from $\mathbb{G}$ to $\mathbb{R}^d$
g	The representation learning function for invariant ego-subgraph
$\mathcal{I}_{\mathcal{E}}$	The invariant ego-subgraph generator set with respect to $\mathcal{E}$
ℓ	The loss function

problem of learning invariant node representations capable of generalizing to distribution shifts, i.e., **out-of-distribution (OOD)** generalized node representation learning, as:

Problem 1. *Let $\mathcal{E}$ denote the random variable on indices of **all** possible environments of nodes V. The goal is to find an optimal predictor $f^*(\cdot)$ mapping nodes to their labels that performs well on all environments:*

$$f^*(\cdot) = \arg\min_{f} \sup_{e \in \text{supp}(\mathcal{E})} \mathcal{R}(f|e), \tag{1}$$

where $\mathcal{R}(f|e)$ is the risk of the predictor f on the nodes that belong to environment e. Equation (1) encourages to learn the predictor whose performance on the worst-case environment is optimal, where such min-max optimality with respect to unseen test environments is proved to satisfy the OOD generalization in the invariant learning literature [3, 40, 64]. We further decompose $f(\cdot) = w \circ h$, where $h(\cdot) : \mathbb{G} \to \mathbb{R}^d$ is the representation learning function, $\mathbb{G}$ is the graph space, d is the dimensionality, and $w(\cdot) : \mathbb{R}^d \to \mathbb{Y}$ is the classifier.

Note that $\text{supp}(\mathcal{E}_{tr}) \subset \text{supp}(\mathcal{E})$. Distribution shifts indicate that $P^e(\mathbf{v}, \mathbf{y}) \neq P^{e'}(\mathbf{v}, \mathbf{y}), e \in \text{supp}(\mathcal{E}_{tr}), e' \in \text{supp}(\mathcal{E}) \setminus \text{supp}(\mathcal{E}_{tr})$, i.e., the joint distribution of node and label is different in training and testing data. The testing nodes are not available in the training stage, meaning that

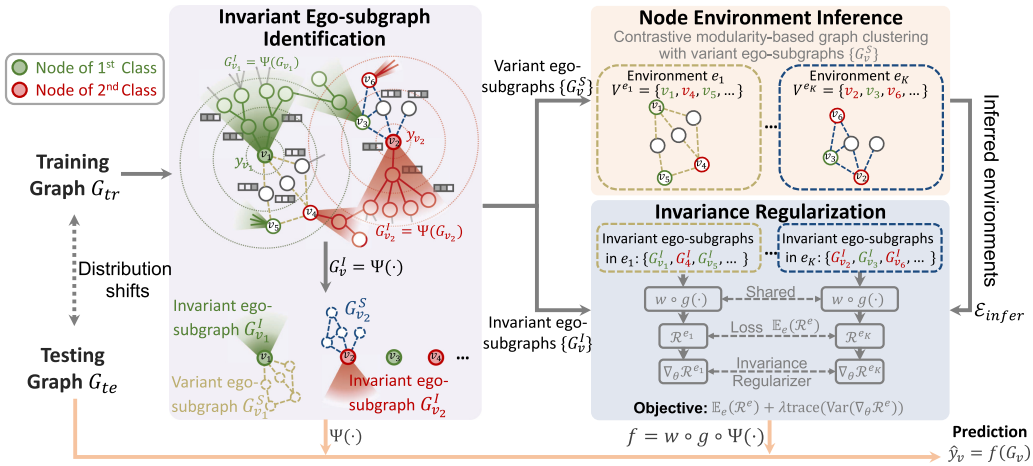


Fig. 1. The framework of **INL** model. Our proposed method jointly optimizes three modules: (1) The invariant ego-subgraph identification module uses $\Psi(\cdot)$ to identify the invariant ego-subgraph G_v^I and the variant ego-subgraph G_v^S for each node v. (2) The node environment inference module uses the variant ego-subgraphs $\{G_v^S\}$ to infer the latent environments by a contrastive modularity-based graph clustering. (3) The invariance regularization module jointly optimizes the invariant ego-subgraph generator $\Psi(\cdot)$, the representation learning function $g(\cdot)$, and the classifier $w(\cdot)$. *Training stage* (shown by grey arrows): We back propagate with the objective function to update model parameters. *Testing stage* (shown by orange arrows): We use the optimized model to make predictions. This example assumes that the node labels have two classes, which are denoted by red and green colors respectively.

we can not obtain a prior distribution of testing nodes for training.[2] However, Problem 1 is difficult to be directly solved, since (1) the nodes which are connected by graph structure are non-independent inducing obstacle for predictions, and (2) the environment labels for the nodes are *unobserved* [4, 40], which are usually unavailable or prohibitively expensive to collect for most real scenarios.

3 METHOD

In this section, we introduce our proposed **INL** in detail. The framework of **INL** is shown in Figure 1. Specifically, we first propose an invariant ego-subgraph identification module. Then, we infer environment labels by proposing a contrastive modularity-based graph clustering method. Last, we optimize the maximal invariant pattern criterion to produce invariant node representations capable of generalizing under distribution shifts with theoretical guarantees.

3.1 Problem Formulation

In this article, we focus on learning invariant node representation by adopting message-passing GNNs. Since only the immediate neighbors of nodes are aggregated in each message-passing layer, the representation of nodes only depends on their L-hop neighbors, where L is the number of message-passing layers. Therefore, we learn representations of nodes by only focusing on their L-order ego-graph, which is the common assumption for most message-passing GNNs [34, 38, 78]. Denote the node v's L-hop neighbors as $\mathcal{N}_v = \{u | d(v,u) \leq L\}$, where $d(v,u)$ is the shortest

[2]We follow this more challenging out-of-distribution generalization [2, 4, 11, 40, 42, 64] setting instead of the semi-supervised/adaptation setting that unlabeled testing graph data is available during training.

path distance between node v and u. The nodes in $\mathcal{N}_v$ and their connections form the ego-graph G_v of node v, which is represented as a local node feature matrix $X_v = \{x_u | u \in \mathcal{N}_v\}$ and local adjacency matrix $A_v = \{a_{u,u'} | u, u' \in \mathcal{N}_v\}$. We use $\mathbf{G_v}$ and G_v to denote the random variable and instance of ego-graphs and use $\mathbf{G}$ and $\mathbf{Y}$ to denote the random variable of input graph and node label vector, respectively. Then, we can reformulate the problem by using ego-subgraphs, i.e., an ego-graph dataset defined as $\mathcal{G} = \{\mathcal{G}^e\}_{e \in \text{supp}(\mathcal{E}_{tr})}$, where $\mathcal{G}^e = \{(G_v^e, y_v^e) | v \in V^e\}$ denotes the ego-graphs in environment e. Notice that ego-graphs are not independent samples, but they can be seen as a Markov blanket [34, 78], so the conditional distribution can be decomposed (conditional independence), i.e., $P(\mathbf{Y}|\mathbf{G}) = \prod_{\mathbf{v}} P(\mathbf{y}|\mathbf{G_v})$.

Problem 2. *Given the training graph where nodes are from multiple latent environments but without environment labels, the task is to jointly infer the node environments $\mathcal{E}_{infer}$ and learn $f^*(\cdot)$ in Problem 1 with $\mathcal{E}_{infer}$ to achieve good OOD generalization performance under distribution shifts.*

3.2 Invariant Ego-subgraph Identification

To enable OOD generalization, recent studies on invariant learning [2, 4, 11, 40, 42, 64] propose to train a predictor using only a portion of features of each input instance that capture the invariant and sufficiently predictive relations with labels. Since we have transformed the node representation learning task into only using ego-graphs G_v, we assume that each ego-graph instance has an invariant subgraph, i.e., ego-subgraph $G_v^I \subset G_v$, that possesses invariant and sufficiently predictive information to the node's label y_v in different environments under distribution shifts. We refer to the rest of each ego-graph, i.e., the complement of G_v^I, as the variant ego-subgraph and denote it as G_v^S. G_v^S represents the surrounding part of the node v whose relationship with the label is variant across different environments, e.g., *spurious correlations* for predicting node v. The graph model will have a better OOD generalization ability if it can identify the invariant ego-subgraph G_v^I for each node accurately and learn node representation based on G_v^I for predictions.

Formally, we denote a generator for each node's ego-graph to obtain the invariant ego-subgraph as $G_v^I = \Psi(G_v)$. Following the invariant learning literature [2, 11, 40, 42, 45, 50], we make the assumption:

Assumption 1. *Given ego-graph $\mathbf{G_v}$, there exists an optimal invariant ego-subgraph generator $\Psi^*(\mathbf{G_v})$ satisfying the following properties:*
(a) Invariance property*: $\forall e, e' \in \text{supp}(\mathcal{E}), P^e(\mathbf{y}|\Psi^*(\mathbf{G_v})) = P^{e'}(\mathbf{y}|\Psi^*(\mathbf{G_v}))$, where $P^e(\cdot)$ and $P^{e'}(\cdot)$ denote the probability distribution in two environments e and e', respectively.*
(b) Sufficiency property*: $\mathbf{y} = w^*(g^*(\Psi^*(\mathbf{G_v}))) + \epsilon$, $\epsilon \perp \mathbf{G_v}$, where $g^*(\cdot)$ denotes a representation learning function, w^* is the classifier, $\perp$ indicates statistical independence, and ϵ is random noise.*

The invariance assumption means that the node representations learned on invariant ego-subgraphs have an invariant relation to the node labels across different environments. The sufficiency assumption means that the node representations learned on invariant ego-subgraphs are sufficiently predictive to the node labels.

In this article, we instantiate $\Psi(\cdot)$ using two learnable masks on node features and graph structures (i.e., edges). First, the edge mask is responsible for splitting the local adjacency matrix A_v of the ego-graph G_v into the local adjacency matrix A_v^I of the invariant ego-subgraph G_v^I and the local adjacency matrix A_v^S of the variant ego-subgraph G_v^S. A straightforward strategy is to train a binary mask matrix $M^{A_v} = \{0, 1\}^{|\mathcal{N}_v| \times |\mathcal{N}_v|}$ on the local adjacency matrix A_v. However, directly optimizing such a mask matrix is a discrete optimization problem and intractable in practice, especially for large-scale graphs [88]. Besides, learning a mask for each ego-subgraph cannot share knowledge among different nodes. Therefore, we adopt a learnable GNN (denoted as GNN^{M}) to parameterize the mask matrix. Specifically, we relax edge masks from binary variables to continuous

variables in $[0, 1]$. The soft mask for each edge $(u, u'), u, u' \in \mathcal{N}_v$ in ego-graph G_v is:

$$M_{u,u'}^{A_v} = \text{Sigmoid}\left(\mathbf{Z}_u^{\text{M}^\top} \cdot \mathbf{Z}_{u'}^{\text{M}}\right), \quad \mathbf{Z}^{\text{M}} = \text{GNN}^{\text{M}}(G_v) \in \mathbb{R}^d. \tag{2}$$

Besides the edge mask, we also adopt a soft F-dimensional feature mask $M^X \in [0, 1]^F$ shared by all the nodes for selecting the invariant node features in the ego-graph G_v. The invariant ego-subgraph $G_v^I = (A_v^I, X_v^I)$ and variant ego-subgraph $G_v^S = (A_v^S, X_v^S)$ of G_v are calculated as:

$$A_v^I = M^{A_v} \odot A_v, X_v^I = M^X \odot X_v; \;\; A_v^S = A_v - A_v^I, X_v^S = X_v - X_v^I, \tag{3}$$

where $\odot$ is the element-wise matrix multiplication. Using the above method, we can generate all the invariant ego-subgraphs $\{G_v^I | v \in V\}$ and variant ego-subgraphs $\{G_v^S | v \in V\}$.

3.3 Node Environment Inference

After splitting the nodes' ego-graphs into invariant and variant subgraphs, we can infer the environment label $\mathcal{E}_{infer}$ using variant subgraphs $\{G_v^S | v \in V\}$. The intuition is that, since the invariant ego-subgraphs capture the invariant relationships between predictive node features and graph structures with the node labels, the variant ego-subgraphs in turn capture variant spurious correlations under different distributions. Consider two nodes v, v' from the same environment (e.g., two proteins from the same species or two papers published in the same period). Their variant ego-subgraphs G_v^S and $G_{v'}^s$ are likely show similar environment patterns. Based on the graph homophily assumption [57] that similar nodes are more likely to connect to each other, the nodes from the same environment will tend to be more densely connected in their variant ego-subgraphs than nodes from different environments (an illustrating example is shown in Figure 1). Therefore, we can infer the environments by conducting graph clustering based on the variant node features and edges.

Specifically, let X^S and A^S denote the node features and edges in $\{G_v^S | v \in V\}$. Assuming there are K latent environments in graph, we design a contrastive modularity-based clustering method to infer the environments by learning a cluster assignment matrix $C = \{C_v | v \in V\}$, where C_v is K-dimensional one-hot vector indicating the environment of node v. We propose to minimize the following contrastive objective for clustering the nodes denoted by (X^S, A^S):

$$\min_C \ell = -\frac{1}{K} \sum_{k=1}^{K} \log \frac{\exp(B_{k,k})}{\sum_{k'=1, k' \neq k}^{K} \exp(B_{k,k'})}, \tag{4}$$

where

$$B = \frac{1}{2m}\left(C^\top A^S C - \frac{1}{2m}\text{diag}\left(C^\top \mathbf{d}\mathbf{d}^\top C\right)\right). \tag{5}$$

In Equation (5), $\mathbf{d}$ and m indicate the degree vector and the number of edges calculated by A^S, respectively. diag$(\cdot)$ means only keeping the diagonal elements of the input matrix. $B \in \mathbb{R}^{K \times K}$ is the modularity matrix [60], whose entry $B_{k,k'}$ is the probability of an edge existing between cluster k and k'. Optimizing Equation (4) can maximize the connection probability between nodes from the same clusters (i.e., positive pairs) and minimize the connecting probability between nodes from the different clusters (i.e., negative pairs) via a contrastive scheme [13], encouraging to form clear clusters. Since optimizing the binary cluster assignment matrix is proven to be NP-hard [8], we follow Reference [73] to relax $C \in [0, 1]^{|V| \times K}$ as a soft cluster assignment and adopt a GNN to calculate the assignment matrix, i.e., $C = \text{Softmax}(\text{GNN}^{\text{C}}(X^S, A^S))$. Finally, the optimal cluster assignment C^* can be used to indicate the inferred environments $\mathcal{E}_{infer}$ of nodes.

3.4 Invariance Regularization

After obtaining the inferred invariant ego-subgraphs $\{G_v^I | v \in V\}$ and environment labels $\mathcal{E}_{infer}$, we propose the invariance regularization module, which can make the graph model to generate node representations capable of OOD generalization under distribution shifts. Specifically, we aim to learn the optimal generator Ψ^* in Assumption 1 by proposing and optimizing the **maximal invariant ego-subgraph generator** criterion. Following the invariant learning literature [11, 40, 50, 51], we give the following definition:

Definition 1. The **invariant ego-subgraph generator set** $\mathcal{I}$ with respect to $\mathcal{E}$ is defined as:

$$\mathcal{I}_{\mathcal{E}} = \{\Psi(\cdot) : P^e(\mathbf{y}|\Psi(\mathbf{G_v})) = P^{e'}(\mathbf{y}|\Psi(\mathbf{G_v})), e, e' \in \text{supp}(\mathcal{E})\}. \tag{6}$$

Then, we show that the optimal generator Ψ^* satisfies the following theorem:

Theorem 1. *A generator $\Psi(\mathbf{G_v})$ is the optimal generator that satisfies Assumption 1 if and only if it is the maximal invariant ego-subgraph generator, i.e., $\Psi^* = \arg\max_{\Psi \in \mathcal{I}_{\mathcal{E}}} I(\mathbf{y}; \Psi(\mathbf{G_v}))$, where $I(\cdot;\cdot)$ is the mutual information between the label and the generated invariant ego-subgraph.*

Proof. Denote $\hat{\Psi} = \arg\max_{\Psi \in \mathcal{I}_{\mathcal{E}}} I(\mathbf{y}; \Psi(\mathbf{G_v}))$. According to the invariance property of Assumption 1, we have $\Psi^* \in \mathcal{I}_{\mathcal{E}}$. Therefore, we prove the theorem by showing that $I(\mathbf{y}; \hat{\Psi}(\mathbf{G_v})) \leq I(\mathbf{y}; \Psi^*(\mathbf{G_v}))$ and consequently, $\hat{\Psi} = \Psi^*$. To show the inequality, we use the functional representation lemma [23], which states that for any random variables $\mathbf{X}_1$ and $\mathbf{X}_2$, there exists a random variable $\mathbf{X}_3$ independent of $\mathbf{X}_1$ such that $\mathbf{X}_2$ can be represented as a function of $\mathbf{X}_1$ and $\mathbf{X}_3$. So, for $\Psi^*(\mathbf{G_v})$ and $\hat{\Psi}(\mathbf{G_v})$, there exists $\Psi'(\mathbf{G_v})$ satisfying that $\Psi'(\mathbf{G_v}) \perp \Psi^*(\mathbf{G_v})$ and $\hat{\Psi}(\mathbf{G_v}) = \gamma(\Psi^*(\mathbf{G_v}), \Psi'(\mathbf{G_v}))$, where $\gamma(\cdot)$ is a function. Then, we can derive that:

$$\begin{aligned} I(\mathbf{y}; \hat{\Psi}(\mathbf{G_v})) &= I(\mathbf{y}; \gamma(\Psi^*(\mathbf{G_v}), \Psi'(\mathbf{G_v}))) \\ &\leq I(\mathbf{y}; \Psi^*(\mathbf{G_v}), \Psi'(\mathbf{G_v})) \\ &= I(w^*(g^*(\Psi^*(\mathbf{G_v}))); \Psi^*(\mathbf{G_v}), \Psi'(\mathbf{G_v})) \\ &= I(w^*(g^*(\Psi^*(\mathbf{G_v}))); \Psi^*(\mathbf{G_v})) \\ &= I(\mathbf{y}; \Psi^*(\mathbf{G_v})), \end{aligned} \tag{7}$$

which finishes the proof. □

Theorem 1 provides us an objective function to optimize the invariant ego-subgraph generator. However, directly solving according to Theorem 1 for a non-linear Ψ is difficult [40]. Following the invariant learning literature [40], we minimize the following invariance regularizer:

$$\mathbb{E}_{e \in \text{supp}(\mathcal{E}_{infer})} \mathcal{R}^e(f(\mathbf{G_v}), \mathbf{y}; \theta) + \lambda \text{trace}\left(\text{Var}_{\mathcal{E}_{infer}}(\nabla_\theta \mathcal{R}^e)\right), \tag{8}$$

where $f(\cdot) = w \circ g \circ \Psi$, $\mathcal{E}_{infer}$ is the inferred environment label, and θ denotes all the learnable parameters. Recall that $g(\cdot)$ is the representation learning function of the invariant ego-subgraphs and $w(\cdot)$ is the classifier. We instantiate g as another GNN as: $\mathbf{Z}_I = \text{GNN}^{\text{I}}(G_v^I)$, where $\mathbf{Z}_I$ are the node representations capturing invariant patterns from the ego-subgraphs. $w(\cdot)$ is instantiated as a multilayer perceptron with the ReLU [1] activation function, followed by the softmax function. By optimizing Equation (8), we can get our desired generator Ψ and the ego-subgraph representation learning function $g(\cdot)$, which collectively serve as our representation learning method $h(\cdot)$, i.e., $h = g \circ \Psi$.

We further theoretically analyze our **INL** model by showing that the maximal invariant ego-subgraph generator can achieve OOD optimality.

Theorem 2. *Let Ψ^* be the optimal invariant ego-subgraph generator for $\mathbf{G_v}$ in Assumption 1 and denote the complement as $\mathbf{G_v}\backslash\Psi^*(\mathbf{G_v})$, i.e., the corresponding variant ego-subgraph. Then, we can obtain the optimal predictor under distribution shifts, i.e., the solution to Problem 1, as follows:*

$$\arg\min_{w,g} w \circ g \circ \Psi^*(\mathbf{G_v}) = \arg\min_{f} \sup_{e\in\text{supp}(\mathcal{E})} \mathcal{R}(f|e), \tag{9}$$

if the following conditions hold: (1) $\Psi^*(\mathbf{G_v}) \perp \mathbf{G_v}\backslash\Psi^*(\mathbf{G_v})$; and (2) $\forall\Psi \in \mathcal{I}_{\mathcal{E}}$, $\exists\, e' \in \text{supp}(\mathcal{E})$ such that $P^{e'}(\mathbf{G_v}, \mathbf{y}) = P^{e'}(\Psi(\mathbf{G_v}), \mathbf{y})P^{e'}(\mathbf{G_v}\backslash\Psi(\mathbf{G_v}))$ and $P^{e'}(\Psi(\mathbf{G_v})) = P^{e}(\Psi(\mathbf{G_v}))$.

Proof. Denote the function to obtain the complement of invariant ego-subgraph as $\Phi(\mathbf{G_v}) = \mathbf{G_v}\backslash\Psi(\mathbf{G_v})$ and $\Phi^*(\mathbf{G_v}) = \mathbf{G_v}\backslash\Psi^*(\mathbf{G_v})$. By assumption, $\Psi^*(\mathbf{G_v}) \perp \Phi^*(\mathbf{G_v})$. Further denote $\hat{f} = \arg\min_{w,g} w \circ g \circ \Psi^*(\mathbf{G_v})$. By Assumption 1, we have

$$\hat{f}(\mathbf{G_v}) = w^* \circ g^* \circ \Psi^*(\mathbf{G_v}). \tag{10}$$

To show that $\hat{f}$ is f^*, our proof strategy is to show that $\forall e \in \text{supp}(\mathcal{E})$, for any possible f, $\mathcal{R}(\hat{f}|e) \le \mathcal{R}(f|e')$ and therefore $\sup_{e\in\text{supp}(\mathcal{E})} \mathcal{R}(\hat{f}|e) \le \sup_{e\in\text{supp}(\mathcal{E})} \mathcal{R}(f|e)$.

To show the inequality, we have:

$$\begin{aligned}
&\mathcal{R}(\hat{f}|e) && (11)\\
&= \mathbb{E}^e_{\mathbf{G_v},\mathbf{y}}[\ell(\hat{f}(\mathbf{G_v}), \mathbf{y})] && (12)\\
&= \sum_{\mathbf{G_v},\mathbf{y}} P^e(\mathbf{G_v}, \mathbf{y})\ell(\hat{f}(\mathbf{G_v}), \mathbf{y}) && (13)\\
&= \sum_{\Phi^*(\mathbf{G_v})} P^e(\Phi^*(\mathbf{G_v})) \left[\sum_{\Psi^*(\mathbf{G_v}),\mathbf{y}} P^e(\Psi^*(\mathbf{G_v}), \mathbf{y}) \cdot \ell\left(w^*(g^*(\Psi^*(\mathbf{G_v}))), \mathbf{y}\right) \right] && (14)\\
&= \sum_{\Psi^*(\mathbf{G_v}),\mathbf{y}} P^e(\Psi^*(\mathbf{G_v}), \mathbf{y})\ell(w^*(g^*(\Psi^*(\mathbf{G_v}))), \mathbf{y}) && (15)\\
&\le \sum_{\Psi(\mathbf{G_v}),\mathbf{y}} P^e(\Psi(\mathbf{G_v}), \mathbf{y})\ell(w(g(\Psi(\mathbf{G_v}))), \mathbf{y}) && (16)\\
&= \sum_{\Phi(\mathbf{G_v})} P^{e'}(\Phi(\mathbf{G_v})) \sum_{\Psi(\mathbf{G_v}),\mathbf{y}} P^e(\Psi(\mathbf{G_v}), \mathbf{y})\ell(w(g(\Psi(\mathbf{G_v}))), \mathbf{y}) && (17)\\
&= \sum_{\Phi(\mathbf{G_v})} \sum_{\Psi(\mathbf{G_v}),\mathbf{y}} P^{e'}(\Psi(\mathbf{G_v}), \mathbf{y})P^{e'}(\Phi(\mathbf{G_v}))\ell(w(g(\Psi(\mathbf{G_v}))), \mathbf{y}) && (18)\\
&= \sum_{\mathbf{G_v},\mathbf{y}} P^{e'}(\mathbf{G_v}, \mathbf{y})\ell(f(\mathbf{G_v}), \mathbf{y}) && (19)\\
&= \mathbb{E}^{e'}_{\mathbf{G_v},\mathbf{y}}[\ell(f(\mathbf{G_v}), \mathbf{y})] && (20)\\
&= \mathcal{R}(f|e'). && (21)
\end{aligned}$$

□

Intuitively, Theorem 2 shows that we can transform the OOD generalization problem into finding the optimal invariant ego-subgraphs while maintaining the optimality. The proofs of the above theorems are inspired by the invariant learning literature [45, 50, 51, 78], and a motivating example for better understanding is provided in Section 3.6. It indicates that our method can get rid of spurious correlations and learn OOD generalized node representations based on the identified invariant ego-subgraphs.

ALGORITHM 1: The training procedure of the proposed **INL**.

Input: The input graph and node labels
Output: An optimized predictor $f(\cdot)$ mapping node to its label

for *epoch* $\leftarrow$ 1 to Epoch **do**
  Generate the edge masks with the shared learnable GNN^{M} by Equation (2).
  Obtain the invariant and variant ego-subgraphs of all nodes by Equation (3).
  for *epoch*$'$ $\leftarrow$ 1 to Epoch_Cluster **do**
    Optimize cluster assignment C by minimizing the objective in Equation (4).
  end for
  Infer environments $\mathcal{E}_{infer}$ by obtaining the environment of each node $e_v = \text{argmax}\, C_v$.
  Generate invariant node representation $\mathbf{Z}_v^I = \text{GNN}^{\text{I}}(G_v^I)$ for all nodes.
  Back propagate with the objective function in Equation (8).
end for

3.5 Training Procedure

We present the pseudocode of **INL** in Algorithm 1 to show the training procedure. Specifically, we first obtain the invariant and variant ego-subgraphs for all nodes with the learnable masks on node features and edges. Then, we infer the environments for all nodes with the variant node features and edges from variant ego-subgraphs. And we learn the invariant node representations with invariance regularization based on the inferred invariant ego-subgraphs and environment labels. Note that the adopted GNNs including GNN^{M}, GNN^{C}, and GNN^{I} for all ego-graphs are *shared*, following References [34, 78]. At the testing stage, we directly adopt the optimized f to conduct predictions. In Algorithm 1, "Epoch" means the overall number of epochs for optimizing the proposed method, and "Epoch_Cluster" denotes the number of epochs for clustering to infer environments in each training epoch. The setting details of the hyper-parameters can be found in Section 4.1.3.

3.6 A Motivating Example

For better understanding our proposed method intuitively, we present a linear toy example and the corresponding theoretical analysis inspired by Reference [78] to show why our method can achieve out-of-distribution generalization by learning node representations based on invariant ego-subgraph G_v^I (i.e., invariant node features X_v^I and structures A_v^I).

For simplification, in this toy example, we consider the ego-graph G_v (and $\mathcal{N}_v$) only contains the centered node v and its 1-hop neighbors (i.e., $L = 1$), which can be split into invariant ego-subgraph G_v^I (and $\mathcal{N}_v^I$) and variant ego-subgraph G_v^S (and $\mathcal{N}_v^S$). And we consider the dimensionality of node features $F = 2$, including one-dimensional invariant node feature x_v^I and variant node feature x_v^S, i.e., $x_v = [x_v^I, x_v^S]$ for each node v. The illustration of ego-graph G_v is shown in Figure 2. The dependence among variables in the toy example is shown in Figure 3. We do not distinguish the notation of random variables and of their particular instances when there is no risk of confusion in this toy example.

Considering the representation learning function g^* that averages the node representations in invariant ego-subgraph G_v^I to produce the centered node representations and classifier w^* is identity mappings in Assumption 1, the node label can be determined by the invariant node features and structures:

$$y_v = \frac{1}{|\mathcal{N}_v^I|} \sum_{u \in \mathcal{N}_v^I} x_u^I + \epsilon_1, \tag{22}$$

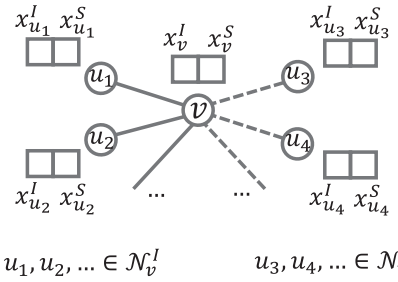


Fig. 2. The ego-graph G_v in the toy example.

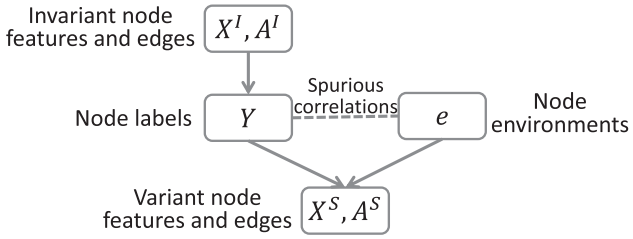


Fig. 3. The dependence among variables in our synthetic datasets.

where ϵ_1 is standard normal noise. And we assume that the variant node feature x_v^S is generated by identity mapping given the input of the node's label y_v and environment e_v, which can be denoted as:

$$x_v^S = y_v + e_v + \epsilon_2, \tag{23}$$

where ϵ_2 is standard normal noise. e_v denotes the node v's environment, following normal distribution whose mean and variance are dependent on node environment. Besides, we assume the variant structures are also dependent on the node environment and the environments of nodes in $\mathcal{N}_v^S$ is e_v. For example, in citation networks, the invariant node features and structures can be the paper published avenues and citations among them that determine the subject topics (i.e., labels), while the variant node features and structures can be the citation indexes and edges between papers with high citations in some publication periods (i.e., environments).

Therefore, given the invariant and variant ego-subgraph, we consider the following predictor model:

$$\hat{y}_v = \frac{1}{|\mathcal{N}_v^I|} \sum_{u \in \mathcal{N}_v^I} \left(\theta_1 x_u^I + \theta_2 x_u^S\right) + \frac{1}{|\mathcal{N}_v^S|} \sum_{u \in \mathcal{N}_v^S} \left(\theta_3 x_u^I + \theta_4 x_u^S\right). \tag{24}$$

Note that the ideal solution for the predictor model is $\theta = [\theta_1, \theta_2, \theta_3, \theta_4] = [1, 0, 0, 0]$, indicating that the predictor accurately identifies the sufficiently predictive and invariant node features and structures for making OOD generalized predictions. However, the following proposition shows that we cannot obtain this ideal solution if only using standard **empirical risk minimization (ERM)**:

Proposition 3. *Denoting the risk (i.e., loss) of the predictor model f as $\mathcal{R} = \frac{1}{|V|}\sum_{v\in V} \mathbb{E}_{\mathbf{y_v}|\mathbf{G_v}=G_v} ||\hat{y}_v - y_v||_2^2$, the optimal solution for objective $min_\theta \mathcal{R}$ is $\theta = [\theta_1, \theta_2, \theta_3, \theta_4] = [1 - \frac{\mu^S}{2(\mu^S - \mu^I)}, \frac{\mu^S}{2(\mu^S - \mu^I)}, \frac{\mu^I}{2(\mu^S - \mu^I)}, \frac{-\mu^I}{2(\mu^S - \mu^I)}]$, assuming $\mu^I \neq \mu^S$, where $\mu^I = \frac{1}{|V|}\sum_{v\in V} \frac{1}{|\mathcal{N}_v^I|}\sum_{u\in\mathcal{N}_v^I} e_u$ and $\mu^S = \frac{1}{|V|}\sum_{v\in V} \frac{1}{|\mathcal{N}_v^S|}\sum_{u\in\mathcal{N}_v^S} e_u$ are dependent on the node environments.*

Table 2. Statistics of the Datasets

	Citeseer	Amazon-Photo	OGB-Arxiv	OGB-Proteins
#Nodes	3,327	7,650	169,343	132,534
#Edges	9,104	238,162	1,166,243	39,561,252
#Classes	6	8	40	2
Metric	Accuracy	Accuracy	Accuracy	ROC-AUC

#Nodes/#Edges are the number of nodes and edges in the graph of the dataset, respectively. #Classes denotes the number of Classes. Metric is the evaluation metric of the dataset.

The proof is in Appendix A.1. Proposition 3 indicates directly optimizing with ERM will inevitably make the predictor model heavily rely on spurious correlations, since $\theta_2, \theta_3, \theta_4$ is not constant zero, so that the model performs poorly under distribution shifts with multiple latent environments. Next, we show that our objective in Equation (8) can mitigate this issue.

Proposition 4. *The solution of optimizing the invariance regularizer in Equation* (8) *to the minimum satisfies* $[\theta_2, \theta_3, \theta_4] = [0, 0, 0]$.

The proof is in Appendix A.2. Proposition 4 indicates our method can get rid of spurious correlations and learn OOD generalized node representations under distribution shifts with multiple latent environments by generating node representations based on the identified invariant ego-subgraph G_v^I.

Intuitively, Proposition 3 shows that the optimal solution under standard **empirical risk minimization (ERM)** in this toy example (as shown in Figure 2) consists of non-zero coefficients of the predictor model for variant ego-subgraph, which means that the predictions rely on variant environment information, e.g., different species that the proteins come from in protein-protein interaction graphs and the publication time of papers in citation networks. Therefore, the OOD generalization performance is poor. However, Proposition 4 shows that the optimal solution using the proposed method in this toy example only includes non-zero coefficients of the predictor model for invariant ego-subgraph, demonstrating that our method can make predictions only based on the invariant information and is not affected by variant spurious correlations, leading to strong OOD generalization ability.

4 EXPERIMENTS

In this section, we empirically evaluate our proposed method through the experiments on both synthetic and real-world datasets, including the experimental setup, quantitative comparisons, complexity analysis, ablation studies, the impact of the hyper-parameters, and so on.

4.1 Experimental Setup

4.1.1 Datasets. We adopt two synthetic datasets with artificial distribution shifts based on two representative node classification benchmarks, Citeseer [86] and Amazon-Photo [69], in which ground-truth generation processes are controllable. And we also consider another two real-world datasets OGB-Arxiv and OGB-Proteins from Open Graph Benchmark [33]. The statistics of these datasets are provided in Table 2.

Synthetic datasets. Citeseer and Amazon-Photo are two commonly used node classification benchmarks. Citeseer is a citation network where nodes represent papers and edges indicate their citations. Amazon-Photo is a co-purchasing network where nodes represent items and edges represent two items purchased together. For evaluating the model's out-of-distribution generalization ability, we introduce distribution shifts between the training and testing data.

Following Reference [78], we first use a randomly initialized 2-layer GCN to generate node labels Y based on the original node features and edges, which can be regarded as invariant and sufficiently predictive information to the labels and denoted by (X^I, A^I). Then, we assign nodes into different environments and create spurious correlations between the label and environment. Based on the label and environment of each node, we generate an additional feature matrix and additional edges as the variant patterns, which are denoted by (X^S, A^S). The generated feature (i.e., X^S) has the same dimensionality as the original feature (i.e., X^I) and the number of generated edges (i.e., A^S) equals the original number of edges (i.e., A^I). We then concatenate the two feature matrices and add the generated edges into the original graph as the input data, i.e., $(X = [X^I, X^S], A = A^I + A^S)$. The dependence among these variables is illustrated in Figure 3.

More specifically, we set the ground truth number of environments as $K = 3$ and adopt a hyper-parameter $r \in [0, 1]$ to control the strength of spurious correlations by setting the probability of node v belonging to the kth environment as $P(v \in V^{e_k}) = r$ if $k \equiv y_v (\text{mod } K)$ and $P(v \in V^{e_k}) = (1-r)/2$ otherwise. Intuitively, nodes with the same labels more likely belong to the same environment. For example, for the nodes whose labels are 1 or 4, the probability of these nodes belonging to the 1st environment is r and the probability belonging to the 2nd or 3rd environment is $(1-r)/2$. In the $K = 3$ case, $r = 1/3$ means there is no spurious correlation and a larger r indicates a higher spurious correlation between the label and environment. We set $r_{test} = 1/3$ and vary r_{train} in $\{1/3, 0.5, 0.7\}$ to generate testing and training graphs, respectively, which simulates different strengths of distribution shifts. We hold out 10% nodes from the training graph for validation.

After obtaining the environment of each node, we generate variant node features X^S by a two-layer MLP given the label and environment ID as the input. Then, we generate variant edges A^S by connecting nodes with similar variant node features. In particular, we first calculate the scores of any potential edges (i.e., edges not in A^I) by cosine similarity of variant node features of the two nodes. According to the scores, we select Top-t edges in all potential edges to form the variant edges A^S, where the number of invariant and variant edges is equal, i.e., t is the number of edges in A^I.

OGB-Arxiv. This dataset consists of Arxiv CS papers from 40 subject areas and their citations. The task is to predict the 40 subject areas of the papers,[3] e.g., cs.AI, cs.LG, cs.OS. Instead of the semi-supervised/adaptation setting where unlabeled testing data is available during training [33], we follow the more common and challenging out-of-distribution generalization [2, 4, 11, 40, 42, 64] setting, i.e., the testing nodes are not available in the training stage. Since several latent influential environment factors (e.g., the popularity of research topics) can change significantly over time, the properties of citation networks will be varying in different time ranges. Therefore, the node distribution shifts on OGB-Arxiv are introduced by selecting papers published before 2011 as training set, within 2011–2014 as validation set, and within 2014–2016/2016–2018/2018–2020 as three testing sets.

OGB-Proteins. In this dataset, nodes represent proteins and edges indicate different types of biologically meaningful associations between proteins, e.g., physical interactions, co-expression, or homology [71]. The task is to predict the presence of protein functions in a binary classification setup. We also follow the out-of-distribution generalization [2, 4, 11, 40, 42, 64] setting, i.e., the testing nodes are not available in the training stage, instead of the semi-supervised setting. Since the latent influential environment factors can vary from different species that the proteins come from, the properties and associations of proteins will also be different in different species. Therefore, the node distribution shifts on OGB-Proteins are introduced by selecting nodes into training/validation/testing sets according to their species. Specifically, the training set and

[3] https://arxiv.org/corr/subjectclasses.

validation set include proteins and their associations from four and one species, respectively. And each of the three testing sets consists of proteins and their associations from one of the left three species.

The datasets are publicly available as follows:

- **Citeseer**: https://github.com/kimiyoung/planetoid with MIT license
- **Amazon-Photo**: https://github.com/shchur/gnn-benchmark with MIT License
- **OGB-Arxiv, OGB-Proteins**: https://ogb.stanford.edu/docs/nodeprop/ with MIT License

4.1.2 Baselines. We compare our **INL** with the following representative state-of-the-art methods:

- **ERM** [74]: We use ERM to denote the backbone GNN models, which are trained with the standard empirical risk minimizing, namely, minimizing the sum of risks across environments and training samples.
- **GroupDRO**[4] [65]: It handles the problem that the distribution minority lacks sufficient training and seeks to explicitly optimize the worst performance over a distribution set to achieve OOD generalization performance.
- **IRM**[5] [4]: It is a representative invariant learning method. To learn invariances across environments for enabling OOD generalization, it seeks to find data representations or features so the optimal classifier on top of that representation matches for all environments. We conduct random environment partitions on the nodes of input graph for training, because this method needs the explicit environment labels in advance.
- **V-REx**[6] [42]: This method is proven to be able to recover the causal mechanisms of the targets and is robust to distribution shifts. Specifically, it minimizes the risk variances of the training environments for reducing the risk variances of the test environments, leading to good OOD generalization. Since this method relies on the explicit environment labels that are unavailable for the nodes in multiple latent environments, we conduct random environment partitions on the nodes of input graph during training stage.
- **EERM**[7] [78]: It is a recent pioneering work that can tackle node-level prediction tasks under distribution shifts and achieves a valid solution for the node-level OOD problem under mild conditions. It studies invariant predictions on graph by assuming all nodes share a single environment. However, it ignores the more common and challenging situation that nodes are from multiple latent environments.
- **GIL** [45]: It learns invariant graph-level representations under distribution shifts. However, it only focuses on the graph-level generalization on graph classification tasks but cannot tackle the key problem studied in this article where distribution shifts exist on nodes. In the experiments, we modify its every module from graph-level to node-level for comparisons.

Since all the methods are model-agnostic, we use GCN [38] as the GNN backbone on the synthetic datasets and adopt GraphSAGE [30] and GAT [75] on the real-world datasets for a comprehensive comparison. Intuitively, the node classification on the synthetic datasets is simpler than that on the real-world datasets. Therefore, the classical GNN model, GCN, is used on the synthetic datasets, while relatively advanced models, GraphSAGE and GAT, are considered on the real-world datasets.

[4]https://github.com/kohpangwei/group_DRO.
[5]https://github.com/facebookresearch/InvariantRiskMinimization.
[6]https://github.com/capybaralet/REx_code_release.
[7]https://github.com/qitianwu/GraphOOD-EERM.

Table 3. Selected Hyper-parameters of λ and $|\mathcal{E}_{infer}|$ of Our Method on Each Dataset

	Citeseer	Amazon-Photo	OGB-Arxiv	OGB-Proteins
λ	10^{-4}	10^{-4}	10^{-2}	10^{0}
$\lvert\mathcal{E}_{infer}\rvert$	3	3	3	4

Table 4. Selected Hyper-parameters of the Baselines on Each Dataset

		Range	Citeseer	Amazon-Photo	OGB-Arxiv	OGB-Proteins
Number of Training Environments	IRM	{2, 3, 4}	3	2	3	2
	GroupDRO	{2, 3, 4, 5}	2	2	4	4
	V-REx	{2, 3, 4}	3	4	2	2
	EERM	{2, 3, 4, 5, 10}	3	5	4	3
	GIL	{2, 3, 4}	2	2	3	3
Regularizer Coefficient	IRM	$\{10^{-4}, 10^{-2}, 10^{0}\}$	10^{-2}	10^{-4}	10^{-2}	10^{-2}
	V-REx	$\{10^{-4}, 10^{-2}, 10^{0}, 10^{2}, 10^{4}\}$	10^{-4}	10^{-4}	10^{0}	10^{-2}
	EERM	$\{10^{-4}, 10^{-2}, \frac{1}{3}, 0.5, 1.0, 2.0, 5.0\}$	10^{-2}	2.0	1.0	1.0
	GIL	$\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$	10^{-4}	10^{-3}	10^{-2}	10^{-2}

4.1.3 Implementation Details. The number of epochs for optimizing our proposed method (i.e., Epoch in Algorithm 1) and baselines is set to 200 for the synthetic datasets (i.e., Citeseer and Amazon-Photo) and 500 for the real-world datasets (i.e., OGB-Arxiv and OGB-Proteins). The number of epochs for clustering to infer environments in each training epoch (i.e., Epoch_Cluster in Algorithm 1) is 20. The Adam optimizer is adopted for gradient descent. Since we focus on node classification tasks, we use the cross-entropy loss as the loss function ℓ. The classifier w is instantiated as a two-layer MLP. The activation function is ReLU [1]. The evaluation metric is ROC-AUC for OGB-Proteins datasets and accuracy for the others. For GNN^{M}, GNN^{C}, and GNN^{I}, the number of layers is set to 2 on all the datasets. The dimensionality of the node representations d is 32 on the synthetic datasets, 128 on OGB-Arxiv, and 256 on OGB-Proteins. Note that these GNNs including GNN^{M}, GNN^{C}, GNN^{I} are shared for all ego-subgraphs following References [34, 78]. The invariance regularizer coefficient λ in Equation (8) is chosen from $\{10^{-4}, 10^{-2}, 10^{0}\}$. The number of the inferred environments $|\mathcal{E}_{infer}|$ is chosen from $\{2, 3, 4\}$, which is the dimensionality of the vector C_v indicating the node v's environment in the cluster assignment matrix C. We report mean results and standard deviations of 10 runs. The selected λ and $|\mathcal{E}_{infer}|$ are reported in Table 3.

As for the baselines, we implement them using the official source codes. We conduct the hyper-parameter search for each baseline covering the search range of both our method and the original paper (if the search range is reported). The search range and the selected hyper-parameters of the baselines are reported in Table 4. The other hyper-parameters of the baselines are kept consistent with our method as described above.

We conduct the experiments with the following hardware and software configurations:

- Operating System: Ubuntu 18.04.1 LTS
- CPU: Intel(R) Xeon(R) CPU E5-2699 v4@2.20 GHz
- GPU: NVIDIA GeForce RTX 3090 with 24 GB of Memory
- Software: Python 3.6.5; NumPy 1.19.2; PyTorch 1.10.1; PyTorch Geometric 2.0.3 [25].

4.2 Experiments on Synthetic Datasets

The experimental results are shown in Table 5, from which we have the following observations: Our proposed **INL** consistently and significantly outperforms the baselines and achieves the best

Table 5. Node Classification Accuracy (%) on Testing Sets of the Synthetic Datasets

	Citeseer			Amazon-Photo		
r_{train}	$r = 1/3$	$r = 0.5$	$r = 0.7$	$r = 1/3$	$r = 0.5$	$r = 0.7$
GCN(ERM)	47.09±3.44	45.36±5.54	40.09±2.12	48.26±2.26	47.91±3.24	39.23±5.27
IRM	48.84±2.75	45.39±2.07	42.89±2.38	<u>53.75±1.31</u>	50.98±3.09	<u>42.23±2.75</u>
GroupDRO	49.32±6.47	46.30±5.44	40.68±2.83	49.62±6.45	47.65± 8.34	41.15±5.50
V-REx	47.53±3.65	43.11±4.06	41.03±4.29	47.13±8.01	48.53± 8.37	37.49±5.39
EERM	53.07±4.39	45.50±3.68	41.53±1.96	52.25±5.90	<u>51.03±2.93</u>	41.69±4.63
GIL	<u>55.71±1.24</u>	<u>47.42±2.10</u>	<u>44.87±3.26</u>	53.19±2.74	50.01±2.06	41.79±3.98
INL	**60.48**±0.77*	**56.74**±0.75*	**54.78**±2.50*	**55.86**±1.63*	**55.07**±2.27*	**46.90**±2.06*
Improvement	4.77↑	9.32↑	9.91↑	2.11↑	4.04↑	4.67↑

In each column, the boldfaced and the underlined score denotes the best and the second-best result, respectively. Numbers in the lower right corner denote standard deviations. "*"indicates the statistically significant improvements (one-tailed t-test with $p < 0.05$) upon the best baseline.

performance in all settings. The results demonstrate the effectiveness of our proposed method in handling distribution shifts, which has a remarkable out-of-distribution generalization ability. The general invariant learning methods, e.g., IRM, GroupDRO, V-REx, only have slight improvements to ERM. EERM is a recently proposed invariant method specifically designed for learning node representations but assumes a single environment is shared for all the nodes. EERM outputs competitive results in some settings but fails to obtain consistent improvements, indicating modeling multiple latent environments is crucial for handling distribution shifts in graph. GIL achieves promising gains over the other baselines, but the proposed method still performs better than it.

In addition, when $r_{train} = 1/3$, i.e., no distribution shifts between training and testing data, our proposed method also achieves the best results, meaning that learning invariant ego-subgraphs for nodes is also beneficial. As r_{train} grows larger, the performance of all the methods tends to decrease, since there exists a larger degree of distribution shift. Nevertheless, our proposed method is able to maintain the most relatively stable performance. In fact, the performance gap between **INL** and the best results of baselines becomes more significant as the degree of distribution shift increases. For example, the accuracy improvements against the strongest baselines increases from 4.77% to 9.91% when r_{train} changes from 1/3 to 0.7 on Citeseer, indicating the powerful OOD generalization ability of our method under various complex distribution shifts.

To further analyze whether our method can accurately capture the invariant ego-subgraphs under distribution shifts, we compare the output invariant node features and structures with the ground-truth on the synthetic dataset Citeseer. The evaluation metric is ROC-AUC. The results in Figure 4 show that the ROC-AUC for discovering invariant node features and structures is around 70% and 80%, respectively, which is significantly higher than random selection (50%). It demonstrates our **INL** can discover the truly predictive invariant ego-subgraphs and further make OOD generalized predictions.

4.3 Experiments on Real-world Graphs

We further evaluate the effectiveness of our method on two real-world graph datasets, i.e., OGB-Arxiv and OGB-Proteins from OGB [33]. The properties of citation networks can change significantly in different time ranges. So, the node distribution shifts on OGB-Arxiv are introduced by selecting papers published before 2011 as training set, within 2011–2014 as validation set, and

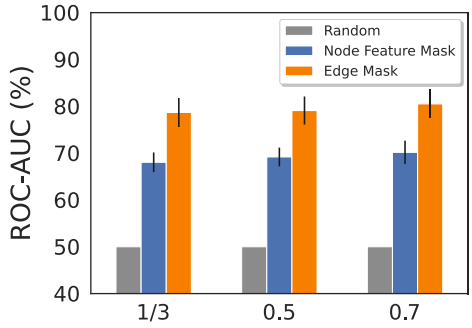


Fig. 4. Results of discovering the ground-truth invariant node features and edges on Citeseer.

Table 6. Node Classification Results (Accuracy for OGB-Arxiv, ROC-AUC for OGB-Proteins, %) on Testing Sets of the Real-world Datasets

Dataset		**OGB-Arxiv**			**OGB-Proteins**		
Backbone	**Method**	2014-2016	2016-2018	2018-2020	Species-1	Species-2	Species-3
GraphSAGE	ERM	45.24±0.60	42.25±1.02	38.75±0.97	66.44±0.48	64.18±0.59	57.61±1.72
	IRM	45.31±0.56	42.48±1.98	40.23±1.07	67.03±0.41	64.38±0.87	57.54±1.13
	GroupDRO	45.35±0.68	42.56±0.88	39.26±0.81	66.28±0.27	64.51±0.35	57.87±0.89
	V-REx	45.27±0.71	42.51±1.13	39.31±0.96	67.43±0.18	64.38±0.51	57.71±1.42
	EERM	46.15±0.98	43.27±1.01	41.61±0.96	66.40±0.59	64.39±0.12	57.12±1.21
	GIL	47.92±0.45	45.78±0.62	41.27±0.91	67.39±0.86	66.54±1.38	55.81±1.76
	INL	**49.43**±0.53*	**49.19**±0.98*	**46.34**±0.87*	**72.20**±0.41*	**69.47**±0.72*	**61.07**±1.45*
GAT	ERM	45.94±1.03	43.52±0.95	40.42±0.98	66.34±0.45	64.35±0.60	57.83±1.75
	IRM	46.73±0.91	44.32±0.91	42.04±0.99	66.33±0.30	64.61±0.43	56.91±0.93
	GroupDRO	45.95±0.89	43.52±1.25	40.43±1.32	66.30±0.27	64.52±0.31	57.95±0.79
	V-REx	45.93±0.87	45.69±0.81	41.01±1.03	66.14±0.58	64.31±0.60	57.73±1.32
	EERM	45.99±1.22	45.32±0.84	42.01±1.36	66.35±0.48	64.32±0.21	56.13±0.98
	GIL	47.70±0.93	45.65±1.41	41.87±1.89	66.31±0.69	67.12±0.89	55.98±0.83
	INL	**50.37**±1.01*	**49.12**±1.23*	**45.35**±1.32*	**73.89**±0.39*	**71.42**±0.28*	**60.36**±1.12*

The boldfaced and the underlined score denotes the best and the second-best result, respectively. Numbers in the lower right corner denote standard deviations. "*"indicates the statistically significant improvements (one-tailed t-test with $p < 0.05$) upon the best baseline.

within 2014–2016/2016–2018/2018–2020 as testing sets. For OGB-Proteins dataset, since the interactions between proteins can vary from different species that the proteins come from, we split the protein nodes into training/validation/test sets according to their species. We assume the test nodes are *strictly unseen* during training stage, which is more common in practice and more challenging than the default setting of OGB [33].

The experimental results are presented in Table 6. Our proposed method consistently achieves the best performance, indicating that **INL** can well handle distribution shifts existing in real-world scenarios. For example, **INL** increases the classification accuracy by 3.41% on OGB-Arxiv (tested on 2016–2018 with GraphSAGE backbone) and ROC-AUC by 7.54% on OGB-Proteins (tested on species-1 with GAT backbone) against the strongest baselines, respectively. Besides, different datasets have different distribution shifts, and none of the baselines can consistently achieve promising OOD generalized performance as our method. Therefore, the results show that our proposed method can well handle diverse types of distribution shifts in real graph datasets.

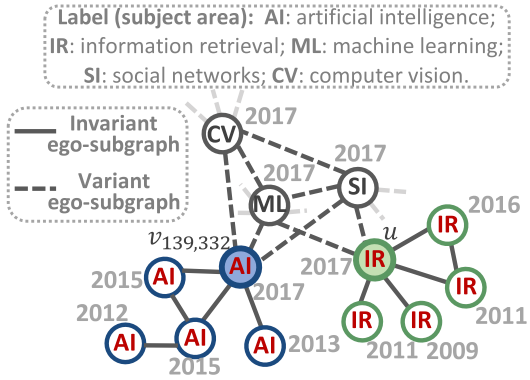


Fig. 5. The learned invariant and variant ego-subgraphs of the papers v and u from OGB-Arxiv.

Besides the quantitative evaluation, we plot a showcase from the OGB-Arxiv to intuitively validate the effectiveness of our method. Figure 5 shows that the learned invariant ego-subgraph G_v^I (denoted by solid lines) and variant ego-subgraph G_v^S (denoted by dashed lines) of one node v (ID: 139,332). We plot the top-five selected edges by the masks for simplicity. It can be observed that the invariant ego-subgraph G_v^I learned by our method accurately corresponds to the neighbors in the ego-graph from the same subject area (i.e., artificial intelligence), which have truly predictive and invariant relations with the centered node. However, the variant ego-subgraph G_v^S highlights the neighbors that are from different subject areas that are published in the same year with the centered node and have a high citation index (spurious feature). Besides, there is another paper u whose subject area is **information retrieval (IR)** that also cites those papers with high citation indexes, meaning that the node u has similar variant patterns with node v so they are in the same environment. We can observe that these nodes form clear cluster structures based on the variant ego-subgraphs, demonstrating the effectiveness of the proposed graph clustering algorithm in inferring latent environments.

4.4 Analysis of Node Environment Inference

In our proposed model, all components are jointly optimized. To show that the node environment inference module and invariance regularization module can mutually promote each other, we record the test accuracy, the modularity, which is a measurement for the quality of graph cluster, and **the normalized mutual information (NMI)** [41], which is another metric (falling within the range $[0, 1]$) for evaluating the clustering accuracy, as the model is trained. The results on Citeseer ($r_{train} = 0.7$) are shown in Figure 6. We can observe that the test accuracy and the modularity (clustering properties) improve synchronously over training. The results show that, as the training stage progresses, the invariant ego-subgraph generator is optimized so it can generate more informative invariant ego-subgraphs and therefore improve the performance on the testing set. However, accurately discovering invariant ego-subgraphs can also promote identifying variant ego-subgraphs, which capture the environment-discriminate features and better infer the latent environments. In addition, we observe that the test accuracy and the NMI (clustering accuracy) also improve collectively over training. Notice that **INL** achieves such results without needing any ground-truth environment label.

These empirical results well support the following points: (1) The invariant and variant patterns widely exist in real-world graphs, and our proposed **INL** can well identify invariant/variant ego-subgraphs under distribution shifts with multiple latent environments. (2) The variant

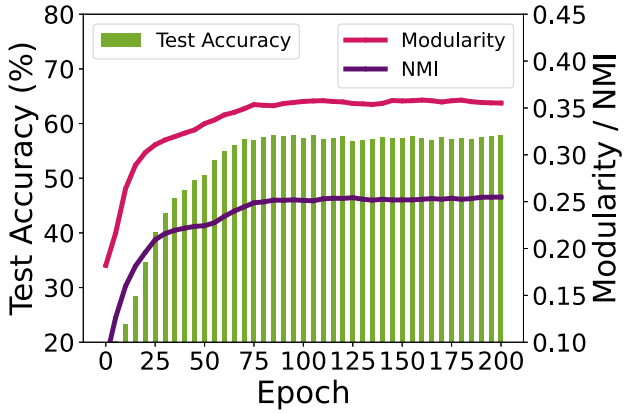


Fig. 6. The test accuracy and the performance of environment inference w.r.t. training epochs.

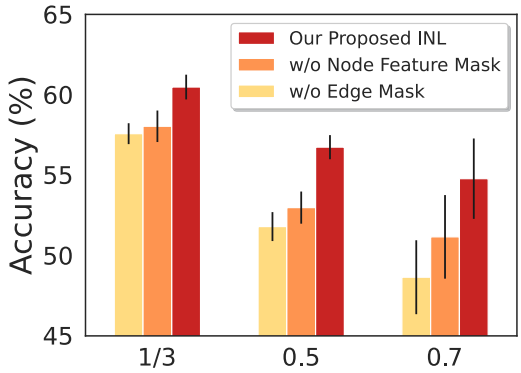
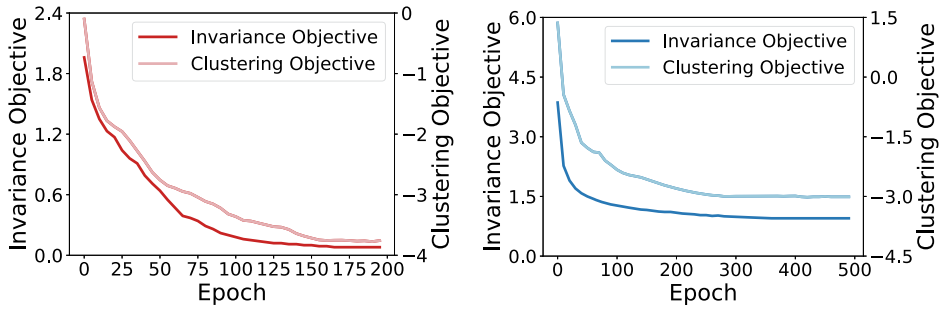


Fig. 7. Ablation studies of our method. We plot the accuracy (%) on the Citeseer datasets with different strengths of spurious correlations.

ego-subgraphs form clear clustering structures, and our **INL** can capture such patterns to accurately infer the environment labels of nodes. (3) Based on the inferred environments, our **INL** learns node representations by the invariant ego-subgraph for each node so it can achieve better OOD generalization performance. The environment inference and invariance regularization module can mutually enhance each other.

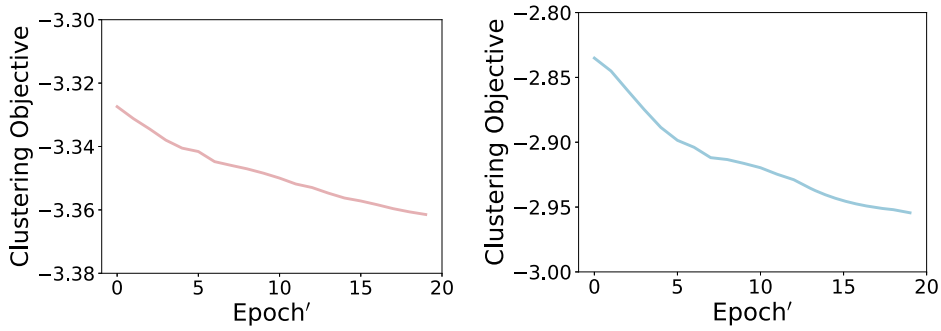
4.5 Ablation Studies

We perform ablation studies over the key components of the invariant ego-subgraph generator Ψ, i.e., masks on node features and edges, to understand their functionalities more deeply. We compare **INL** with the following two ablated versions: (1) w/o node feature mask: It removes the node feature mask by setting both invariant and variant node features in the ego-graph G_v to X_v, i.e., $X_v^I = X_v^S = X_v$. (2) w/o edge mask: It removes the edge mask by setting both invariant and variant edges in the ego-graph G_v to A_v, i.e., $A_v^I = A_v^S = A_v$. The results of the two ablated versions drop compared with **INL**, as shown in Figure 7. The performance gaps between **INL** and the two ablated versions become more significant as the degree of distribution shift increases (i.e., r_{train} from 1/3 to 0.7), which demonstrates the significance of accurately identifying the invariant node features and edges by the learnable masks.



(a) Citeseer (b) OGB-Arxiv

Fig. 8. The invariance objective and clustering objective in the training process on two datasets.



(a) Citeseer (b) OGB-Arxiv

Fig. 9. The clustering objective in one epoch of the training process on two datasets.

4.6 Training Dynamics

We can observe the convergence of our proposed method empirically, although the clustering objective in environment inference (i.e., Equation (4)) and invariance objective in invariance regularization (i.e., Equation (8)) are iteratively optimized. In Figures 8(a) and (b), we show the two objectives in the training process on Citeseer ($r_{train} = 0.7$) and OGB-Arxiv, respectively. The loss converges before reaching the maximal training epoch, while the results on the other datasets show similar patterns.

In Figure 9, we also show the objective of the inner iteration in Algorithm 1, i.e., the training dynamics of the clustering objective in one epoch of the outer iteration. The epoch of the outer iteration is specified as 100 and 250 for Citeseer ($r_{train} = 0.7$) and OGB-Arxiv, respectively, which is the middle of the whole training process, while the results in other epochs of the outer iteration show similar patterns.

4.7 Time Complexity Analysis

The time complexity of the proposed **INL** is $O(|E|\,d + |V|\,d^2)$, where $|V|$ and $|E|$ denote the number of nodes and edges, respectively, and d is the dimensionality of the node representations. Specifically, we adopt the message-passing GNN, which has a complexity of $O(|E|\,d + |V|\,d^2)$, to instantiate the GNN components in **INL**, and the GNNs are *shared* for all ego-graphs. Since we only need to generate mask for the existing edges in graphs, the time complexity of generating invariant and variant ego-subgraphs and further obtaining their representations is $O(|E|\,d + |V|\,d^2)$. The

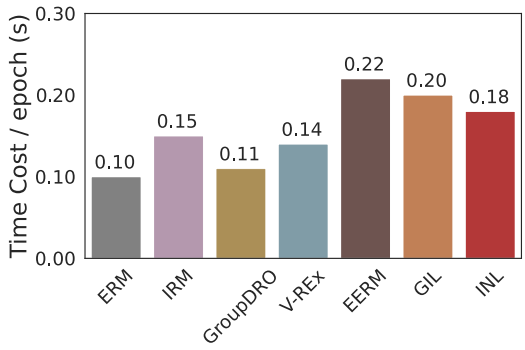


Fig. 10. The comparisons of empirical time cost per epoch during training our method and baselines on Citeseer ($r_{train} = 0.7$).

Table 7. Results (ROC-AUC, %) of Discovering the Ground-truth Invariant Node Features and Edges on Citeseer

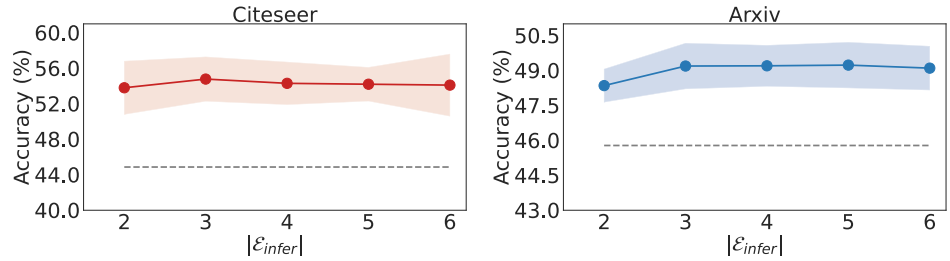
	Node Feature Mask			Edge Mask		
r_{train}	$r = 1/3$	$r = 0.5$	$r = 0.7$	$r = 1/3$	$r = 0.5$	$r = 0.7$
GNNExplainer	61.75±2.38	50.18±3.09	40.87±4.19	77.30±3.91	67.09±4.15	51.94±7.10
INL	**68.04**±2.19	**69.18**±2.06	**70.16**±2.54	**78.68**±3.10	**79.09**±3.21	**80.51**±3.13

time complexity of calculating the modularity matrix B in environment inference is $O(|E|\,(d + |\mathcal{E}_{infer}|) + |V|\,(d + |\mathcal{E}_{infer}|)^2)$, where $|\mathcal{E}_{infer}|$ denotes the number of inferred environments. The time complexity of the invariance regularizer is $O(|\mathcal{E}_{infer}|d^2)$, as the number of parameters for most GNNs is $O(d^2)$. Since $|\mathcal{E}_{infer}|$ are small constants, the overall time complexity of **INL** is $O(|E|\,d + |V|\,d^2)$. In comparison, the time complexity of other GNN-based node representation methods is also $O(|E|\,d + |V|\,d^2)$. Therefore, the time complexity of our proposed **INL** is on par with the existing methods.

In addition to the analysis of the time complexity, the empirical time cost of the proposed method and baselines are also tested. We show the results on Citeseer ($r_{train} = 0.7$) in Figure 10 while the results on other datasets show similar patterns. The results indicate that **INL** does not introduce infeasible time cost for achieving the best performances in practice. Its time cost for each training epoch is comparable with the baselines and more efficient than some competitive methods, demonstrating the efficiency and effectiveness of our method.

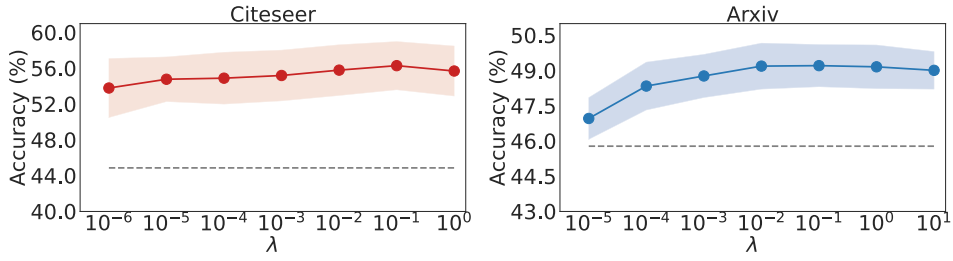
4.8 Comparisons with GNNExplainer

We compare the output invariant node features and structures generated by the proposed **INL** and GNNExplainer [88] with the ground-truth on the synthetic dataset Citeseer. Specifically, we generate post hoc explanations from GNNExplainer as the identified invariant ego-subgraphs, where we use the models trained under ERM as the models to explain. The evaluation metric is ROC-AUC. The results in Table 7 show that the masks on invariant node features and edges generated by GNNExplainer can be easily affected by the spurious correlations. Moreover, even when spurious correlations do not exist, the ROC-AUC of masks on invariant node features and edges generated by our **INL** still outperforms the result of the explainability method GNNExplainer, showing the effectiveness of **INL** when identifying invariant patterns.



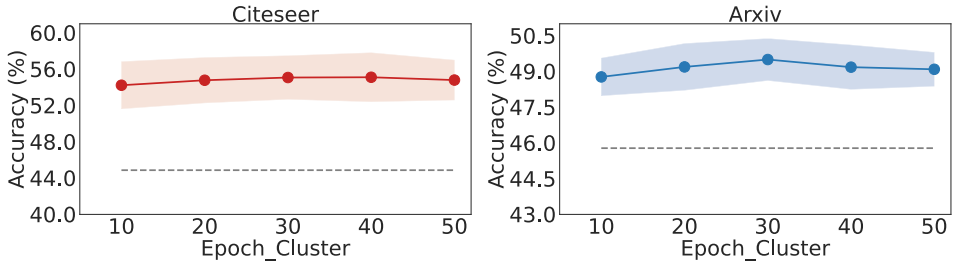
(a) Results on Citeseer (b) Results on OGB-Arxiv

Fig. 11. Impact of the number of inferred environment $|\mathcal{E}_{infer}|$. Red and blue lines denote the results of our **INL** , and grey dashed lines are the best results of all baselines.



(a) Results on Citeseer (b) Results on OGB-Arxiv

Fig. 12. Impact of the invariance regularizer coefficient λ. Red and blue lines denote the results of our **INL** , and grey dashed lines are the best results of all baselines.



(a) Results on Citeseer (b) Results on OGB-Arxiv

Fig. 13. Impact of the number of epochs for clustering to infer environments in each training epoch (i.e., Epoch_Cluster in Algorithm 1). Red and blue lines denote the results of our **INL** , and grey dashed lines are the best results of all baselines.

4.9 Hyper-parameter Sensitivity

We investigate the sensitivity of hyper-parameters of our method, including the number of inferred environments $|\mathcal{E}_{infer}|$, the invariance regularizer coefficient λ, and the number of epochs for clustering to infer environments in each training epoch (i.e., Epoch_Cluster in Algorithm 1). For simplicity, we only report the results on Citeseer ($r_{train} = 0.7$) and OGB-Arxiv (2016–2018 with GraphSage backbone) in Figures 11–13, while the results on other datasets show similar patterns.

First, the number of inferred environments has a slight impact on the model performance. For Citeseer, the performance reaches a peak when $|\mathcal{E}_{infer}| = 3$, showing that **INL** achieves the best

result when the number of environments matches the ground truth. For OGB-Arxiv, the best number of environments is $|\mathcal{E}_{infer}| = 5$. A plausible reason is that OGB-Arxiv dataset consists of more nodes and edges, which form more environments than Citeseer. Second, we also find the coefficient λ has a slight influence on the performance, indicating that we need to properly balance the classification loss and the invariance regularizer term. Finally, a proper value of the hyper-parameter Epoch_Cluster is important. A small value may not be sufficient to infer the environments accurately, while a very large value is unnecessary and may affect the training efficiency. Although an appropriate choice of hyper-parameters can further improve the performance, our method is not very sensitive to hyper-parameters. Figures 11–13 show that **INL** can outperform the best baselines with a wide range of hyper-parameters choices.

5 RELATED WORKS

In this section, we review the related works of node representation learning, generalization of GNNs, explainability of GNNs, invariant learning, and modularity.

5.1 Node Representation Learning

Node representation learning on graphs has been extensively studied, such as random-walk based methods [19, 29, 63] and matrix factorization-based methods [10, 12, 62]. Recently, **graph neural networks (GNNs)** [28, 38, 75] have revolutionized the field of node representation learning [96]. They generally utilize a neighborhood aggregation (or message passing) paradigm to capture the structural information within nodesâĂŹ neighborhood. The message passing of the tth layer in GNNs is usually denoted as:

$$\mathbf{Z}_v^{(t)} = \text{COMBINE}^{(t)}\left(\mathbf{Z}_v^{(t-1)}, \mathbf{m}_v^{(t)}\right), \quad \mathbf{m}_v^{(t)} = \text{AGGREGATION}^{(t)}\left(\{\mathbf{Z}_u^{(t-1)}\}\right), \tag{25}$$

where u is the neighbor of node v. $\mathbf{Z}_v^{(t)}$ represents the embedding of node v at the tth layer, and $\mathbf{Z}_v^{(0)}$ is initialized with the input node feature. $\mathbf{m}_v^{(t)}$ represents the aggregated message from the neighbors of node v. $\text{COMBINE}^{(t)}(\cdot)$ and $\text{AGGREGATION}^{(t)}(\cdot)$ are the combination and aggregation functions of GNNs [89]. Many GNNs and their variants [30, 46, 53, 59, 90, 98] have been proposed, achieving state-of-the-art performance on various tasks and demonstrating profound successes in challenging applications, such as recommendation systems [9, 26, 31, 77, 83], information retrieval [17, 91, 95], drug discovery [18, 80], protein function prediction [33, 36], traffic forecasting [21, 37], and so on. However, most existing GNNs do not consider the out-of-distribution generalization ability, so their performances drop substantially on testing data with distribution shifts [33, 44, 80].

5.2 Generalization of GNNs

A few recent works begin to study the generalization ability of GNNs. The early works [27, 48, 66, 76] focus on the generalization bounds over the training distribution, i.e., in-distribution generalization, which is orthogonal to the OOD generalization and not suitable for the distribution shifts studied in this article. More recently, the OOD generalization ability of GNNs starts to receive research interest [7, 39, 43, 58, 79, 82, 87]. In particular, Bevilacqua et al. [7] learn size-invariant representations for tackling the distribution shifts that exist on graph size. DIR [79] is proposed to discover invariant rationales for GNNs. GIL [45] focuses on capturing the invariant relationships between predictive graph structural information and labels under distribution shifts for OOD generalization. These works mostly concentrate on graph-level tasks and largely ignore the more challenging node-level tasks with multiple latent environments. Some works [24, 54, 99] are proposed to deal with semi-supervised node classification under non-I.I.D. setting. They focus on the

adaptation ability of GNNs under distribution shifts, i.e., transferring GNN models trained on the source domain (i.e., environment) to the related target domain with different distributions. For example, SR-GNN [99] is proposed to handle distribution shifts between the selected training and testing nodes by adopting CMD [93] and importance sampling. Reference [24] proposes to learn GNN models by considering agnostic label selection bias. However, these works assume that test data are available and will participate in the training process, which is not in the scope of the OOD generalization problem studied in this article. One exception is the very recent pioneering work EERM [78], which studies invariant node learning by assuming all nodes share a single environment. However, it ignores the more common and challenging situation that nodes are from multiple latent environments. We empirically show that our proposed method greatly outperforms EERM by effectively identifying and modeling multiple latent environments.

5.3 Explainability of GNNs

The studies on the explainability of GNNs aim to understand the predictions of black-box GNNs by providing the explanations [20, 72, 92]. They generally try to answer which nodes, edges, or features of the input graph are more important for predicting the labels. Several works are proposed to find a subgraph structure and a small subset of node features for the target nodes as the explanations for GNN's predictions [49, 52, 88]. For example, GNNExplainer [88] learns the soft masks on edges and node features to explain the predictions with the mask optimization. PGExplainer [52] further learns the approximated discrete masks on edges to explain the predictions with a parameterized mask predictor. GraphMask [68] is a post hoc method for explaining the importance of edges in the graph convolution layer. A recent work [79] finds that these explainability works are very sensitive to distribution shifts as most GNN models and proposes discovering invariant explanations in graph-level classification tasks. However, these works focus on understanding the predictions of GNNs instead of learning node representations for better generalization ability under distribution shifts.

5.4 Invariant Learning

Invariant learning has received surging attentions to enable OOD generalization, aiming to generalize to unseen environments by exploiting the invariant relationships between features and labels across distribution shifts. Several works [2, 4, 11, 40, 42, 64] are proposed to learn invariant model and show guaranteed generalization under distribution shifts. However, most existing methods heavily rely on additional environment labels that have to be explicitly provided in the training dataset. Such annotations for the nodes on graph data are usually unavailable and prohibitively expensive to collect, so that these invariant learning methods are inapplicable. A few works study OOD generalization on latent environments in computer vision [16, 51, 56], which cannot be directly applied to graph data. In summary, how to learn invariant node representations under distribution shifts without explicit environment labels remains largely unexplored in the literature.

5.5 Modularity

The Modularity is generally used to measure the divergence between the number of intra-cluster edges and the expected number of a random graph [60], where nodes v and u with degrees d_v and d_u are connected with probability $d_v d_u / 2m$ and m is the edge number. By maximizing the modularity, the nodes are densely connected within each cluster [73]:

$$\max_{C} Q = \frac{1}{2m} \text{trace} \left(C^\top A C - \frac{1}{2m} \text{diag} \left(C^\top \mathbf{d} \mathbf{d}^\top C \right) \right), \tag{26}$$

where C is a cluster assignment matrix, and A is the adjacency matrix of the input graph for clustering. $\mathbf{d}$ and m indicate the degree vector and the number of edges, respectively. However, there are two obstacles for directly adopting this classical modularity maximization method to learn cluster assignment as the inferred environments. The first is that the modularity maximization ignores the inter-cluster edges whose connecting probability should be minimized in the meantime. The second is that we should use the variant patterns (X^S, A^S) of the input graph for clustering rather than use the whole input graph (X, A). Since the invariant patterns capture the invariant relationships between predictive node features and graph structures with the node labels, the variant patterns in turn capture variant spurious correlations under different distributions.

6 CONCLUSIONS

In this article, we study learning invariant node representations under distribution shifts with multiple latent environments and propose a principled and novel method (**INL**). The proposed method can identify the invariant and variant ego-subgraphs of nodes, infer the environment label of nodes without supervisions, and learn invariant node representations through regularization. Extensive experiments on both synthetic and real-world node classification benchmarks demonstrate the superiority of our method against state-of-the-art baselines when there exist distribution shifts.

APPENDIX

A PROOFS

A.1 Proof of Proposition 3

Proof. Let $a_v^{I,I} = \frac{1}{|\mathcal{N}_v^I|}\sum_{u\in\mathcal{N}_v^I} x_u^I$ be the aggregated invariant node features from invariant ego-subgraph G_v^I. Similarly, we define $a_v^{S,I} = \frac{1}{|\mathcal{N}_v^I|}\sum_{u\in\mathcal{N}_v^I} x_u^S$, $a_v^{I,S} = \frac{1}{|\mathcal{N}_v^S|}\sum_{u\in\mathcal{N}_v^S} x_u^I$, and $a_v^{S,S} = \frac{1}{|\mathcal{N}_v^S|}\sum_{u\in\mathcal{N}_v^S} x_u^S$. The first and second superscript of a_v indicate the invariant/variant node features and structures, respectively. We further denote $e_v^I = \frac{1}{|\mathcal{N}_v^I|}\sum_{u\in\mathcal{N}_v^I} e_u$, and $e_v^S = \frac{1}{|\mathcal{N})_v^S|}\sum_{u\in\mathcal{N}_v^S} e_u$. The risk of predictor f is:

$$
\begin{aligned}
\mathcal{R} &= \frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\mathrm{y_v}|\mathrm{G_v}=G_v}\left[||\hat{y}_v - y_v||_2^2\right] \\
&= \frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\epsilon_1,\epsilon_2}\left[||\left(\theta_1 a_v^{I,I} + \theta_2 a_v^{S,I} + \theta_3 a_v^{I,S} + \theta_4 a_v^{S,S}\right) - \left(a_v^{I,I} + \epsilon_1\right)||_2^2\right] \\
&= \frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\epsilon_1,\epsilon_2}\left[||\left(\theta_1+\theta_2-1\right)a_v^{I,I} + \left(\theta_3+\theta_4\right)a_v^{I,S} + \theta_2\left(\epsilon_1+\epsilon_2+e_v^I\right) + \theta_4\left(\epsilon_1+\epsilon_2+e_v^S\right) - \epsilon_1||_2^2\right]. \quad (27)
\end{aligned}
$$

The first-order derivative w.r.t. θ_1 is:

$$
\begin{aligned}
&\frac{\partial\mathcal{R}}{\partial\theta_1} \\
&= \frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\epsilon_1,\epsilon_2}\left[2\left(\left(\theta_1+\theta_2-1\right)a_v^{I,I} + \left(\theta_3+\theta_4\right)a_v^{I,S} + \theta_2\left(\epsilon_1+\epsilon_2+e_v^I\right) + \theta_4\left(\epsilon_1+\epsilon_2+e_v^S\right) - \epsilon_1\right)a_v^{I,I}\right] \\
&= \frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\epsilon_1,\epsilon_2}\left[2\left(\left(\theta_1+\theta_2-1\right)a_v^{I,I}a_v^{I,I} + \left(\theta_3+\theta_4\right)a_v^{I,I}a_v^{I,S}\right)\right], \quad (28)
\end{aligned}
$$

where the second equation holds because $a_v^{I,I}$ is independent with ϵ_1, ϵ_2, e_v^I, and e_v^S. Therefore, let $\frac{\partial\mathcal{R}}{\partial\theta_1} = 0$, we have

$$
\frac{1}{|V|}\sum_{v\in V}\mathbb{E}_{\epsilon_1,\epsilon_2}\left[\left(\theta_1+\theta_2-1\right)a_v^{I,I}a_v^{I,I} + \left(\theta_3+\theta_4\right)a_v^{I,I}a_v^{I,S}\right] = 0. \quad (29)
$$

The first-order derivative w.r.t. θ_2 is:

$$\begin{aligned}
&\frac{\partial \mathcal{R}}{\partial \theta_2} \\
&= \frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,S} + \theta_2 (\epsilon_1 + \epsilon_2 + e_v^I) + \theta_4 \left(\epsilon_1 + \epsilon_2 + e_v^S \right) - \epsilon_1 \right) \left(a_v^{I,I} + \epsilon_1 + \epsilon_2 + e_v^I \right) \right] \\
&= \frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,I} a_v^{I,S} + \theta_2 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^I \right) + \theta_4 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^S \right) - 1 \right) \right] \\
&= \frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left(\theta_2 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^I \right) + \theta_4 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^S \right) - 1 \right) \right] \\
&= \frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left(\theta_2 \left(2 + e_v^I e_v^I \right) + \theta_4 \left(2 + e_v^I e_v^S \right) - 1 \right) \right], \qquad (30)
\end{aligned}$$

where the second equation holds because of the independence among $a_v^{I,I}$, ϵ_1, ϵ_2, and e_v^I or e_v^S. The third equation holds, since we let $\frac{\partial \mathcal{R}}{\partial \theta_1} = 0$. The last equation holds, since ϵ_1 and ϵ_2 follow standard normal distribution. We further let $\frac{\partial \mathcal{R}}{\partial \theta_2} = 0$ and obtain:

$$\frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[\theta_2 \left(2 + e_v^I e_v^I \right) + \theta_4 \left(2 + e_v^I e_v^S \right) - 1 \right] = 0. \qquad (31)$$

Similarly, let $\frac{\partial \mathcal{R}}{\partial \theta_3} = 0$, we have

$$\frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[(\theta_1 + \theta_2 - 1)\, a_v^{I,I} a_v^{I,S} + (\theta_3 + \theta_4)\, a_v^{I,S} a_v^{I,S} \right] = 0. \qquad (32)$$

And let $\frac{\partial \mathcal{R}}{\partial \theta_4} = 0$, we have

$$\frac{1}{|V|} \sum_{v \in V} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[\theta_2 \left(2 + e_v^I e_v^S \right) + \theta_4 \left(2 + e_v^S e_v^S \right) - 1 \right] = 0. \qquad (33)$$

Finally, given Equations (29) and (31)–(33), we can derive the solution:

$$\theta_1 = 1 - \frac{\mu^S}{2(\mu^S - \mu^I)}, \quad \theta_2 = \frac{\mu^S}{2(\mu^S - \mu^I)}, \quad \theta_3 = \frac{\mu^I}{2(\mu^S - \mu^I)}, \quad \theta_4 = \frac{-\mu^I}{2(\mu^S - \mu^I)}. \qquad (34)$$

□

A.2 Proof of Proposition 4

Proof. If the invariance regularizer $\text{trace}(\text{Var}_{\mathcal{E}_{infer}}(\nabla_\theta \mathcal{R}^e))$ in Equation (8) reaches the minimum, then we have $\text{trace}(\text{Var}_{\mathcal{E}_{infer}}(\nabla_\theta \mathcal{R}^e)) = 0$. It means that the variance of $\frac{\partial \mathcal{R}^e}{\partial \theta_i}$ among all environments is 0, i.e., $\frac{\partial \mathcal{R}^e}{\partial \theta_i}$ keeps invariant between any two environments, $i = 1, 2, 3, 4$. Recall that

$$\begin{aligned}
&\frac{\partial \mathcal{R}^e}{\partial \theta_1} \\
&= \frac{1}{|V^e|} \sum_{v \in V^e} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,S} + \theta_2 \left(\epsilon_1 + \epsilon_2 + e_v^I \right) + \theta_4 \left(\epsilon_1 + \epsilon_2 + e_v^S \right) - \epsilon_1 \right) a_v^{I,I} \right] \\
&= \frac{1}{|V^e|} \sum_{v \in V^e} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,I} a_v^{I,S} \right) \right] \qquad (35)
\end{aligned}$$

and

$$\begin{aligned}
&\frac{\partial \mathcal{R}^e}{\partial \theta_2} \\
&= \frac{1}{|V^e|} \sum_{v \in V^e} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,S} + \theta_2 \left(\epsilon_1 + \epsilon_2 + e_v^I \right) + \theta_4 \left(\epsilon_1 + \epsilon_2 + e_v^S \right) - \epsilon_1 \right) \left(a_v^{I,I} + \epsilon_1 + \epsilon_2 + e_v^I \right) \right] \\
&= \frac{1}{|V^e|} \sum_{v \in V^e} \mathbb{E}_{\epsilon_1, \epsilon_2} \left[2 \left((\theta_1 + \theta_2 - 1)\, a_v^{I,I} a_v^{I,I} + (\theta_3 + \theta_4)\, a_v^{I,I} a_v^{I,S} + \theta_2 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^I \right) + \theta_4 \left(\epsilon_1^2 + \epsilon_2^2 + e_v^I e_v^S \right) - 1 \right) \right].
\end{aligned}$$

Therefore, $\frac{\partial \mathcal{R}^e}{\partial \theta_i}$ can keep invariant between any two environments for $i = 1, 2, 3, 4$, only when satisfying $\theta_3 + \theta_4 = 0$, $\theta_2 = 0$, and $\theta_4 = 0$, Finally, optimizing the invariance regularizer in Equation (8) to the minimum can lead to $[\theta_2, \theta_3, \theta_4] = [0, 0, 0]$, so the model can make predictions only based on the invariant patterns and achieve promising OOD generalization under distribution shifts. □

REFERENCES

[1] Abien Fred Agarap. 2018. Deep learning using rectified linear units (ReLU). *arXiv preprint arXiv:1803.08375* (2018).

[2] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. 2021. Invariance principle meets information bottleneck for out-of-distribution generalization. In *Neural Information Processing Systems (NeurIPS)*.

[3] Kartik Ahuja, Jun Wang, Amit Dhurandhar, Karthikeyan Shanmugam, and Kush R. Varshney. 2021. Empirical or invariant risk minimization? A sample complexity perspective. In *International Conference on Learning Representations*.

[4] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893* (2019).

[5] Albert-Laszlo Barabasi and Zoltan N. Oltvai. 2004. Network biology: Understanding the cell's functional organization. *Nat. Rev. Genet.* 5, 2 (2004), 101–113.

[6] Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. 2019. A meta-transfer objective for learning to disentangle causal mechanisms. In *International Conference on Learning Representations*.

[7] Beatrice Bevilacqua, Yangze Zhou, and Bruno Ribeiro. 2021. Size-invariant graph representations for graph classification extrapolations. In *38th International Conference on Machine Learning*. 837–851.

[8] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. 2006. Maximizing modularity is hard. *arXiv preprint physics/0608255* (2006).

[9] Desheng Cai, Shengsheng Qian, Quan Fang, Jun Hu, and Changsheng Xu. 2022. User cold-start recommendation via inductive heterogeneous graph neural network. *ACM Trans. Inf. Syst.* 41, 3 (2022).

[10] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *24th ACM International on Conference on Information and Knowledge Management (CIKM'15)*. 891–900.

[11] Shiyu Chang, Yang Zhang, Mo Yu, and Tommi Jaakkola. 2020. Invariant rationalization. In *International Conference on Machine Learning*. PMLR, 1448–1458.

[12] Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient neural matrix factorization without sampling for recommendation. *ACM Trans. Inf. Syst.* 38, 2 (2020), 1–28.

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*.

[14] Xu Chen, Kun Xiong, Yongfeng Zhang, Long Xia, Dawei Yin, and Jimmy Xiangji Huang. 2020. Neural feature-aware recommendation with signed hypergraph convolutional network. *ACM Trans. Inf. Syst.* 39, 1 (2020), 1–22.

[15] Gene Ontology Consortium. 2019. The gene ontology resource: 20 years and still GOing strong. *Nucleic Acids Res.* 47, D1 (2019).

[16] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. 2021. Environment inference for invariant learning. In *International Conference on Machine Learning*. PMLR, 2189–2200.

[17] Hejie Cui, Jiaying Lu, Yao Ge, and Carl Yang. 2022. How can graph neural networks help document retrieval: A case study on CORD19 with concept map generation. In *European Conference on Information Retrieval*. Springer, 75–83.

[18] Limeng Cui and Dongwon Lee. 2022. KETCH: Knowledge graph enhanced thread recommendation in healthcare forums. In *45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 492–501.

[19] Peng Cui, Shao-Wei Liu, Wen-Wu Zhu, Huan-Bo Luan, Tat-Seng Chua, and Shi-Qiang Yang. 2014. Social-sensed image search. *ACM Trans. Inf. Syst.* 32, 2 (2014), 1–23.

[20] Enyan Dai and Suhang Wang. 2021. Towards self-explainable graph neural network. In *30th ACM International Conference on Information & Knowledge Management (CIKM'21)*. 302–311.

[21] Austin Derrow-Pinion, Jennifer She, David Wong, Oliver Lange, Todd Hester, Luis Perez, Marc Nunkesser, Seongjae Lee, Xueying Guo, Brett Wiltshire, et al. 2021. ETA prediction with graph neural networks in Google Maps. In *30th ACM International Conference on Information & Knowledge Management (CIKM'21)*. 3767–3776.
[22] David Easley, Jon Kleinberg, et al. 2012. *Networks, Crowds, and Markets*. Cambridge Books.
[23] Abbas El Gamal and Young-Han Kim. 2011. *Network Information Theory*. Cambridge University Press.
[24] Shaohua Fan, Xiao Wang, Chuan Shi, Kun Kuang, Nian Liu, and Bai Wang. 2022. Debiased graph neural networks with agnostic label selection bias. *IEEE Trans. Neural Netw. Learn. Syst.* (2022).
[25] Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.
[26] Chen Gao, Xiang Wang, Xiangnan He, and Yong Li. 2022. Graph neural networks for recommender system. In *15th ACM International Conference on Web Search and Data Mining (WSDM'22)*. 1623–1625.
[27] Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. 2020. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*. PMLR, 3419–3430.
[28] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*. PMLR, 1263–1272.
[29] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 855–864.
[30] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *31st International Conference on Neural Information Processing Systems*. 1025–1035.
[31] Xiangnan He, Zhaochun Ren, Emine Yilmaz, Marc Najork, and Tat-Seng Chua. 2021. Graph technologies for user modeling and recommendation: Introduction to the special issue—Part 1. *ACM Trans. Inf. Syst.* 40, 2 (Sep. 2021). DOI: https://doi.org/10.1145/3477596
[32] Kanglin Hsieh et al. 2021. Drug repurposing for COVID-19 using graph neural network and harmonizing multiple evidence. *Scient. Rep.* 11, 1 (2021).
[33] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Adv. Neural Inf. Process. Syst.* 33 (2020), 22118–22133.
[34] Kexin Huang and Marinka Zitnik. 2020. Graph meta learning via local subgraphs. *Neural Inf. Process. Syst.* 33 (2020).
[35] Liwei Huang, Yutao Ma, Yanbo Liu, Bohong Danny Du, Shuliang Wang, and Deyi Li. 2021. Position-enhanced and time-aware graph convolutional network for sequential recommendations. *ACM Trans. Inf. Syst.* 41, 1 (2021).
[36] Biaobin Jiang, Kyle Kloster, David F. Gleich, and Michael Gribskov. 2017. AptRank: An adaptive PageRank model for protein function prediction on bi-relational graphs. *Bioinformatics* 33, 12 (2017), 1829–1836.
[37] Weiwei Jiang and Jiayun Luo. 2021. Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174* (2021).
[38] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.
[39] Boris Knyazev, Graham W. Taylor, and Mohamed Amer. 2019. Understanding attention and generalization in graph neural networks. *Adv. Neural Inf. Process. Syst.* 32 (2019), 4202–4212.
[40] Masanori Koyama and Shoichiro Yamaguchi. 2020. Out-of-distribution generalization with maximal invariant predictor. *arXiv preprint arXiv:2008.01883* (2020).
[41] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Phys. Rev. E* 69, 6 (2004), 066138.
[42] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-distribution generalization via risk extrapolation (REX). In *International Conference on Machine Learning*.
[43] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. 2022. OOD-GNN: Out-of-distribution generalized graph neural network. *IEEE Trans. Knowl. Data Eng.* (2022).
[44] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. 2022. Out-of-distribution generalization on graphs: A survey. *arXiv preprint arXiv:2202.07987* (2022).
[45] Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu. 2022. Learning invariant graph representations for out-of-distribution generalization. In *Advances in Neural Information Processing Systems*.
[46] Jianxin Li, Hao Peng, Yuwei Cao, Yingtong Dou, Hekai Zhang, Philip Yu, and Lifang He. 2021. Higher-order attribute-enhancing heterogeneous graph neural networks. *IEEE Trans. Knowl. Data Eng.* 35, 1 (2021).
[47] Yang Li, Buyue Qian, Xianli Zhang, and Hui Liu. 2020. Graph neural network-based diagnosis prediction. *Big Data* 8, 5 (2020), 379–390.
[48] Renjie Liao, Raquel Urtasun, and Richard Zemel. 2020. A PAC-Bayesian approach to generalization bounds for graph neural networks. In *International Conference on Learning Representations*.

[49] Wanyu Lin, Hao Lan, and Baochun Li. 2021. Generative causal explanations for graph neural networks. In *International Conference on Machine Learning*. PMLR, 6666–6679.
[50] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Heterogeneous risk minimization. In *International Conference on Machine Learning*. PMLR.
[51] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Integrated latent heterogeneity and invariance learning in kernel space. In *Advances in Neural Information Processing Systems*.
[52] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. 2020. Parameterized explainer for graph neural network. *Adv. Neural Inf. Process. Syst.* 33 (2020).
[53] Zihan Luo, Jianxun Lian, Hong Huang, Hai Jin, and Xing Xie. 2022. Ada-GNN: Adapting to local patterns for improving graph neural networks. In *15th ACM International Conference on Web Search and Data Mining (WSDM'22)*. 638–647.
[54] Jiaqi Ma, Junwei Deng, and Qiaozhu Mei. 2021. Subgroup generalization and fairness of graph neural networks. *Adv. Neural Inf. Process. Syst.* 34 (2021).
[55] Ting Ma, Longtao Huang, Qianqian Lu, and Songlin Hu. 2022. KR-GCN: Knowledge-aware reasoning with graph convolution network for explainable recommendation. *ACM Trans. Inf. Syst.* 41, 1 (2022).
[56] Toshihiko Matsuura and Tatsuya Harada. 2020. Domain generalization using a mixture of multiple latent domains. In *AAAI Conference on Artificial Intelligence*. 11749–11756.
[57] Miller McPherson, Lynn Smith-Lovin, and James M. Cook. 2001. Birds of a feather: Homophily in social networks. *Ann. Rev. Sociol.* 27, 1 (2001).
[58] Siqi Miao, Mia Liu, and Pan Li. 2022. Interpretable and generalizable graph learning via stochastic attention mechanism. In *International Conference on Machine Learning*.
[59] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2019. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *AAAI Conference on Artificial Intelligence*. 4602–4609.
[60] Mark E. J. Newman. 2006. Modularity and community structure in networks. *Proc. Nat. Acad. Sci.* 103, 23 (2006).
[61] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2015), 11–33.
[62] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1105–1114.
[63] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 701–710.
[64] Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. 2018. Invariant models for causal transfer learning. *J. Mach. Learn. Res.* 19, 1 (2018).
[65] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731* (2019).
[66] Franco Scarselli, Ah Chung Tsoi, and Markus Hagenbuchner. 2018. The Vapnik–Chervonenkis dimension of graph and recursive neural networks. *Neural Netw.* 108 (2018), 248–259.
[67] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.
[68] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. 2021. Interpreting graph neural networks for NLP with differentiable edge masking. In *International Conference on Learning Representations*.
[69] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868* (2018).
[70] Jonathan M. Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M. Donghia, Craig R. MacNair, Shawn French, Lindsey A. Carfrae, Zohar Bloom-Ackermann, et al. 2020. A deep learning approach to antibiotic discovery. *Cell* 180, 4 (2020), 688–702.
[71] Damian Szklarczyk, Annika L. Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T. Doncheva, John H. Morris, Peer Bork, et al. 2019. STRING v11: Protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.* 47, D1 (2019), D607–D613.
[72] Juntao Tan, Shijie Geng, Zuohui Fu, Yingqiang Ge, Shuyuan Xu, Yunqi Li, and Yongfeng Zhang. 2022. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In *ACM Web Conference*. 1018–1027.
[73] Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel Müller. 2020. Graph clustering with graph neural networks. *arXiv preprint arXiv:2006.16904* (2020).
[74] Vladimir N. Vapnik. 1999. An overview of statistical learning theory. *IEEE Trans. Neural Netw.* 10, 5 (1999), 988–999.

[75] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.
[76] Saurabh Verma and Zhi-Li Zhang. 2019. Stability and generalization of graph convolutional neural networks. In *25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1539–1548.
[77] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 169–178.
[78] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. 2022. Handling distribution shifts on graphs: An invariance perspective. In *International Conference on Learning Representations*.
[79] Yingxin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat-Seng Chua. 2022. Discovering invariant rationales for graph neural networks. In *International Conference on Learning Representations*.
[80] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. 2018. MoleculeNet: A benchmark for molecular machine learning. *Chem. Sci.* 9, 2 (2018), 513–530.
[81] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations*.
[82] Keyulu Xu, Mozhi Zhang, Jingling Li, Simon S. Du, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2021. How neural networks extrapolate: From feedforward to graph neural networks. In *International Conference on Learning Representations*.
[83] Jun Yang, Weizhi Ma, Min Zhang, Xin Zhou, Yiqun Liu, and Shaoping Ma. 2021. LegalGNN: Legal information enhanced graph neural network for recommendation. *ACM Trans. Inf. Syst.* 40, 2 (2021), 1–29.
[84] Tianchi Yang, Linmei Hu, Chuan Shi, Houye Ji, Xiaoli Li, and Liqiang Nie. 2021. HGAT: Heterogeneous graph attention networks for semi-supervised short text classification. *ACM Trans. Inf. Syst.* 39, 3 (2021), 1–29.
[85] Yiying Yang, Zhongyu Wei, Qin Chen, and Libo Wu. 2019. Using external knowledge for financial event prediction based on graph neural networks. In *28th ACM International Conference on Information and Knowledge Management (CIKM'19)*. 2161–2164.
[86] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*. PMLR, 40–48.
[87] Gilad Yehudai, Ethan Fetaya, Eli Meirom, Gal Chechik, and Haggai Maron. 2021. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*. PMLR, 11975–11986.
[88] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNexplainer: Generating explanations for graph neural networks. *Adv. Neural Inf. Process. Syst.* 32 (2019), 9240.
[89] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Adv. Neural Inf. Process. Syst.* 33 (2020), 5812–5823.
[90] Wenhui Yu, Xiao Lin, Jinfei Liu, Junfeng Ge, Wenwu Ou, and Zheng Qin. 2021. Self-propagation graph neural network for recommendation. *IEEE Trans. Knowl. Data Eng.* 34, 12 (2021).
[91] Xueli Yu, Weizhi Xu, Zeyu Cui, Shu Wu, and Liang Wang. 2021. Graph-based hierarchical relevance matching signals for ad-hoc retrieval. In *Web Conference*. 778–787.
[92] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. 2020. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445* (2020).
[93] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. 2017. Central moment discrepancy (CMD) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811* (2017).
[94] Ge Zhang, Zhao Li, Jiaming Huang, Jia Wu, Chuan Zhou, Jian Yang, and Jianliang Gao. 2022. eFraudCom: An e-commerce fraud detection system via competitive graph neural networks. *ACM Trans. Inf. Syst.* 40, 3 (2022), 1–29.
[95] Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR meets graph embedding: A ranking model for product search. In *World Wide Web Conference*. 2390–2400.
[96] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2020. Deep learning on graphs: A survey. *IEEE Trans. Knowl. Data Eng.* 34, 1 (2020).
[97] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transport. Syst.* 21, 9 (2019), 3848–3858.
[98] Meiqi Zhu, Xiao Wang, Chuan Shi, Houye Ji, and Peng Cui. 2021. Interpreting and unifying graph neural networks with an optimization framework. In *Web Conference*. 1215–1226.
[99] Qi Zhu, Natalia Ponomareva, Jiawei Han, and Bryan Perozzi. 2021. Shift-robust GNNs: Overcoming the limitations of localized graph training data. *Adv. Neural Inf. Process. Syst.* 34 (2021).
[100] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* 34, 13 (2018), i457–i466.

Received 20 September 2022; revised 11 April 2023; accepted 22 May 2023