Pruning Spurious Subgraphs for Graph Out-of-Distribution Generalization

Tianjun Yao[1] **Haoxuan Li**[1] **Yongqiang Chen**[1,2] **Tongliang Liu**[3,1]
Le Song[1] **Eric Xing**[1,2] **Zhiqiang Shen**[1]

[1]Mohamed bin Zayed University of Artificial Intelligence
[2]Carnegie Mellon University [3]The University of Sydney

{tianjun.yao,haoxuan.li,yongqiang.chen}@mbzuai.ac.ae
tongliang.liu@sydney.edu.au, {le.song,eric.xing,zhiqiang.shen}@mbzuai.ac.ae

Abstract

Graph Neural Networks (GNNs) often encounter significant performance degradation under distribution shifts between training and test data, hindering their applicability in real-world scenarios. Recent studies have proposed various methods to address the out-of-distribution (OOD) generalization challenge, with many methods in the graph domain focusing on directly identifying an invariant subgraph that is predictive of the target label. However, we argue that identifying the edges from the invariant subgraph directly is challenging and error-prone, especially when some spurious edges exhibit strong correlations with the targets. In this paper, we propose PrunE, the first pruning-based graph OOD method that eliminates spurious edges to improve OOD generalizability. By pruning spurious edges, PrunE retains the invariant subgraph more comprehensively, which is critical for OOD generalization. Specifically, PrunEemploys two regularization terms to prune spurious edges: 1) *graph size constraint* to exclude uninformative spurious edges, and 2) ϵ*-probability alignment* to further suppress the occurrence of spurious edges. Through theoretical analysis and extensive experiments, we show that PrunE achieves superior OOD performance and outperforms previous state-of-the-art methods significantly. Codes are available at: https://github.com/tianyao-aka/PrunE-GraphOOD.

1 Introduction

Graph Neural Networks (GNNs) [25, 61, 55] often encounter significant performance degradation under distribution shifts between training and test data, hindering their applicability in real-world scenarios [18, 19, 27]. To address the out-of-distribution (OOD) generalization challenge, recent studies propose to utilize the causally invariant mechanism to learn invariant features that remain stable across different environments [44, 2, 1, 23, 29, 10]. In graph domain, various methods have been proposed to address the OOD generalization problem [59, 34, 9, 38, 52, 16, 64], Most OOD methods, both in the general domain and the graph domain, aim to learn invariant features directly. To achieve this, many graph-specific OOD methods utilize a subgraph selector to model independent edge probabilities to directly identify invariant subgraphs that remain stable across different training environments [9, 42, 59, 52]. However, we argue that directly identifying invariant subgraphs can be challenging and error-prone, particularly when spurious edges exhibit strong correlations with target labels. In such scenarios, certain edges in the invariant subgraph G_c may be misclassified (i.e., assigned low predicted probabilities), leading to *partial* preservation of the invariant substructure and thereby degrading OOD generalization performance. In contrast, while a subset of spurious edges may correlate strongly with targets, the majority of spurious edges are relatively uninformative and easier to identify due to their weak correlations with labels. Consequently, pruning these less

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

informative edges is more likely to preserve the invariant substructure effectively. In this work, we raise the following research question:

Can we prune spurious edges instead of directly identifying invariant edges to enhance OOD generalization ability?

To address this question, we propose the first *pruning-based* OOD method. Unlike most existing graph OOD methods that aim to directly identify edges in the invariant subgraph, our method focuses on pruning spurious edges to achieve OOD generalization (Figure 1). We first begin with a case study to investigate the differences between our method and previous ones that directly identify invariant subgraphs, in terms of how the induced subgraph selector estimates edges from the invariant subgraph G_c and spurious subgraph G_s. Specifically, we observe that previous methods tend to misclassify some edges in G_c as unimportant edges with low probabilities, while assigning high probabilities to certain edges in G_s. As a result, *the invariant substructure in the graph is not preserved*. In contrast, our pruning-based method preserves the invariant subgraph more effectively (i.e., estimating the edges in G_c with high probabilities), although a small number of spurious edges may still remain due to the strong correlation with the targets. However, by preserving the invariant substructure more effectively, our method PrunE(***Prun***ing spurious ***E***dges for OOD generalization) achieves enhanced OOD performance compared to previous approaches that directly identify invariant subgraphs.

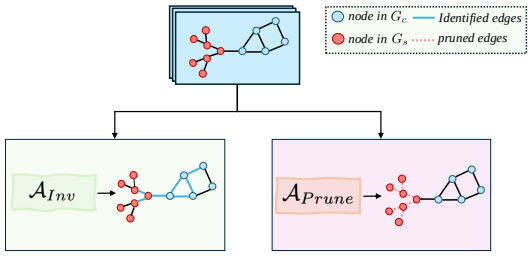


Figure 1: Illustration of two learning paradigms for graph-specific OOD methods. Previous methods seek to identify edges from the invariant subgraph directly, while our approach prunes edges from the spurious subgraph, which is more effective at preserving the invariant substructure.

The core insight behind PrunE is that Empirical Risk Minimization (ERM) [54] tends to capture all "useful" features that are correlated with the targets [26, 8]. In our context, ERM pushes the subgraph selector to preserve substructures that are more informative for prediction. By forcing uninformative edges to be excluded, G_c is preserved due to its strong correlation with the targets and the inherent inductive bias of ERM. To prune spurious edges, our proposed OOD objective consists of two terms that act on the subgraph selector, without adding additional OOD objective: 1) *graph size constraint.* This constraint limits the total edge weights derived from the subgraph selector to $\eta|G|$ for a graph G, where $\eta < 1$, thereby excluding some uninformative edges. 2) *ϵ-probability alignment.* This term aligns the probabilities of the lowest $K\%$ edges to be close to zero, further suppressing the occurrence of uninformative edges. Through theoretical analysis and extensive empirical validation, we demonstrate that PrunE significantly outperforms existing methods in OOD generalization, establishing state-of-the-art performance across various benchmarks. Our contributions are summarized as follows:

- **Novel framework.** We propose a *pruning-based* graph OOD method PrunE, which introduces a novel paradigm focusing on removing spurious edges rather than directly identifying edges in G_c. By pruning spurious edges, PrunEpreserves more edges in G_c than previous methods, thereby improving its OOD generalization performance.
- **Theoretical guarantee.** We provide theoretical analyses, demonstrating that: 1) The proposed graph size constraint provably enhances OOD generalization ability by reducing the size of G_s; 2) The proposed learning objective (Eqn. 5) provably identifies the invariant subgraph by pruning spurious edges.
- **Strong empirical performance.** We conduct experiments on both synthetic datasets and real-world datasets, compare against 15 baselines, PrunEoutperforms the second-best method by up to 24.19%, highlighting the superior OOD generalization ability.

2 Preliminaries

Notation. Throughout this work, an undirected graph G with n nodes and m edges is denoted by $G := \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the node set and $\mathcal{E}$ denotes the edge set. G is also represented by the

adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times D}$ with D feature dimensions. We use G_c and G_s to denote invariant subgraph and spurious subgraph. $\widehat{G}_c$ and $\widehat{G}_s$ denote the estimated invariant and spurious subgraph. $t : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times D} \to \mathbb{R}^{n\times n}$ refers to a learnable subgraph selector that models each independent edge probability, $\widetilde{G} \sim t(G)$ represents $\widetilde{G}$ is sampled from $t(G)$. We use $\mathbf{w}$ to denote a vector, and $\mathbf{W}$ to denote a matrix respectively. Finally, a random variable is denoted as W, a set is denoted using $\mathcal{W}$. A more complete set of notations is presented in Appendix A.

OOD Generalization. We consider the problem of graph classification under various forms of distribution shifts in hidden environments. Given a set of graph datasets $\mathcal{G} = \{G^e\}_{e\in\mathcal{E}_{\text{tr}}}$, a GNN model $f = \rho \circ h$, comprises an encoder h: $\mathbb{R}^{n\times n} \times \mathbb{R}^{n\times D} \to \mathbb{R}^F$ that learns a representation $\mathbf{h}_G$ for each graph G, followed by a downstream classifier ρ: $\mathbb{R}^F \to \mathbb{Y}$ to predict the label $\widehat{Y}_G = \rho(\mathbf{h}_G)$. In addition, a subgraph selector $t(\cdot)$ is employed to generate a graph with structural modifications. The objective of OOD generalization in our work is to learn an optimal composite function $f \circ t$ that encodes stable features by regularizing $t(\cdot)$ to prune spurious edges while preserving the edges in G_c.

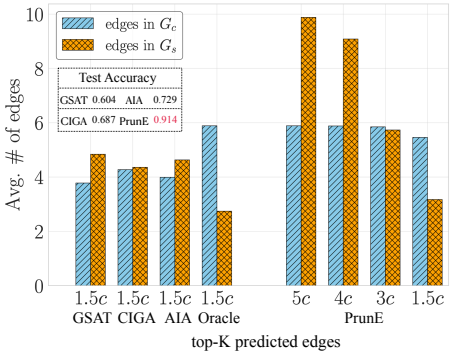


Figure 2: Illustration of the average number of edges from G_c and G_s included in the top-K predicted edges, where c denotes $|G_c|$.

Assumption 1. Given a graph $G \in \mathcal{G}$, there exists a stable subgraph G_c for every class label $Y \in \mathcal{Y}$, satisfying: a) $\forall e, e' \in \mathcal{E}_{tr}, P^e(Y \mid G_c) = P^{e'}(Y \mid G_c)$; b) The target Y can be expressed as $Y = f^*(G_c) + \epsilon$, where $\epsilon \perp\!\!\!\perp G$ represents random noise.

Assumption 1 posits the existence of a subgraph G_c that remains stable across different environments and causally determines the target Y, thus is strongly correlated with the target labels. Our goal in this work is to identify edges in G_c by excluding spurious edges to achieve OOD generalization.

3 Should We Identify Invariant Subgraphs or Prune Spurious Subgraphs?

Datasets. We use GOODMotif [15] dataset with *base* split for the case study. More details of this dataset can be found in Appendix J.

In this section, we conduct a case study to explore the differences between previous graph OOD methods and our proposed approach in the estimated edge probabilities. Through experiments, we observe that our pruning-based method is more effective at preserving G_c compared to previous methods that aim to directly identify G_c, thereby facilitating better OOD generalization performance for our approach. Next we detail the experimental setup and observations.

Experiment Setup. We use GSAT [42], CIGA [9], and AIA [52] as baseline methods representing three different lines of work for comparison, all of which utilize a subgraph selector to directly identify G_c for OOD generalization. After training and hyperparameter tuning, we obtain a model and a well-trained subgraph selector for each method. We evaluate the test performance on the Motif-base dataset and calculate the average number of edges in G_c and G_s among the top-K predicted edges for all methods. Here, we set $K = 1.5|G_c|$. For our method, we also present the statistics under different values of K.

Observations. From Figure 2, we observe that: (1) `PrunE` outperforms all the baselines by a significant margin, demonstrating superior OOD generalization ability; (2) When $K = 1.5|G_c|$, our method preserves more edges from G_c compared to other methods. Moreover, as K transitions from $5|G_c|$ to $1.5|G_c|$, the average number of edges in G_c remains nearly constant, while the number of edges from G_s decreases significantly. This indicates that most edges from G_c have predicted probabilities greater than those from G_s; (3) When compared with the oracle, the average number of edges in G_c under our method is still slightly lower than the oracle value, suggesting that a small number of spurious edges are estimated with high probability. In conclusion, compared to directly identifying invariant edges (i.e., edges in G_c), pruning spurious edges preserves more edges in G_c, even if some spurious edges remain challenging to eliminate. However, the OOD performance can be

substantially improved by retaining the invariant subgraphs, which explains why our pruning-based method outperforms previous approaches. We also provide a detailed discussion in Appendix I on why traditional graph OOD methods tend to assign low probabilities to edges in G_c, and how our pruning-based approach avoids this pitfall. Next, we detail the design of our pruning-based method.

4 Proposed Method

In this section, we present our pruning-based method PrunE, which directly regularizes the subgraph selector without requiring any additional OOD regularization. The pseudocode of PrunE is shown in Appendix E.

Subgraph selector. Following previous studies [66, 40, 59], we model each edge $e_{ij} \sim Bernoulli(p_{ij})$ independently which is parameterized by p_{ij}. The probability of the graph G is factorized over all the edges, i.e., $P(G) = \prod_{e_{ij} \in \mathcal{E}} p_{ij}$. Specifically, we employ a GNN model to derive the node representation for each node v, followed by an MLP to obtain the logits w_{ij} as following:

$$\begin{aligned} &\mathbf{h}_v = \mathrm{GNN}(v \mid G),\ v \in \mathcal{V}, \\ &w_{ij} = \mathrm{MLP}\left(\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_i \| \mathbf{h}_j\right), e_{ij} \in \mathcal{E}, \end{aligned} \tag{1}$$

here $\|$ denotes the concatenation operator. To ensure the sampling process from w_{ij} is differentiable and facilitate gradient-based optimization, we leverage the Gumbel-Softmax reparameterization trick [4, 41], which is applied as follows:

$$\begin{aligned} &p_{ij} = \sigma\left(\left(\log \epsilon - \log(1-\epsilon) + \omega_{ij}\right)/\tau\right), \epsilon \sim \mathcal{U}(0,1), \\ &\widetilde{\mathbf{A}}_{ij} = 1 - \mathrm{sg}(p_{ij}) + p_{ij}, \end{aligned} \tag{2}$$

here $\widetilde{\mathbf{A}}$ denotes the sampled adjacency matrix, τ is the temperature, $sg(\cdot)$ denotes the stop-gradient operator, and $\mathcal{U}(0,1)$ denotes the uniform distribution. $\mathbf{A}_{ij}$ is the edge weight for e_{ij}, which remains binary and differentiable for the gradient-based optimization.

Next, we introduce the proposed OOD objectives in PrunE that directly act on the subgraph selector to prune spurious edges: (1) *Graph size constraint*, which excludes a portion of uninformative spurious edges by limiting the total edge weights in the graph; (2) ϵ*-probability alignment*, which further suppresses the presence of uninformative edges by aligning the predicted probabilities of certain edges close to zero.

Graph size constraint. We first introduce a regularization term $\mathcal{L}_e$ which encourages a graph size distinction between $\widetilde{G} \sim t(G)$ and G:

$$\mathcal{L}_e = \mathbb{E}_{\mathcal{G}} \left(\frac{\sum_{(i,j) \in \mathcal{E}} \widetilde{\mathbf{A}}_{ij}}{|\mathcal{E}|} - \eta \right)^2, \tag{3}$$

where η is a hyper-parameter that controls the budget for the total number of edges pruned by $t(\cdot)$. The core insight is that when $\mathcal{L}_e$ acts as a regularization term for ERM, the subgraph selector will prune spurious edges while preserving edges in G_c, since ERM learns all useful patterns that are highly correlated with the target labels [26, 8]. Therefore, given Assumption 1, G_c will be preserved due to its strong correlation to the targets, and (a subset of) edges in G_s will be excluded. In practice, we find that $\eta \in \{0.5, 0.75, 0.85\}$ works well for most datasets. In Proposition 1, we demonstrate that the graph size regularization $\mathcal{L}_e$ provably prunes spurious edges while retaining invariant edges.

ϵ-probability alignment. Although $\mathcal{L}_e$ is able to prune a subset of spurious edges, it is challenging to get rid of all spurious edges. To further suppress the occurrence of spurious edges, we propose the following regularization on $t(\cdot)$:

$$\mathcal{L}_s = \mathbb{E}_{\mathcal{G}} \frac{1}{|\mathcal{E}_s|} \sum_{e_{ij} \in \mathcal{E}_s} |p_{ij} - \epsilon|. \tag{4}$$

Here,ϵ is a value close to zero, p_{ij} denotes the normalized probability of the edge e_{ij}, and $\mathcal{E}_s$ is the lowest $K\%$ of edges among all estimated edge weights $w_{ij} \in \mathcal{E}$ by the subgraph selector $t(\cdot)$.

The key insight is that edges from G_c are likely to exhibit higher predicted probabilities compared to edges in G_s. Thus, by aligning the bottom $K\%$ edges with the lowest predicted probability to a small

probability score ϵ, it becomes more likely to suppress spurious edges rather than invariant edges. When K gets larger, $\mathcal{L}_s$ will inevitably push down the probabilities of edges in G_c. However, ERM will drive up the probabilities of informative edges for accurate prediction, ensuring that the important edges are included in $\widehat{G}_c$. Therefore, the penalty for $\mathcal{L}_s$ should be relatively small compared to the penalty of ERM. In practice, we find that $\lambda_2 \in \{1e{-}2, 1e{-}3\}$ work stably across most datasets. In all experiments, we set $\epsilon = \frac{1}{|\mathcal{E}|}$, which works well for all the datasets.

Final objective. The overall objective is formulated as:

$$\mathcal{L} = \mathcal{L}_{GT} + \lambda_1 \mathcal{L}_e + \lambda_2 \mathcal{L}_s, \tag{5}$$

here $\lambda_i, i \in \{1, 2\}$ are hyperparameters that balance the contribution of each component to the overall objective, and $\mathcal{L}_{GT}$ denotes the ERM objective:

$$\mathcal{L}_{GT} = -\mathbb{E}_{\mathcal{G}} \sum_{k \in \mathcal{C}} Y_k \log \left(f(t(G))_k \right), \tag{6}$$

where Y_k denotes the class label k for graph G, $f(t(G))_k$ is the predicted probability for class k of graph G.

5 Theoretical Analysis

In this section, we provide some theoretical analysis on our proposed method PrunE. All proofs are included in Appendix F.

Proposition 1. *Under Assumption 1, the size constraint loss $\mathcal{L}_e$, when acting as a regularizer for the ERM loss $\mathcal{L}_{GT}$, will prune edges from the spurious subgraph G_s, while preserving the invariant subgraph G_c given a suitable η.*

Prop. 1 demonstrates that by enforcing graph size constraint, $\mathcal{L}_e$ will only prune spurious edges, thus making the size of G_s to be smaller. Next we show that $\mathcal{L}_e$ provably improves OOD generalization ability by shrinking $|G_s|$.

Theorem 5.1. *Let $l((x_i, x_j, y, G); \theta)$ denote the 0-1 loss function for predicting whether edge e_{ij} presents in graph G using $t(\cdot)$, and*

$$\begin{aligned} L(\theta; D) &:= \frac{1}{n} \sum_{(x_i, x_j, y, G) \sim D} l((x_i, x_j, y, G); \theta), \forall e_{ij} \in \mathcal{E}. \\ L(\theta; S) &:= \frac{1}{m} \sum_{(x_i, x_j, y, G) \sim S} l((x_i, x_j, y, G); \theta), \forall e_{ij} \in \mathcal{E}. \end{aligned} \tag{7}$$

where D and S represent the training and test set distributions, respectively,c is a constant, and n and m denotes the sample size in training set and test set respectively. Then, with probability at least $1 - \delta$ and $\forall \theta \in \Theta$, we have:

$$|L(\theta; D) - L(\theta; S)| \leq 2(c|G_s| + 1)M, \tag{8}$$

where $M = \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2n}} + \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2m}}$.

Theorem 5.1 establishes an OOD generalization bound that incorporates $|G_s|$ due to domain shifts. When $|G_s| = 0$, Eqn. 8 reduces to the traditional in-distribution generalization bound. Theorem 5.1 shows that $\mathcal{L}_e$ enhances the OOD generalization ability by reducing the size of G_s and tightens the generalization bound.

Theorem 5.2. *Let $\Theta^* = \arg\inf_\Theta \ \mathcal{L}(\Theta)$, where $\Theta^* = \{\rho^*(\cdot), h^*(\cdot), t^*(\cdot)\}$. For any graph G with target label $y \in \mathcal{Y}$, we have $G_c \approx \mathbb{E}_G[t^*(G)]$. Consequently, sampling from $t^*(G)$ in expectation will retain only the invariant subgraph G_c, which remains stable and sufficiently predictive for the target label y.*

Theorem 5.2 demonstrates the ability to retain only G_c by sampling from $t^*(G)$. While previous methods aim to directly identify G_c, PrunE is able to achieve the similar goal more effectively by pruning spurious edges.

Table 1: Performance on synthetic and real-world datasets. Numbers in **bold** indicate the best performance, while the underlined indicates the second best performance. $*$ denotes the test performance is statistically significantly better than the second-best method, with p-value less than 0.05.

Method	GOODMotif		GOODHIV		EC50			OGBG-Molbbbp	
	base	size	scaffold	size	scaffold	size	assay	scaffold	size
ERM	68.66±4.25	51.74±2.88	69.58±2.51	59.94±2.37	62.77±2.14	61.03±1.88	64.93±6.25	68.10±1.68	78.29±3.76
IRM	70.65±4.17	51.41±3.78	67.97±1.84	59.00±2.92	63.96±3.21	62.47±1.15	72.27±3.41	67.22±1.15	77.56±2.48
GroupDRO	68.24±8.92	51.95±5.86	70.64±2.57	58.98±2.16	64.13±1.81	59.06±1.50	70.52±3.38	66.47±2.39	79.27±2.43
VREx	71.47±6.69	52.67±5.54	70.77±2.84	58.53±2.88	64.23±1.76	63.54±1.03	68.23±3.19	68.74±1.03	78.76±2.37
DropEdge	45.08±4.46	45.63±4.61	70.78±1.38	58.53±1.26	63.91±2.56	61.93±1.41	73.79±4.06	66.49±1.55	78.32±3.44
G-Mixup	59.66±7.03	52.81±6.73	70.01±2.52	59.34±2.43	61.90±2.08	61.06±1.74	69.28±1.36	67.44±1.62	78.55±4.16
FLAG	61.12±5.39	51.66±4.14	68.45±2.30	60.59±2.95	64.98±0.87	64.28±0.54	74.91±1.18	67.69±2.36	79.26±2.26
LiSA	54.59±4.81	53.46±3.41	70.38±1.45	52.36±3.73	62.60±3.62	60.96±1.07	69.73±0.62	68.11±0.52	78.62±3.74
DIR	62.07±8.75	52.27±4.56	68.07±2.29	58.08±2.31	63.91±2.92	61.91±3.92	66.13±3.01	66.86±2.25	76.40±4.43
DisC	51.08±3.08	50.39±1.15	68.07±1.75	58.76±0.91	59.10±5.69	57.64±1.57	61.94±7.76	67.12±2.11	56.59±10.09
CAL	65.63±4.29	51.18±5.60	67.37±3.61	57.95±2.24	65.03±1.12	60.92±2.02	74.93±5.12	68.06±2.60	79.50±4.81
GREA	56.74±9.23	54.13±10.02	67.79±2.56	60.71±2.20	64.67±1.43	62.17±1.78	71.12±1.87	69.72±1.66	77.34±3.52
GSAT	60.42±9.32	53.20±8.35	68.66±1.35	58.06±1.98	65.12±1.07	61.90±2.12	74.77±4.31	66.78±1.45	75.63±3.83
CIGA	68.71±10.9	49.14±8.34	69.40±2.39	59.55±2.56	65.42±1.53	64.47±0.73	74.94±1.91	64.92±2.09	65.98±3.31
AIA	72.91±5.62	55.85±7.98	71.15±1.81	61.64±3.37	64.71±0.50	63.43±1.35	76.01±1.18	**70.79±1.53**	81.03±5.15
PrunE	**91.48*****±0.40**	**66.53*****±8.55**	**71.84*****±0.61**	**64.99*****±1.63**	**67.56*****±0.34**	**65.46*****±0.88**	**78.01*****±0.42**	70.32±1.73	**81.59±5.35**

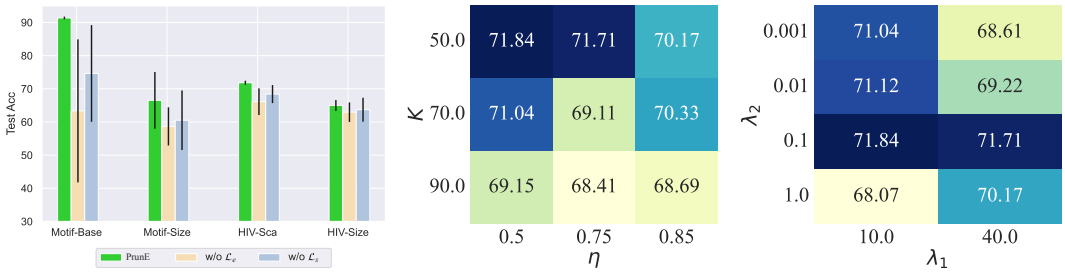
6 Related Work

OOD generalization on graphs. To tackle the OOD generalization challenge on graph, various methods have been proposed recently. MoleOOD [62], GIL [34] and MILI [57] aim to learn graph invariant features with environment inference. CIGA [9] adopts supervised contrastive learning to identify invariant subgraphs for OOD generalization. Several methods [59, 38, 52, 22, 37] utilize graph data augmentation to enlarge the training distribution without perturbing the stable patterns in the graph, enabling OOD generalization by identifying stable features across different augmented environments. SizeShiftReg [5] proposes a method for size generalization for graph-level classification using coarsening techniques. GSAT [42] uses the information bottleneck principle [53] to identify the minimum sufficient subgraph that explains the model's prediction. EQuAD [64] and LIRS [65] learn invariant features by disentangling spurious features in latent space. Many existing methods attempt to directly identify the invariant subgraph to learn invariant features. However, this approach can be error-prone, especially when spurious substructures exhibit strong correlations with the targets, leading to the failure to preserve the invariant substructure and ultimately limiting the OOD generalization capability. In contrast, `PrunE` aims to exclude spurious edges without directly identifying invariant edges, resulting in preserving the invariant substructure more effectively, and enhanced generalization performance.

Feature learning in the presence of spurious features. Several studies have explored the inductive bias and SGD training dynamics of neural networks in the presence of spurious features [45, 46, 49]. [49] shows that in certain scenarios neural networks can suffer from simplicity bias and rely on simple spurious features, while ignoring the core features. More recently, [26] and [8] found that even when neural networks heavily rely on spurious features, the core (causal) features can still be learned sufficiently well. Inspired by these studies, the subgraph selector should be able to include G_c to encode invariant features using ERM as the learning objective, given that G_c is both strongly correlated with and predictive of the targets (Assumption 1). This insight motivates us to propose a pruning-based graph OOD method. Compared to previous approaches, `PrunE` is capable of preserving a more intact set of edges from G_c to enhance OOD performance, at the cost that certain spurious edges may remain difficult to eliminate.

7 Experiments

In this section, we evaluate the effectiveness of `PrunE` on both synthetic datasets and real-world datasets, and answer the following research questions. **RQ1.** How does our method perform compared with SOTA baselines? **RQ2.** How do the individual components and hyperparameters in `PrunE` affect the overall performance? **RQ3.** Can the optimal subgraph selector $t^*(G)$ correctly identify G_c? **RQ4.** Do edges in G_c predicted by $t(\cdot)$ exhibit higher probability scores than edges in G_s? **RQ5.** How does `PrunE` perform on datasets with concept shift? **RQ6.** How do different GNN architectures impact



(a) Ablation study.

(b) Hyperparameter sensitivity analysis.

Figure 3: (a) Ablation on $\mathcal{L}_e$ and $\mathcal{L}_s$; (b) Hyperparameter sensitivity on GOODHIV-scaffold.

the OOD performance? More details on the datasets, experiment setup and experimental results are presented in Appendix J.

7.1 Experimental Setup

Datasets. We adopt GOOD datasets [15], OGBG-Molbbbp datasets [18, 60], and DrugOOD datasets [21] to comprehensively evaluate the OOD generalization performance of our proposed framework.

Baselines. Besides ERM [54], we compare our method against two lines of OOD baselines: (1) OOD algorithms on Euclidean data, including IRM [2], VREx [29], and GroupDRO [48]; (2) graph-specific methods which utilize a subgraph selector for OOD generalization, and data augmentation methods, including DIR [59], GSAT [42], GREA [38], DisC [11], CIGA [9], AIA [52], DropEdge [47], $\mathcal{G}$-Mixup [17], FLAG [28], and LiSA [67].

Evaluation. We report the ROC-AUC score for GOOD-HIV, OGBG-Molbbbp, and DrugOOD datasets, where the tasks are binary classification. For GOOD-Motif datasets, we use accuracy as the evaluation metric. We run experiments 4 times with different random seeds, and report the mean and standard deviations on the test set.

7.2 Experimental Results

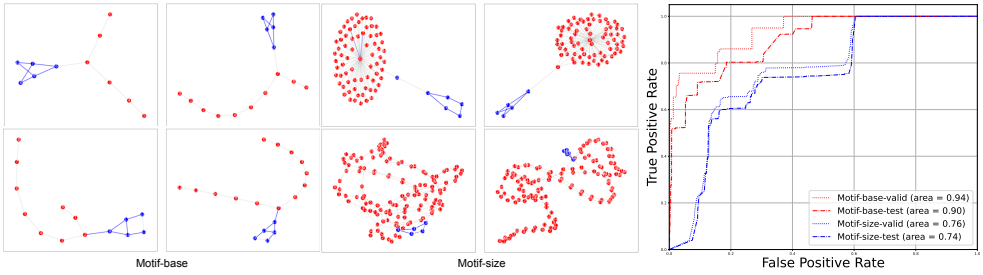
We report the main results on both synthetic and real-world datasets, as shown in Table 1.

Synthetic datasets. In synthetic datasets, `PrunE` outperforms graph OOD methods that attempt to directly identify G_c by a large margin, and surpasses the best baseline method AIA by 24.19% and 19.13% in Motif-base and Motif-size datasets respectively. This highlights the effectiveness of the pruning-based paradigm. We also observe that on the Motif-size dataset, the performance of most methods drop significantly, which can be attributed to the increased size of $|G_s|$ and the presence of more complex spurious substructures, leading to the overestimation of spurious edges by `PrunE`, as well as by other baselines that rely on invariant subgraph identification for OOD generalization.

Real-world datasets. In real-world datasets, many graph OOD algorithms exhibit instability, occasionally underperforming ERM. In contrast, our approach consistently achieves stable and superior performance across a diverse set of distribution shifts, and outperforms the best baseline method AIA which seeks to identify invariant subgraph directly by an average of 2.38% in 7 real-world datasets. This also demonstrates that the proposed pruning-based paradigm can be effectively applied to various real-world scenarios, highlighting its applicability.

7.3 Ablation Study

In this section, we evaluate the impact of $\mathcal{L}_e$ and $\mathcal{L}_s$ using the GOODMotif and GOODHIV datasets. As illustrated in Figure 3(a), removing either $\mathcal{L}_e$ or $\mathcal{L}_s$ leads to a significant drop in test performance across all datasets, and a larger variance. The removal of $\mathcal{L}_e$ results in a more significant decline, as this regularization penalty is stronger (e.g., λ_1 is set to 10 or 40 in the experiments). However, even with $\mathcal{L}_e$, some spurious edges may still exhibit high probabilities, potentially inducing a large variance.



(a) Visualizations on learned subgraph by $t^*(\cdot)$, where blue nodes are ground-truth nodes in G_c, and red nodes are ground-truth nodes in G_s. The highlighted blue edges are top-K edges predicted by $t^*(\cdot)$, where K is the number of ground-truth invariant edges.

(b) The ROC-AUC curve for predicted edges and ground-truth edges on GOODMotif-base and GOODMotif-size datasets.

Figure 4: Empirical visualization and analysis on $t^*(\cdot)$.

By further employing $\mathcal{L}_s$, PrunE effectively reduce predicted probabilities for most spurious edges, thus further reduce the variance and improve the performance.

7.4 Hyper-parameter Sensitivity

In this section, We study the impact of hyperparameter sensitivity on the edge budget η in $\mathcal{L}_e$, the bottom $K\%$ edges with the lowest probability, and the alignment probability ϵ in $\mathcal{L}_s$. Additionally, we investigate the effects of varying the penalty weights λ_1 and λ_2 for $\mathcal{L}_e$ and $\mathcal{L}_s$. As illustrated in Figure 3(b), the test performance for GOODHIV scaffold remains stable across different hyperparameter settings. Additional results on more datasets are included in Appendix J, which also exhibit stable performance across a wide range of hyperparameter settings, further highlighting the stability of PrunE. Furthermore, we investigate the impact of ϵ in $\mathcal{L}_s$. As shown in Table 2, the optimal performance is observed when ϵ is a small value close to zero. As ϵ increases, the test performance declines, especially on synthetic datasets. This decline occurs because larger values of ϵ weaken the suppression effect, potentially leading to adverse effect that hinder generalization. Notably, when $\epsilon = \frac{1}{|\mathcal{E}|}$, the suppression strength is dynamically adjusted for each graph instance, resulting in stable performance across diverse datasets. In summary, although setting certain hyperparameters outside appropriate ranges may lead to failures in OOD generalization—for example, a small η (e.g., 0.1) may prune edges in G_c, and a small λ_1 may have insufficient effect on suppressing spurious edges—PrunE exhibits stable performance across a variety of datasets when hyperparameters are chosen within reasonable ranges (hyperparameters in Figure 3(b)).

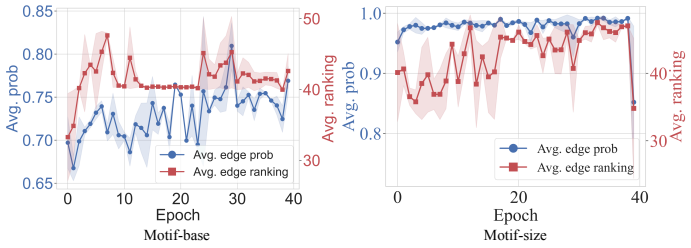
Table 2: Test performance with varying ϵ.

	Motif-base	Motif-size	EC50-assay
$\epsilon = 0.01$	91.63±0.73	60.38±8.35	77.76±1.11
$\epsilon = 0.1$	88.14±0.67	62.38±10.76	76.65±1.92
$\epsilon = 0.3$	80.93±4.33	50.65±4.95	76.07±2.65
$\epsilon = 0.5$	74.52±19.89	50.28±8.35	75.93±1.27
$\epsilon = \frac{1}{\lvert\mathcal{E}\rvert}$	91.48±0.40	66.53±8.55	78.01±0.42

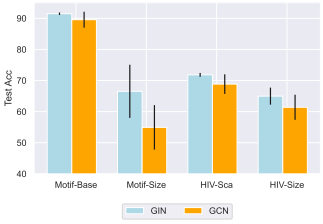
7.5 In-depth Analysis

Can $t^*(\cdot)$ identify G_c? To verify whether $t^*(\cdot)$ can indeed identify G_c, we conduct experiments using GOOD-Motif datasets with both *Motif-base* and *Motif-size* splits. These synthetic datasets are suitable for this analysis as they provide ground-truth labels for edges and nodes that are causally related to the targets. First, we collect the target label for each edge, and the predicted probability score from $t^*(\cdot)$ for correctly predicted samples and plot the ROC-AUC curve for both the validation and test sets for the two datasets. As illustrated in Figure 4(b), the AUC scores for both datasets exhibit high values, demonstrating that $t^*(\cdot)$ accurately identifies G_c, which is consistent with the theoretical insights provided in Theorem 5.2. Figure 4(a) illustrates some visualization results using $t^*(\cdot)$, demonstrating that $t^*(\cdot)$ correctly identify invariant edges from G_c. More visualization results for the identified edges using $t^*(\cdot)$ are provided in Appendix J.

Do edges in G_c exhibit a higher probability than edges in G_s? We assess the probability scores and ranking of edges in G_c compared to those in G_s using the GOOD-Motif datasets. Specifically,



(a) Average predicted probability and ranking of edges in G_c.



(b) Performance with different GNN encoders.

Figure 5: (a) Average probability and ranking of edges in G_c for every training epoch. Invariant edges are generally assigned higher scores, though spurious edges may be overestimated due to label correlation. (b) Test performance with different GNN encoders. PrunE benefits from expressive architectures under OOD settings.

we plot the average probability and ranking of edges in G_c over the first 40 epochs (excluding the first 10 epochs for ERM pretraining), using the ground-truth edge labels. As shown in Figure 5(a), for both the Motif-base and Motif-size datasets, the invariant edges in G_c exhibit high probability scores, ranking among the top 50% in both datasets. This demonstrates that the edges from the invariant subgraph generally get higher predicted probability scores compared to spurious edges. However, certain spurious edges may still be overestimated due to their strong correlation with the target labels.

How does PrunE perform on datasets with concept shift? In the main results, we use covariate shift to evaluate the OOD performance of various methods, where unseen environments arise in validation and test datasets. We also adopt concept shift to evaluate the effectiveness of PrunE, where spurious correlation strength varies in training and test sets. As shown in Table 3, PrunE also outperforms the SOTA methods significantly. For Motif-base dataset, most of the methods underperform ERM, while PrunE achieves 90.28% test accuracy, which is 10.86% higher than ERM.

How do different GNN encoders affect the model performance? We examine the effect of using different GNN encoders, specifically GCN [25] and GIN [61], with the same hidden dimensions and number of layers as $h(\cdot)$. As illustrated in Figure 5(b), across all four datasets, employing GIN as the feature encoder leads to a increase in test performance. This is likely due to GIN's higher expressivity than GCN [61], being as powerful as the 1-WL test [31], which allows it to generate more distinguishable features compared to GCN. These enhanced features benefits the optimization of $t(\cdot)$, thereby improving the identification of G_c for OOD generalization. This also highlights another advantage of PrunE: utilizing a GNN encoder with enhanced expressivity may further facilitate OOD generalization by more accurately identifying G_c through $t(\cdot)$.

Table 3: Model performance with concept shift.

Method	GOODHIV	GOODMotif
	size	base
ERM	63.26±2.47	81.44±0.45
IRM	59.90±3.15	80.71±0.46
VRex	60.23±1.70	81.56±0.35
GSAT	56.76±7.16	76.07±3.48
GREA	60.07±5.40	78.27±4.29
CIGA	73.62±0.86	81.68±3.01
AIA	74.21±1.81	82.51±2.81
PrunE	**79.50±1.57**	**90.28±1.72**

8 Conclusion

Many graph-specific OOD methods aim to directly identify edges in the invariant subgraph to achieve OOD generalization, which can be challenging and prone to errors. In response, we propose PrunE, a pruning-based OOD method that focuses on removing spurious edges by imposing regularization terms on the subgraph selector, without introducing any additional OOD objectives. Through a case study, we demonstrate that, compared to conventional methods, PrunE exhibits enhanced OOD generalization capability by retaining more edges in the invariant subgraph. Theoretical analysis and extensive experiments across various datasets validate the effectiveness of this novel learning paradigm. Future research directions include: (1) Extending the pruning-based paradigm to a self-supervised setting without relying on the power of ERM; (2) Expanding this learning paradigm to other scenarios, such as dynamic graphs under distribution shifts.

References

[1] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34:3438–3450, 2021.

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

[3] Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996.

[4] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[5] Davide Buffelli, Pietro Liò, and Fabio Vandin. Sizeshiftreg: a regularization method for improving size-generalization in graph neural networks. *Advances in Neural Information Processing Systems*, 35:31871–31885, 2022.

[6] Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. A unified lottery ticket hypothesis for graph neural networks. In *International conference on machine learning*, pages 1695–1706. PMLR, 2021.

[7] Yongqiang Chen, Yatao Bian, Kaiwen Zhou, Binghui Xie, Bo Han, and James Cheng. Does invariant graph learning via environment augmentation learn invariance? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[8] Yongqiang Chen, Wei Huang, Kaiwen Zhou, Yatao Bian, Bo Han, and James Cheng. Understanding and improving feature learning for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 36:68221–68275, 2023.

[9] Yongqiang Chen, Yonggang Zhang, Yatao Bian, Han Yang, MA Kaili, Binghui Xie, Tongliang Liu, Bo Han, and James Cheng. Learning causally invariant representations for out-of-distribution generalization on graphs. *Advances in Neural Information Processing Systems*, 35:22131–22148, 2022.

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

[11] Shaohua Fan, Xiao Wang, Yanhu Mo, Chuan Shi, and Jian Tang. Debiasing graph neural networks via learning disentangled causal substructure. *Advances in Neural Information Processing Systems*, 35:24934–24946, 2022.

[12] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric, 2019.

[13] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[14] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272. PMLR, 2017.

[15] Shurui Gui, Xiner Li, Limei Wang, and Shuiwang Ji. GOOD: A graph out-of-distribution benchmark. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[16] Shurui Gui, Meng Liu, Xiner Li, Youzhi Luo, and Shuiwang Ji. Joint learning of label and environment causal independence for graph out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 36, 2023.

[17] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, and Xia Hu. G-mixup: Graph data augmentation for graph classification. In *International Conference on Machine Learning*, pages 8230–8248. PMLR, 2022.

[18] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in Neural Information Processing Systems*, 33:22118–22133, 2020.

[19] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548*, 2021.

[20] Bo Hui, Da Yan, Xiaolong Ma, and Wei-Shinn Ku. Rethinking graph lottery tickets: Graph sparsity matters. *arXiv preprint arXiv:2305.02190*, 2023.

[21] Yuanfeng Ji, Lu Zhang, Jiaxiang Wu, Bingzhe Wu, Long-Kai Huang, Tingyang Xu, Yu Rong, Lanqing Li, Jie Ren, Ding Xue, et al. Drugood: Out-of-distribution (ood) dataset curator and benchmark for ai-aided drug discovery–a focus on affinity prediction problems with noise annotations. *arXiv preprint arXiv:2201.09637*, 2022.

[22] Tianrui Jia, Haoyang Li, Cheng Yang, Tao Tao, and Chuan Shi. Graph invariant learning with subgraph co-mixup for out-of-distribution generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 8562–8570, 2024.

[23] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Domain extrapolation via regret minimization. *arXiv preprint arXiv:2006.03908*, 3, 2020.

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

[25] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[26] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. In *The Eleventh International Conference on Learning Representations*, 2023.

[27] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[28] Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. Robust optimization as data augmentation for large-scale graphs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 60–69, 2022.

[29] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[30] Steinar Laenen. One-shot neural network pruning via spectral graph sparsification. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pages 60–71. PMLR, 2023.

[31] Andrei Leman and Boris Weisfeiler. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

[32] Gaotang Li, Marlena Duda, Xiang Zhang, Danai Koutra, and Yujun Yan. Interpretable sparsification of brain graphs: Better practices and effective designs for graph neural networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1223–1234, 2023.

[33] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. Ood-gnn: Out-of-distribution generalized graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[34] Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu. Learning invariant graph representations for out-of-distribution generalization. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[35] Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu. Invariant node representation learning under distribution shifts with multiple latent environments. *ACM Transactions on Information Systems*, 42(1):1–30, 2023.

[36] Xiner Li, Shurui Gui, Youzhi Luo, and Shuiwang Ji. Graph structure and feature extrapolation for out-of-distribution generalization. *arXiv preprint arXiv:2306.08076*, 2023.

[37] Xiner Li, Shurui Gui, Youzhi Luo, and Shuiwang Ji. Graph structure extrapolation for out-of-distribution generalization. In *Forty-first International Conference on Machine Learning*, 2024.

[38] Gang Liu, Tong Zhao, Jiaxin Xu, Tengfei Luo, and Meng Jiang. Graph rationalization with environment-based augmentations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22. ACM, August 2022.

[39] Yang Liu, Xiang Ao, Fuli Feng, Yunshan Ma, Kuan Li, Tat-Seng Chua, and Qing He. Flood: A flexible invariant learning framework for out-of-distribution generalization on graphs. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1548–1558, 2023.

[40] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in Neural Information Processing Systems*, 33:19620–19631, 2020.

[41] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[42] Siqi Miao, Mia Liu, and Pan Li. Interpretable and generalizable graph learning via stochastic attention mechanism. In *International Conference on Machine Learning*, pages 15524–15543. PMLR, 2022.

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[44] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 78(5):947–1012, 2016.

[45] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

[46] N Rahaman, A Baratin, D Arpit, F Draxler, M Lin, F Hamprecht, Y Bengio, and A Courville. On the spectral bias of neural networks: International conference on machine learning. *arXiv*, 2019.

[47] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

[48] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[49] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

[50] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[51] Yong-Duo Sui, Xiang Wang, Tianlong Chen, Meng Wang, Xiang-Nan He, and Tat-Seng Chua. Inductive lottery ticket learning for graph neural networks. *Journal of Computer Science and Technology*, 39(6):1223–1237, 2024.

[52] Yongduo Sui, Qitian Wu, Jiancan Wu, Qing Cui, Longfei Li, Jun Zhou, Xiang Wang, and Xiangnan He. Unleashing the power of graph data augmentation on covariate distribution shift. *Advances in Neural Information Processing Systems*, 36, 2023.

[53] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle, 2015.

[54] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.

[55] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[56] Kun Wang, Yuxuan Liang, Pengkun Wang, Xu Wang, Pengfei Gu, Junfeng Fang, and Yang Wang. Searching lottery tickets in graph neural networks: A dual perspective. In *The Eleventh International Conference on Learning Representations*, 2022.

[57] Ruijia Wang, Haoran Dai, Cheng Yang, Le Song, and Chuan Shi. Advancing molecule invariant representation via privileged substructure identification. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3188–3199, 2024.

[58] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. Handling distribution shifts on graphs: An invariance perspective. *arXiv preprint arXiv:2202.02466*, 2022.

[59] Yingxin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat-Seng Chua. Discovering invariant rationales for graph neural networks. In *International Conference on Learning Representations*, 2022.

[60] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[61] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[62] Nianzu Yang, Kaipeng Zeng, Qitian Wu, Xiaosong Jia, and Junchi Yan. Learning substructure invariance for out-of-distribution molecular representations. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[63] Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pages 25407–25437. PMLR, 2022.

[64] Tianjun Yao, Yongqiang Chen, Zhenhao Chen, Kai Hu, Zhiqiang Shen, and Kun Zhang. Empowering graph invariance learning with deep spurious infomax. In *Forty-first International Conference on Machine Learning*, 2024.

[65] Tianjun Yao, Yongqiang Chen, Kai Hu, Tongliang Liu, Kun Zhang, and Zhiqiang Shen. Learning graph invariance by harnessing spuriosity. In *The Thirteenth International Conference on Learning Representations*, 2025.

[66] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[67] Junchi Yu, Jian Liang, and Ran He. Mind the label shift of augmentation-based graph ood generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11620–11630, 2023.

[68] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):5782–5799, 2022.

[69] Guibin Zhang, Kun Wang, Wei Huang, Yanwei Yue, Yang Wang, Roger Zimmermann, Aojun Zhou, Dawei Cheng, Jin Zeng, and Yuxuan Liang. Graph lottery ticket automated. In *The Twelfth International Conference on Learning Representations*, 2024.

[70] Xiang Zhuang, Qiang Zhang, Keyan Ding, Yatao Bian, Xiao Wang, Jingsong Lv, Hongyang Chen, and Huajun Chen. Learning invariant molecular representation in latent discrete space. *Advances in Neural Information Processing Systems*, 36, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main contributions are highlighted in the abstract and introduction section.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations of our work in the Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We have provided detailed proof in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided detailed experimental setup in the Appendix, and disclose our project code anonymously.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use public datasets, and disclose our code anonymously.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, details about experimental setting are provided in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run experiments four times, and report error bars for the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have detailed our hardware and software used in the experiments in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research won't bring any potential harmful societal impact.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have included a section in appendix regarding broader impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The safeguards is not related to our study.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have cited the relevant work for data and code.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: There are no new assets in our work.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This topic is not related with our work.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This topic is related with our work.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The declaration is not required for our study.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix of PrunE

Contents

A Notations

We present a set of notations used throughout our paper for clarity. Below are the main notations along with their definitions.

Table 4: Notation Table

Symbols	Definitions
$\mathcal{G}$	Set of graph datasets
$\mathcal{E}_{\text{tr}}$	Set of environments used for training
$\mathcal{E}_{\text{all}}$	Set of all possible environments
G	An undirected graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$
$\mathcal{V}$	Node set of graph G
$\mathcal{E}$	Edge set of graph G
$\mathbf{A}$	Adjacency matrix of graph G
$\mathbf{X}$	Node feature matrix of graph G
D	Feature dimension of node features in $\boldsymbol{X}$
G_c	Invariant subgraph of G
G_s	Spurious subgraph of G
$\widehat{G}_c$	Estimated invariant subgraph
$\widehat{G}_s$	Estimated spurious subgraph
$\lvert G\rvert$	The number of edges in graph G.
Y	Target label variable
$\mathbf{w}$	A vector
$\mathbf{W}$	A matrix
W	A random variable
$\mathcal{W}$	A set
$f = \rho \circ h$	A GNN model comprising encoder $h(\cdot)$ and classifier $\rho(\cdot)$
$t(\cdot)$	Learnable data transformation function for structural modifications
$\widetilde{G} \sim t(\cdot)$	A view sampled from $t(\cdot)$, e.g., $\widetilde{G} \sim t(\cdot)$. We may use $t(G)$ to denote a sampled view from G via $t(\cdot)$, e.g., $I(G; t(G))$
$\mathbf{h}_v$	Representation of node $v \in \mathcal{V}$ of graph G

B Broad Impact

This work proposes a novel paradigm for OOD generalization that departs from conventional invariant learning approaches, which typically attempt to learn invariant features directly through various optimization objectives [2, 1, 23, 29, 10]. Instead, we advocate for a complementary perspective: enhancing OOD generalization by explicitly pruning spurious features. In the graph domain, we realize this paradigm by pruning spurious edges, which allows the model to retain invariant substructures. This shift in perspective offers a new direction with substantial potential, especially in domains where identifying invariant substructures offers valuable insights.

The proposed method not only achieves strong empirical performance under distribution shifts but also offers better explainability by providing explicit insights into which parts of the graph structure effectively explains the model prediction. This contributes toward more transparent and trustworthy machine learning models. Furthermore, our approach is generally applicable to graph-structured data across a wide range of applications, including molecular property prediction, social network analysis, and knowledge reasoning.

C More Preliminaries

Graph Neural Networks. In this work, we adopt message-passing GNNs for graph classification due to their expressiveness. Given a simple and undirected graph $G = (\mathbf{A}, \mathbf{X})$ with n nodes and m edges, where $\mathbf{A} \in \{0,1\}^{n \times n}$ is the adjacency matrix, and $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the node feature

matrix with d feature dimensions, the graph encoder $h : \mathbb{G} \to \mathbb{R}^h$ aims to learn a meaningful graph-level representation h_G, and the classifier $\rho : \mathbb{R}^h \to \mathbb{Y}$ is used to predict the graph label $\widehat{Y}_G = \rho(h_G)$. To obtain the graph representation h_G, the representation $\mathbf{h}_v^{(l)}$ of each node v in a graph G is iteratively updated by aggregating information from its neighbors $\mathcal{N}(v)$. For the l-th layer, the updated representation is obtained via an AGGREGATE operation followed by an UPDATE operation:

$$\mathbf{m}_v^{(l)} = \text{AGGREGATE}^{(l)}\left(\left\{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\right\}\right), \tag{9}$$

$$\mathbf{h}_v^{(l)} = \text{UPDATE}^{(l)}\left(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l)}\right), \tag{10}$$

where $\mathbf{h}_v^{(0)} = \mathbf{x}_v$ is the initial node feature of node v in graph G. Then GNNs employ a READOUT function to aggregate the final layer node features $\left\{\mathbf{h}_v^{(L)} : v \in \mathcal{V}\right\}$ into a graph-level representation $\mathbf{h}_G$:

$$\mathbf{h}_G = \text{READOUT}\left(\left\{\mathbf{h}_v^{(L)} : v \in \mathcal{V}\right\}\right). \tag{11}$$

D Additional Related Work

OOD Generalization on Graphs. Recently, there has been a growing interest in learning graph-level representations that are robust under distribution shifts, particularly from the perspective of invariant learning. MoleOOD [62] and GIL [34] propose to infer environmental labels to assist in identifying invariant substructures within graphs. DIR [59], GREA [38] and iMoLD [70] employ environment augmentation techniques to facilitate the learning of invariant graph-level representations. These methods typically rely on the explicit manipulation of unobserved environmental variables to achieve generalization across unseen distributions. AIA [52] employs an adversarial augmenter to explore OOD data by generating new environments while maintaining stable feature consistency. To circumvent the need for environmental inference or augmentation, CIGA [9] and GALA [7] utilizes supervised contrastive learning to identify invariant subgraphs based on the assumption that samples sharing the same label exhibit similar invariant subgraphs. LECI [16] and G-Splice [36] assume the availability of environment labels, and study environment exploitation strategies for graph OOD generalization. LECI [16] proposes to learn a causal subgraph selector by jointly optimizing label and environment causal independence, and G-Splice [36] studies graph and feature space extrapolation for environment augmentation, which maintains causal validity. EQuAD [64] and LIRS [65] learn invariant features by disentangling spurious features via two stage learning paradigm, i.e., learning spurious features via self-supervised learning followed by disentangling spurious features via ERM. On the other hand, some works do not utilize the invariance principle for graph OOD generalization. DisC [11] initially learns a biased graph representation and subsequently focuses on unbiased graphs to discover invariant subgraphs. GSAT [42] utilizes information bottleneck principle [53] to learn a minimal sufficient subgraph for GNN explainability, which is shown to be generalizable under distribution shifts. OOD-GNN [33] proposes to learn disentangled graph representation by computing global weights of all data.

Node-level OOD Generalization. There has been substantial work on OOD generalization for node-level classification tasks. Most existing methods [58, 39, 35, 67] adopt invariant learning to address node-level OOD challenges. Compared to graph-level OOD generalization, node-level OOD problems face unique difficulties, including: (1) distinct types of distribution shifts (e.g., structural or feature-level shifts), (2) non-i.i.d. node dependencies due to the interconnected nature, and (3) computational bottlenecks from subgraph extraction when reducing to graph-level OOD tasks. Due to these challenges, our pruning-based approach cannot be directly extended to node-level tasks. We leave this adaptation to future work.

Lottery Ticket Hypothesis and Graph Sparsification. The Lottery Ticket Hypothesis [13] suggests that large neural networks contain small subnetworks (i.e., winning tickets) that, when trained in isolation, can match the performance of the original model. This idea has been extended to GNNs through the Graph Lottery Ticket (GLT) Hypothesis, which posits that sparse subgraphs can preserve GNN performance. [6] introduced a unified framework for pruning both GNN parameters and edges, with many follow-up studies [20, 30, 32, 56, 51, 69] that improve pruning strategies, robustness, and even transferability across graphs.

While both graph sparsification and PrunE perform edge pruning, they differ in motivation and optimization. GLT and Graph sparsification targets large-scale graphs where computational bottlenecks are critical, seeking maximal compression while preserving performance through lottery ticket principles. In contrast, PrunE addresses graph-level OOD generalization on smaller graphs where efficiency is not the primary concern. Technically, graph sparsification optimizes masks under ERM with sparsity regularization and employs rewinding operations. PrunE follows the typical OOD optimization framework where ERM loss is regularized to encourage invariance. Thus, while both involve edge pruning, graph sparsification seeks compact subnetworks for efficiency backed by lottery ticket hypothesis, whereas PrunE leverages pruning as a regularization strategy to suppress spurious edges and improve OOD generalization.

E Algorithmic Pseudocode of PrunE

In this section, we provide the pseudocode of PrunE in Algorithm 1.

Algorithm 1 The proposed method

Input: Graph dataset $\mathcal{G}$, epochs E, learning rates η, hyperparameters λ_1, λ_2
Output: Optimized GNN model $f^* = \rho^* \circ h^*$, and the subgraph selector $t^*(\cdot)$.
Initialize: GNN encoder $h(\cdot)$, classifier $\rho(\cdot)$, and the learnable data transformation $t(\cdot)$.
for epoch $e = 1$ **to** E **do**
    for each minibatch $\mathcal{B} \in \mathcal{G}$ **do**
        Calculate w_{ij} using Eqn. 1 for each graph $G \in \mathcal{B}$
        Calculate $\mathcal{L}_e$ using Eqn. 3
        Calculate $\mathcal{L}_s$ using Eqn. 4
        Sample $\widetilde{G} \sim t(G)$ using $t(\cdot)$ for each $G \in \mathcal{B}$
        Calculate cross-entropy loss $\mathcal{L}_{GT}$ using Eqn. 6
        Compute the total loss $\mathcal{L} = \mathcal{L}_{GT} + \lambda_1 \mathcal{L}_e + \lambda_2 \mathcal{L}_s$
        Perform backpropagation to update the parameters of $h(\cdot)$, $\rho(\cdot)$, and $t(\cdot)$
    end for
end for

F Proofs of Theoretical Results

F.1 Proof of Proposition 1

Proof. We begin by expanding the cross-entropy loss $\mathcal{L}_{GT}$ as:

$$\mathcal{L}_{GT} = -\mathbb{E}_{\mathcal{G}} \left[\log \mathbb{P}(Y \mid f(\widetilde{G})) \right], \tag{12}$$

where $\widetilde{G} \sim t(G)$. Supposing that $|\widetilde{G}| > |G_c|$, which can be controlled by the hyperparameter η in Eqn. 3, further assume that $\widetilde{G}$ does not include the invariant subgraph G_c. Let a subgraph g be subtracted from $\widetilde{G}$ and $|g| = |G_c|$, we then define a new subgraph $G' = \widetilde{G} \setminus g$, and we add G_c to G' to form the new graph $G' \cup G_c$.

Under Assumption 1, we know that the invariant subgraph G_c holds sufficient predictive power to Y, and G_c is more informative to Y than G_s, therefore including G_c will always make the prediction more certain, i.e.,

$$\mathbb{P}(Y \mid f(G' \cup G_c)) > \mathbb{P}(Y \mid f(G' \cup g)), \forall g \subseteq \widetilde{G}, \tag{13}$$

As a result, $\mathcal{L}_{GT}$ will become smaller. Therefore, we conclude that under the graph size regularization imposed by $\mathcal{L}_e$, the optimal solution $\widetilde{G} \sim t(G)$ will always include the invariant subgraph G_c, while pruning edges from the spurious subgraph G_s. This completes the proof. □

Remark. When η is set too small, the loss term $\mathcal{L}_e$ may inadvertently prune edges in G_c, thereby corrupting the invariant substructure and degrading OOD generalization performance. In practice, we observe that $\eta = \{0.5, 0.75, 0.85\}$ works well across most datasets stably.

F.2 Proof of Theorem 5.1

Proof. We first formally define the notations in our proof. Let $l((x_i, x_j, y, G); \theta)$ denote the 0-1 loss for the edge e_{ij} being presented in graph G, and

$$\begin{aligned} L(\theta; D) &:= \frac{1}{n} \sum_{(x_i, x_j, y, G) \sim D} l\left((x_i, x_j, y, G); \theta\right), \\ L(\theta; S) &:= \frac{1}{m} \sum_{(x_i, x_j, y, G) \sim S} l\left((x_i, x_j, y, G); \theta\right), \end{aligned} \tag{14}$$

where D and S represent the training and test distributions, with n and m being their respective sample sizes. We define:

$$\begin{aligned} L_c(\theta; D) &= \frac{1}{n} \sum_{(x_i, x_j, y, G) \sim D} l((x_i, x_j, y, G_c); \theta), \forall e_{ij} \in G_c. \\ L_s(\theta; D) &= \frac{1}{n} \sum_{(x_i, x_j, y, G) \sim D} l\left((x_i, x_j, y, G_s); \theta\right), \forall e_{ij} \in G_s. \end{aligned} \tag{15}$$

Similarly, $L_c(\theta; S)$ and $L_s(\theta; S)$ can be defined for the test distribution. Under Assumption 1, $L_c(\theta; D)$ and $L_c(\theta; S)$ are identically distributed due to the stability of G_c across environments, while $L_s(\theta; D)$ and $L_s(\theta; S)$ differ because of domain shifts in G_s. We assume:

$$L_s(\theta; \cdot) := c\,|G_s|\, L_c(\theta; \cdot), \tag{16}$$

where c is a proportionality constant. As $L_s(\cdot)$ is defined to a summation over all spurious edges, we put $|G_s|$ in the r.h.s to account for this factor. When $|G_s| = 0$, the loss reduces to the in-distribution case $L_c(\theta; \cdot)$.

$$\begin{aligned} |L(\theta; D) - L(\theta; S)| &= |L_c(\theta; D) + L_s(\theta; D) - L_c(\theta; S) - L_s(\theta; S)| && (17) \\ &\leq |L_c(\theta; D) - L_c(\theta; S)| + |L_s(\theta; D) - L_s(\theta; S)| && (18) \\ &= |L_c(\theta; D) - L_c(\theta; S)| + c\,|G_s|\,|L_c(\theta; D) - L_c(\theta; S)| && (19) \\ &= (c\,|G_s| + 1)\,|L_c(\theta; D) - L_c(\theta; S)|\,. && (20) \end{aligned}$$

To bound $|L_c(\theta; D) - L_c(\theta; S)|$, we decompose it as:

$$|L_c(\theta; D) - L_c(\theta; S)| \leq |L_c(\theta; D) - \mathbb{E}\left[L_c(\theta; D)\right]| + |\mathbb{E}\left[L_c(\theta; S)\right] - L_c(\theta; S)|\,. \tag{21}$$

Applying Hoeffding's Inequality to each term:

$$\begin{aligned} \mathbb{P}\left(|\mathbb{E}\left[L_c(\theta; D)\right] - L_c(\theta; D)| \geq \epsilon\right) &\leq 2\exp\left(-2\epsilon^2 n\right), && (22) \\ \mathbb{P}\left(|\mathbb{E}\left[L_c(\theta; S)\right] - L_c(\theta; S)| \geq \epsilon\right) &\leq 2\exp\left(-2\epsilon^2 m\right). && (23) \end{aligned}$$

Union bounding over all $\theta \in \Theta$:

$$\begin{aligned} \mathbb{P}\left(\exists\theta \in \Theta : |\mathbb{E}\left[L_c(\theta; D)\right] - L_c(\theta; D)| \geq \epsilon\right) &\leq 2|\Theta|\exp\left(-2\epsilon^2 n\right), && (24) \\ \mathbb{P}\left(\exists\theta \in \Theta : |\mathbb{E}\left[L_c(\theta; S)\right] - L_c(\theta; S)| \geq \epsilon\right) &\leq 2|\Theta|\exp\left(-2\epsilon^2 m\right). && (25) \end{aligned}$$

Setting both probabilities to $\delta/2$ and solving for ϵ:

$$\epsilon_D = \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2n}}, \tag{26}$$

$$\epsilon_S = \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2m}}. \tag{27}$$

Thus, with probability at least $1 - \delta$:

$$|L_c(\theta; D) - L_c(\theta; S)| \leq \epsilon_D + \epsilon_S \tag{28}$$

$$= \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2n}} + \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2m}}. \tag{29}$$

Substituting into Eqn. 20:

$$|L(\theta; D) - L(\theta; S)| \leq 2(c|G_s| + 1) \left(\sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2n}} + \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2m}} \right). \tag{30}$$

Letting $M = \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2n}} + \sqrt{\frac{\ln(4|\Theta|) - \ln(\delta)}{2m}}$, we obtain the final bound:

$$|L(\theta; D) - L(\theta; S)| \leq 2(c|G_s| + 1)M. \tag{31}$$

□

F.3 Proof of Theorem 5.2

Proof. Our proof consists of the following steps.

Step 1. We start by decomposing $\mathbb{E}[t^*(G)]$ into two components: the invariant subgraph G_c and a partially retained spurious subgraph $G_s^{\mathcal{P}}$.

$$\begin{aligned} \mathbb{E}[t^*(G)] &= \mathbb{E}\left[G_c + G_s^{\mathcal{P}}\right] \\ &= \mathbb{E}\left[G_c\right] + \mathbb{E}\left[G_s^{\mathcal{P}}\right] \\ &= G_c + \mathbb{E}\left[G_s^{\mathcal{P}}\right] \end{aligned} \tag{32}$$

In Eqn. 32, $\mathbb{E}\left[G_c\right] = G_c$ is due to that for any given label y, G_c is a constant according to Assumption 1, while $G_s^{\mathcal{P}}$ is a random variable.

Step 2. We then model $G_s^{\mathcal{P}}$ as a set of independent edges, and calculate the expected total edge weights of G_c and $G_s^{\mathcal{P}}$ respectively. First, we define W_c as the sum of binary random variables corresponding to the edges in G_c. Each edge e_{ij} in G_c is associated with a Bernoulli random variable X_{ij} such that:

$$W_c = \sum_{e_{ij} \in G_c} X_{ij}. \tag{33}$$

Similarly, we define $W_s^{\mathcal{P}}$ as the sum of binary random variables corresponding to the edges in $G_s^{\mathcal{P}}$. Each edge e_{ij} in $G_s^{\mathcal{P}}$ is associated with a Bernoulli random variable X'_{ij} such that:

$$W_s^{\mathcal{P}} = \sum_{e_{ij} \in G_s^{\mathcal{P}}} X'_{ij}. \tag{34}$$

W_c and $W_s^{\mathcal{P}}$ are denoted as random r.v. for the total edge weights of G_c and $G_s^{\mathcal{P}}$.

Step 3. We then calculate the expected edge weights $\mathbb{E}[W_c]$ and $\mathbb{E}[W_s^{\mathcal{P}}]$ as following.

$$\mathbb{E}[W_c] = \mathbb{E}[\sum_{e_{ij} \in G_c} X_{ij}] = \sum_{e_{ij} \in G_c} \mathbb{E}[X_{ij}] = |G_c|\,, \tag{35}$$

$$\mathbb{E}[W_s^{\mathcal{P}}] = \mathbb{E}[\sum_{e_{ij} \in G_s^{\mathcal{P}}} X'_{ij}] = \sum_{e_{ij} \in G_s^{\mathcal{P}}} \mathbb{E}[X'_{ij}] = \frac{|G_s^{\mathcal{P}}|}{|\mathcal{E}|} = \frac{\eta|\mathcal{E}| - |G_c|}{|\mathcal{E}|}. \tag{36}$$

Here $\mathcal{E}$ is the set of edges in graph G, $\eta|\mathcal{E}|$ is the total edge number limits due to $\mathcal{L}_e$. In Eqn. 35, $\mathbb{E}[X_{ij}] = 1, \forall e_{ij} \in G_c$ is due to that $\mathbb{P}(X_{ij}) = 1$, as $t^*(G)$ always include G_c using the results from Prop. 1; In Eqn. 36, $\mathbb{E}[X'_{ij}] = \frac{1}{|\mathcal{E}|}, \forall e_{ij} \in G_s^{\mathcal{P}}$, due to that $\mathbb{P}(X'_{ij}) = \frac{1}{|\mathcal{E}|}$ enforced by ϵ-probability alignment penalty $\mathcal{L}_s$. Therefore, given a suitable η that prunes spurious edges from G_s, $|\mathcal{E}||G_c| \gg \eta|\mathcal{E}| - |G_c|$, i.e., $\mathbb{E}[t^*(G)]$ will be dominated by G_c in terms of edge probability mass, therefore, we conclude that $G_c \approx \mathbb{E}[t^*(G)]$. □

G More Discussion on Pruning-based Learning Paradigm

While PrunE focuses on pruning spurious edges for OOD generalization, recent studies [63, 65] attempt to disentangle spurious features from ERM-learned representations in the latent space, demonstrating capability in capturing more invariant substructures [65]. In Table 5, we provide additional comparisons with LIRS [65].

For synthetic dataset, LIRS and PrunE exhibit notable differences in performance across datasets with different characteristics. Specifically, on the Motif-base dataset, PrunE achieves 91.40% accuracy, significantly outperforming LIRS (75.51%). In contrast, on the Motif-size dataset, LIRS performs better than PrunE. This also hightlights the different inductive bias between these two methods: For a graph with more complex spurious patterns and large graph size, PrunE struggle to prune all the spurious edges, thus leading to decreased performance, while LIRS adopt self-supervised learning to capture spurious features first, a smaller ratio of G_c would lead to more effective spuriosity learning, thus facilitating the subsequent feature disentanglement. In contrast, the learning of spurious features would be more challenging in Motif-base due to the large ratio of invariant subgraph.

In real-world datasets, we observe that the performance of PrunE is on par with LIRS, however, PrunE presents several advantages over LIRS [65] and EQuAD [64]. Specifically, (1) LIRS involves multiple stages, with nearly 100 hyperparameter combinations in total; In contrast, PrunE demonstrates robust performance with limited set of hyperparameters across datasets, greatly reducing model tuning efforts. Moreover, as PrunE only introduces two lightweight regularization terms, it is highly efficient in runtime and memory cost, and is 3.15x faster than LIRS in terms of running time. (2) LIRS operates in latent space and thus lacks interpretability in terms of input structures. PrunE, by operating in the input space, not only being efficient and effective, but also offers interpretability by identifying critical subgraphs that explain the model prediction.

Table 5: Test OOD performance on synthetic and realworld datasets.

	Motif		GOODHIV		EC50		
	Base	Size	Scaffold	Size	Assay	Size	Scaffold
LIRS	75.51	**74.51**	**72.82**	**66.64**	76.81	64.20	65.11
PrunE	**91.4**	66.53	71.84	64.99	**78.01**	**65.46**	**67.56**

H Complexity Analysis

Time Complexity. The time complexity is $\mathcal{O}(CkmF)$, where k is the number of GNN layers, m is the total number of edges in graph G, and F is the feature dimensions. Compared to ERM, PrunE incurs an additional constant $C > 1$, as it uses a GNN model $t(\cdot)$ for edge selection, and another GNN encoder $h(\cdot)$ for learning feature representations. However, C is a small constant, hence the time cost is on par with standard ERM.

Space Complexity. The space complexity for PrunE is $\mathcal{O}(C'|\mathcal{B}|mkF)$, where $|\mathcal{B}|$ denotes the batch size. The constant $C' > 1$ is due to the additional subgraph selector $t(\cdot)$. As C' is also a small integer, the space complexity of PrunE is also on par with standard ERM.

In Table 6, we report the memory consumption and runtime of various OOD methods. While PrunE incurs higher overhead than IRM and VRex due to the usage of subgraph selector, it remains more efficient than most graph OOD baselines. This is attributed to the use of two lightweight OOD objectives, in contrast to the more computationally intensive operations such as data augmentations and contrastive training employed by other methods. Notably, PrunE is **3.15×** faster than LIRS on the Molbbbp dataset, owing to its single-stage training paradigm. When considering the additional overhead from hyperparameter tuning, the runtime advantage of PrunE becomes even more pronounced.

Table 6: Memory consumption and running time on Motif-base and OGBG-Molbbbp datasets.

(a) Memory overhead (in MB).

Method	Motif-base	Molbbbp
ERM	40.62	32.43
IRM	51.76	36.19
VRex	51.52	35.92
GREA	103.22	76.28
GSAT	90.12	58.02
CIGA	104.43	72.47
AIA	99.29	81.55
LIRS	89.15	107.37
PrunE	74.15	61.07

(b) Runtime (in Second)

Method	Motif-base	Molbbbp
ERM	494.34 ± 117.86	92.42 ± 0.42
IRM	968.94 ± 164.09	151.84 ± 7.53
VRex	819.94 ± 124.54	129.13 ± 12.93
GREA	1612.43 ± 177.36	262.47 ± 45.71
GSAT	1233.68 ± 396.19	142.47 ± 25.71
CIGA	1729.14 ± 355.62	352.14 ± 93.32
AIA	1422.34 ± 69.33	217.36 ± 11.04
LIRS	504.87 ± 24.04	421.32 ± 19.86
PrunE	501.62 ± 7.64	133.35 ± 3.47

I The Pitfall of Directly Identifying Edges in G_c

Most graph-specific OOD methods that model edge probabilities incorporate OOD objectives as regularization terms for ERM. These OOD objectives attempt to directly identify the invariant subgraph for OOD generalization. For example, GSAT [42] utilizes the information bottleneck to learn a minimal sufficient subgraph for accurate model prediction; CIGA [9] adopt supervised contrastive learning to identify the invariant subgraph that remains stable across different environments within the same class; DIR [59] and AIA [52] identify the invariant subgraph through training environments augmentation. However, when spurious substructures exhibit comparable or stronger correlation strength than invariant edges (i.e., edges in G_c) with the targets, these methods are unlikely to identify all invariant edges, and preserve the invariant subgraph patterns Since the spurious substructure may be mistakenly identified as the stable pattern. Consequently, while achieving high training accuracy, these methods suffer from poor validation and test performance.

In contrast, PrunE avoids this pitfall by proposing OOD objectives that focus on pruning uninformative spurious edges rather than directly identifying causal ones. While strongly correlated spurious edges may still persist, edges in G_c are preserved due to their strong correlation with targets. As a key conclusion, PrunE achieves enhanced OOD performance compared to prior methods, as the invariant patterns are more likely to be retained, even if some spurious edges cannot be fully excluded.

J More Details about Experiments

J.1 Datasets details

In our experimental setup, we utilize five datasets: GOOD-HIV, GOOD-Motif, SPMotif, OGBG-Molbbbp, and DrugOOD. The statistics of the datasets are illustrated in Table 7.

GOOD-HIV [15]. GOOD-HIV is a molecular dataset derived from the MoleculeNet [60] benchmark, where the primary task is to predict the ability of molecules to inhibit HIV replication. The molecular

structures are represented as graphs, with nodes as atoms and edges as chemical bonds. Following [15], We adopt the covariate shift split, which refers to changes in the input distribution between training and testing datasets while maintaining the same conditional distribution of labels given inputs. This setup ensures that the model must generalize to unseen molecular structures that differ in these domain features from those seen during training. We focus on the Bemis-Murcko scaffold [3] and the number of nodes in the molecular graph as two domain features to evaluate our method.

GOOD-Motif [15]. GOOD-Motif is a synthetic dataset designed to test structure shifts. Each graph in this dataset is created by combining a base graph and a motif, with the motif solely determining the label. The base graph type and the size are selected as domain features to introduce covariate shifts. By generating different base graphs such as wheels, trees, or ladders, the dataset challenges the model's ability to generalize to new graph structures not seen during training. We employ the covariate shift split, where these domain features vary between training and testing datasets, reflecting real-world scenarios where underlying graph structures may change.

SPMotif [59]. In SPMotif datasets [59], each graph comprises a combination of invariant and spurious subgraphs. The spurious subgraphs include three structures (Tree, Ladder, and Wheel), while the invariant subgraphs consist of Cycle, House, and Crane. The task for a model is to determine which one of the three motifs (Cycle, House, and Crane) is present in a graph. A controllable distribution shift can be achieved via a pre-defined parameter b. This parameter manipulates the spurious correlation between the spurious subgraph G_s and the ground-truth label Y, which depends solely on the invariant subgraph G_c. Specifically, given the predefined bias b, the probability of a specific motif (e.g., House) and a specific base graph (Tree) will co-occur is b while for the others is $(1-b)/2$ (e.g., House-Ladder, House-Wheel). When $b = \frac{1}{3}$, the invariant subgraph is equally correlated to the three spurious subgraphs in the dataset.

OGBG-Molbbbp [18]. OGBG-Molbbbp is a real-world molecular dataset included in the Open Graph Benchmark [18]. This dataset focuses on predicting the blood-brain barrier penetration of molecules, a critical property in drug discovery. The molecular graphs are detailed, with nodes representing atoms and edges representing bonds. Following **(author?)** [52], we create scaffold shift and graph size shift to evaluate our method. Similarly to **(author?)** [15], the Bemis-Murcko scaffold [3] and the number of nodes in the molecular graph are used as domain features to create scaffold shift and size shift respectively.

DrugOOD [21]. DrugOOD dataset is designed for OOD challenges in AI-aided drug discovery. This benchmark offers three environment-splitting strategies: Assay, Scaffold, and Size. In our study, we adopt the EC50 measurement. Consequently, this setup results in three distinct datasets, each focusing on a binary classification task for predicting drug-target binding affinity.

Table 7: Details about the datasets used in our experiments.

DATASETS	Split	# TRAINING	# VALIDATION	# TESTING	# CLASSES	METRICS
GOOD-Motif	Base	18000	3000	3000	3	ACC
	Size	18000	3000	3000	3	ACC
SPMotif	Correlation	9000	3000	3000	3	ACC
GOOD-HIV	Scaffold	24682	4113	4108	2	ROC-AUC
	Size	26169	4112	3961	2	ROC-AUC
OGBG-Molbbbp	Scaffold	1631	204	204	2	ROC-AUC
OGBG-Molbace	Scaffold	1210	152	151	2	ROC-AUC
EC50	Assay	4978	2761	2725	2	ROC-AUC
	Scaffold	2743	2723	2762	2	ROC-AUC
	Size	5189	2495	2505	2	ROC-AUC

J.2 Detailed experiment setting

GNN Encoder. For GOOD-Motif datasets, we utilize a 4-layer GIN [61] without Virtual Nodes [14], with a hidden dimension of 300; For GOOD-HIV datasets, we employ a 4-layer GIN without Virtual Nodes, and with a hidden dimension of 128; For the OGBG-Molbbbp dataset, we adopt a 4-layer GIN with Virtual Nodes, and the dimensions of hidden layers is 64; For the DrugOOD datasets, we

use a 4-layer GIN without Virtual Nodes. For SPMotif datasets, we use a 5-layer GIN without Virtual Nodes. All GNN backbones adopt sum pooling for graph readout.

Training and Validation. By default, we use Adam optimizer [24] with a learning rate of $1e-3$ and a batch size of 64 for all experiments. For DrugOOD, GOOD-Motif and GOOD-HIV datasets, our method is pretrained for 10 epochs with ERM, and for other datasets, we do not use ERM pretraining. We employ an early stopping of 10 epochs according to the validation performance for DrugOOD datasets and GOOD-Motif datasets, and do not employ early stopping for other datasets. Test accuracy or ROC-AUC is obtained according to the best validation performance for all experiments. All experiments are run with 4 different random seeds, the mean and standard deviation are reported using the 4 runs of experiments.

Baseline setup and hyperparameters. In our experiments, for the GOOD and OGBG-Molbbbp datasets, the results of ERM, IRM, GroupDRO, and VREx are reported from [15], while the results for DropEdge, DIR, GSAT, CIGA, GREA, FLAG, $\mathcal{G}$-Mixup and AIA on GOOD and OGBG datasets are reported from [52]. To ensure fairness, we adopt the same GIN backbone architecture as reported in [52]. For the EC50 datasets and SPMotif datasets, we conduct experiments using the provided source codes from the baseline methods. The hyperparameter search is detailed as follows.

For IRM and VREx, the weight of the penalty loss is searched over $\{1e-2, 1e-1, 1, 1e1\}$. For GroupDRO, the step size is searched over $\{1.0, 1e-1, 1e-2\}$. The causal subgraph ratio for DIR is searched across $\{1e-2, 1e-1, 0.2, 0.4, 0.6\}$. For DropEdge, the edge masking ratio is searched over: $\{0.1, 0.2, 0.3\}$. For GREA, the weight of the penalty loss is tuned over $\{1e-2, 1e-1, 1.0\}$, and the causal subgraph size ratio is tuned over $\{0.05, 0.1, 0.2, 0.3, 0.5\}$. For GSAT, the causal graph size ratio is searched over $\{0.3, 0.5, 0.7\}$. For CIGA, the contrastive loss and hinge loss weights are searched over $\{0.5, 1.0, 2.0, 4.0, 8.0\}$. For DisC, we search over q in the GCE loss: $\{0.5, 0.7, 0.9\}$. For LiSA, the loss penalty weights are searched over:$\{1, 1e-1, 1e-2, 1e-3\}$. For $\mathcal{G}$-Mixup, the augmented ratio is tuned over $\{0.15, 0.25, 0.5\}$. For FLAG, the ascending steps are set to 3 as recommended in the paper, and the step size is searched over $\{1e-3, 1e-2, 1e-1\}$. For AIA, the stable feature ratio is searched over $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, and the adversarial penalty weight is searched over $\{0.01, 0.1, 0.2, 0.5, 1.0, 3.0, 5.0\}$.

Hyperparameter search for `PrunE`. For `PrunE`, the edge budget η in $\mathcal{L}_e$ is searched over: $\{0.5, 0.75, 0.85\}$; K for the $K\%$ edges with lowest probability score in $\mathcal{L}_s$ is searched over:$\{50, 70, 90\}$; λ_1, λ_2 for balancing $\mathcal{L}_e$ and $\mathcal{L}_s$ are searched over: $\{10, 40\}$ and $\{1e-1, 1e-2, 1e-3\}$ respectively. The encoder of subgraph selector $t(\cdot)$ is searched over $\{GIN, GCN\}$, with the number of layers: $\{2, 3\}$.

J.3 More Experimental Results

We provide more experiment details regarding: (1) Experiment results when there are multiple invariant substructures in a graph. (2) Experiment results for more application domains. (3) Ablation study on ERM pretraining. (4) The capability of `PrunE` of identifying spurious edges. (5) More visualization results on GOOD-Motif datasets in Figure 6 and Figure 7. (6) Hyperparameter sensitivity analysis on Motif-base, OGBG-Molbbbp, and EC50 assay datasets, in Figure 8.

Table 8: Experimental results on SPMotif datasets with 2 invariant subgraphs in each graph.

Method	SPMotif ($\#G_c = 2$)		
	$b = 0.40$	$b = 0.60$	$b = 0.90$
ERM	53.48±3.31	52.59±4.61	56.76±8.06
IRM	52.47±3.63	55.62±7.90	48.66±2.33
VRex	49.68±8.66	48.89±4.79	47.97±2.61
GSAT	59.34±7.96	58.43±10.64	55.68±3.18
GREA	64.87±5.76	67.66±6.29	59.40±10.26
CIGA	69.74±6.81	71.19±2.46	<u>65.83±10.41</u>
AIA	**71.61±2.09**	<u>72.01±2.13</u>	58.14±4.21
PrunE	<u>70.41±7.53</u>	**74.61±3.17**	**66.75±4.33**

Model performance for graphs with multiple invariant subgraphs. While Assumption 1 assumes the existence of a single invariant substructure causally related to each target label, many real-world graph applications [18, 15] may contain multiple such invariant subgraphs. However, Assumption 1 can be reformulated to accommodate multiple G_c without compromising the validity of our assumptions and theoretical results. Specifically, suppose there are K invariant subgraphs, denoted as $G_{c,i}$ for $i \in [K]$. For any specific $G_{c,i}$, the spurious subgraph G_s' can be redefined as $G_s' = G_s \cup \{G_{c,j} \mid j \neq i\}$. Given this redefinition, and under the presence of G_s, Assumption 1 still holds. Consequently, the assumptions and theoretical results presented in this work remain valid, even when multiple G_c exist within the datasets. To further support our claim, we curated a dataset based on SPMotif [59], where in the train/valid/test datasets, two invariant substructures are attached to the spurious subgraph. Our method performs effectively under this scenario, as shown in Table 8.

Experiment results on more application domains. To further evaluate the effectiveness of PrunE across different application domains, we conduct experiments on GOOD-CMNIST [15] and Graph-Twitter [50, 68] datasets, the evaluation metric for these datasets is accuracy.

Table 9: Test performance on GOOD-CMNIST and Graph-Twitter datasets.

Method	CMNIST	Graph-Twitter
ERM	28.60±1.87	60.47±2.24
IRM	27.83±2.13	56.93±0.99
Vrex	28.48±2.87	57.54±0.93
DisC	24.99±1.78	48.61±8.86
GSAT	28.17±1.26	60.96±1.18
GREA	29.02±3.26	59.47±2.09
CIGA	32.22±2.67	<u>62.31±1.63</u>
AIA	**36.37±4.44**	61.10±0.47
PrunE	<u>33.89±1.65</u>	**63.37±0.76**

As demonstrated in Table 9, PrunE also achieves superior performance in application domains beyond molecular applications, indicating its superior OOD performance and broad applicability.

Ablation study on ERM pretraining. We conduct ablation study across 5 datasets without using ERM pretraining. The results are presented in Table 10. As illustrated, incorporating ERM pretraining improves OOD performance in most cases, as the GNN encoder is able to learn useful representations before incorporating $\mathcal{L}_e$ and $\mathcal{L}_s$ to train $t(\cdot)$. Intuitively, this facilitates the optimization of $t(\cdot)$, therefore improving the test performance.

Table 10: Ablation study on test datasets.

	Motif-basis	Motif-size	EC50-Assay	EC50-Sca	HIV-size
w/ pretraining	91.48±0.40	66.53±8.55	78.01±0.42	67.56±1.63	64.99±1.63
w/o pretraining	91.04±0.76	61.48±8.29	76.58±2.14	66.19±1.56	65.46±1.85

The capability of PrunE to identify spurious edges. To verify the ability of PrunE to identify spurious edges while preserving critical edges in G_c, we conduct experiments and provide empirical results on *Recall@*K and *Precision@*K on GOODMotif datasets, where K denotes the $K\%$ edges with lowest estimated probability scores. As illustrated in Table 11, PrunE is able to identify a subset of spurious edges with precision higher than 90% across all datasets, even with $K = 50$, indicating that PrunE can preserve G_c, thus enhancing the OOD generalization performance.

Visualization results on GOOD-Motif datasets. We provide more visualization results on GOOD-Motif datasets in Figure 6 and Figure 7, in which the blue nodes represent the ground-truth nodes in G_c, and blue edges are estimated edges by $t^*(\cdot)$. We visualize top-K edges with highest probability scores derived from $t(\cdot)$. As shown, PrunE is able to identify edges in G_c, demonstrating the effectiveness of pruning spurious edges, and aligns with the theoretical results from Theorem 5.2.

Table 11: Recall@K and Precision@K for Motif-base and Motif-size datasets, where K denotes the $K\%$ edges with lowest estimated probability scores.

K%	Motif-base		Motif-size	
	Recall	Precision	Recall	Precision
10%	0.1467	1.0000	0.0963	0.9199
20%	0.3076	0.9831	0.2023	0.9602
30%	0.4556	0.9465	0.3093	0.9735
40%	0.6056	0.9374	0.4153	0.9801
50%	0.7356	0.9017	0.5243	0.9841

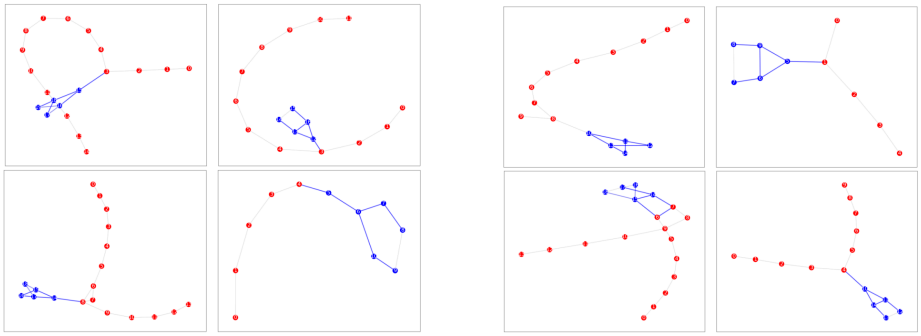


Figure 6: More visualization results on Motif-base dataset. The blue nodes are ground-truth nodes in G_c, and red nodes are ground-truth nodes in G_s. The highlighted blue edges are top-K edges predicted by $t^*(\cdot)$, where K is the number of ground-truth edges from G_c in a graph.

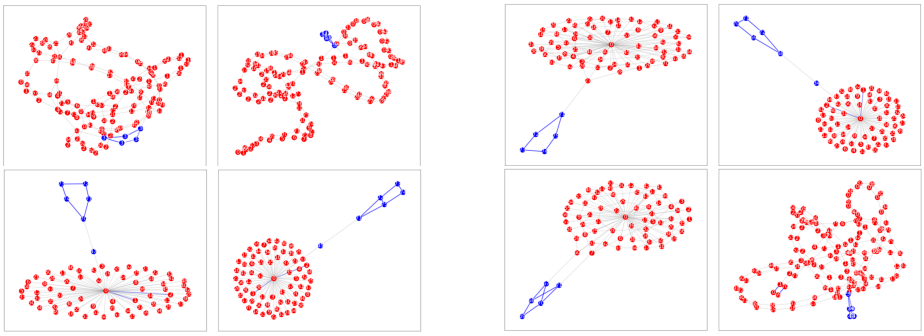


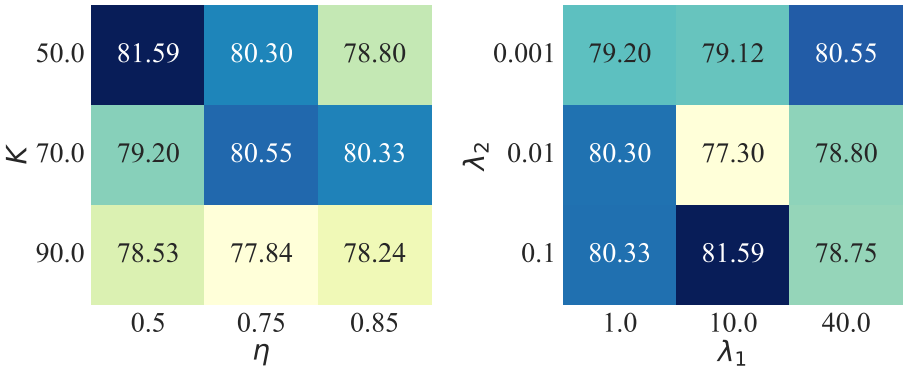
Figure 7: More visualization results on Motif-size dataset. The blue nodes are ground-truth nodes in G_c, and red nodes are ground-truth nodes in G_s. The highlighted blue edges are top-K edges predicted by $t^*(\cdot)$, where K is the number of ground-truth edges from G_c in a graph.

Hyperparameter sensitivity. We provide more experimental results on hyperparameter sensitivity on synthetic and real-world datasets. As shown in Figure 8, `PrunE` exhibits stable performance across the real-world datasets, highlighting its robustness to varying hyperparameter configurations. For the Motif-base dataset, the performance is not as stable as on real-world datasets. However this behavior is expected: when η is set too low, `PrunE` may mistakenly prune invariant edges, resulting in performance degradation. Since G_s constitutes around 50% of the graph in Motif-base dataset, setting $\eta = 0.5$ leads to the removal of edges in G_c, ultimately causing failure in OOD generalization. Therefore, setting $\eta \geq 0.75$ leading to enhanced OOD performance.

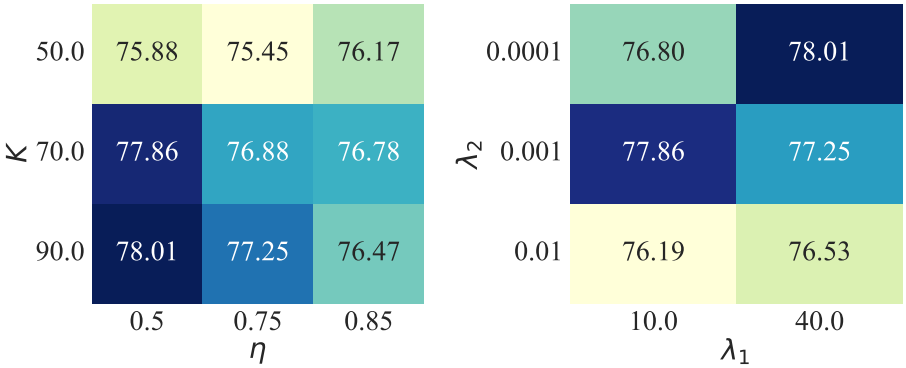
K Limitations of `PrunE`

While our proposed framework introduces a novel and effective pruning-based paradigm for invariant learning, it is currently tailored to graph-structured data due to its reliance on a learnable subgraph selector. Extending this approach to other data modalities, such as text or images, remains non-trivial.

(a) Hyperparameter sensitivity on Motif-base dataset.



(b) Hyperparameter sensitivity on OGBG-Molbbbp size.



(c) Hyperparameter sensitivity on EC50 assay.

Figure 8: Hyperparameter sensitivity analysis across different datasets.

Additionally, although our method effectively removes a significant portion of spurious edges, some spurious edges may still persist due to their strong correlation with target labels. Developing more effective pruning approaches is an important direction for our future research.

L Software and Hardware

We conduct all experiments using PyTorch [43] (v2.1.2) and PyTorch Geometric [12] on Linux servers equipped with NVIDIA RTX4090 GPUs and CUDA 12.1.