A CLOSER LOOK AT DISTRIBUTION SHIFTS AND OUT-OF-DISTRIBUTION GENERALIZATION ON GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Distribution shifts, in which the training distribution differs from the testing distribution, can significantly degrade the performance of Graph Neural Networks (GNNs). Although some existing graph classification benchmarks consider distribution shifts, we are far from understanding the effects of distribution shifts on graphs, and more specifically how they differ from distribution shifts in tensor data like images. We ask: (1) how useful are existing domain generalization methods for tackling distribution shifts on graph data? (2) are GNNs capable of generalizing to test graphs from unseen distributions? As a first step to answering these questions, we curate GDS, a benchmark of 8 datasets reflecting a diverse range of distribution shifts across graphs. We observe that in most cases, we need both a suitable domain generalization algorithm and a strong GNN backbone model to optimize out-of-distribution test performance. However, even if we carefully pick such combinations of models and algorithms, the out-of-distribution performance is still much lower than the in-distribution performance. This large gap emphasizes the need for domain generalization algorithms specifically tailored for graphs and strong GNNs that generalize well to out-of-distribution graphs. To facilitate further research, we provide an open-source package that administers the GDS benchmark with modular combinations of popular domain generalization algorithms and GNN backbone models.

1 INTRODUCTION

Distribution shifts, in which training and testing distributions differ, often make machine learning systems fail in spectacular ways (Torralba & Efros, 2011). Typically, when tested outside the distribution of training examples, the performance of learning systems degrades significantly. The over-reliance on the training distribution makes it challenging to apply systems in practical scenarios where distribution shifts are common, such as graph classification problems.

Graphs are standard data structures that abstract complex systems of interacting objects, such as molecular graphs, biological networks (Szklarczyk et al., 2019), and social networks (Dou et al., 2021). Distribution shifts arise naturally in many graph classification problems since it is infeasible to prepare a training set that covers all domains of interest. In problems such as drug discovery and social media fact-checking, molecular graph structure often differs at inference (Macarron et al., 2011; Wu et al., 2018; Hu et al., 2020), and news propagation graphs may grow with time (Hassan et al., 2017). See Section 3 for a detailed treatment of these application contexts.

In this paper, we consider *domain generalization* (see Section 2 for a detailed problem definition), where training and test graphs are collected from related but different domains (see Fig. 1). Such domain shifts challenge the *out-of-distribution (OOD) generalization* abilities of graph neural networks (GNNs), as we often see that OOD test performance is significantly lower than in-distribution (ID) test performance if we randomly shuffle training and test splits.

Despite the real-world prevalence of distribution shifts on graphs, they have not been thoroughly evaluated and discussed to the best of our knowledge. Graph learning benchmarks such as Hu et al. (2020) note the broad existence of distribution shifts on graph data and have considered using domain information to split training and test sets. We further aim to (1) provide domain labels separately from features which allows the applications of domain generalization algorithms; (2) characterize the types of domain shifts exhibited in the datasets using statistical means; (3) and discuss

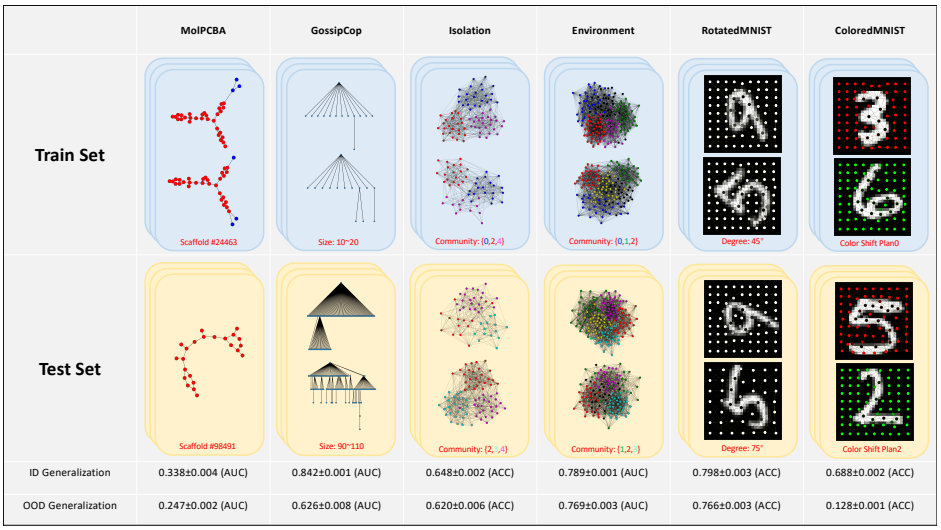


Figure 1: GDS datasets and types of domain shifts reflected, along with the out-of-distribution and in-distribution generalization performance with Graph Isomorphism Network (GIN) and Empirical Risk Minimization (ERM) (i.e., no transfer).

Table 1: OOD generalization algorithms, augmentation algorithms, and GNN models/techniques considered in this paper.

Types	**Domain Generalization and Augmentation Algorithms**
Baselines	ERM
DRO	GroupDRO (Sagawa et al., 2019)
IRM	IRM (Arjovsky et al., 2019)
Distribution Matching	DeepCORAL (Sun & Saenko, 2016) DANN (Ganin et al., 2016) DANN-Graph
Meta Learning	MLDG (Li et al., 2018)
Augmentation	FLAG (Kong et al., 2020) SA

Types	**GNN Models & Techniques**
Baselines	GIN (Xu et al., 2018) MLP GIN-Deep
Global	Virtual Node (Gilmer et al., 2017)
Spectral	ChebNet (Defferrard et al., 2016)
WL-Isomorphism	3WL-GNN (Maron et al., 2019)
Structure-Aware	GSN (Bouritsas et al., 2020)

what challenges such domain shifts impose on GNN models. By only looking at the associated benchmark leader-boards, discerning whether a method is highly performant due to its expressive power or its robustness to certain domain shifts is difficult.

In this paper, we curate GDS (Graph Distribution Shift), a benchmark of 8 datasets reflecting a diverse range of distribution shifts across graphs; see Fig. 1. We release **GDS**, an open-source framework that administers our curated graph-distribution-shift benchmarks with modular combinations of popular domain generalization algorithms and GNN backbone models. This framework facilitates further research and streamlines rigorous and reproducible experiments. Details of the datasets are discussed in Section 3.

Another major contribution of this work is a rigorous comparison of methods for improving OOD generalization on graphs with respect to **GDS**. The three high-level categories considered are (1) existing OOD generalization algorithms; (2) augmentations for graph data; (3) and GNN models and techniques that improve OOD generalization. See Table 1 for a list of algorithms and models considered in this paper and Section 4 for their details.

Domain generalization algorithms aim to bias a model towards learning statistical invariances across the training distributions under the assumption that such invariances hold in unseen test domains. These methods specify both a prior on the types of invariances desired and an algorithm for estimating them from training samples. Since the specific statistical invariances these algorithms learn may be highly dependent on the type of data and the network architectures to which they are applied, each technique's applicability to graph data and GNNs models is unclear. Given the diverse array of domain generalization algorithms available (Table 1), and the absence of a thorough analysis on

the topic to date[1], we investigate the effectiveness of existing domain generalization methods when tackling domain shifts on graph-structured data.

Although data augmentation methods are a standard component in image classification pipelines, augmentation methods for graph data are not as well studied or prevalent in practice. We consider two types of graph augmentations: (1) augmentations to node features following an adversarial-based augmentation algorithm (Kong et al., 2020) (FLAG); (2) and augmentations to graph structures using the node dropping and edge perturbation operations proposed in You et al. (2020).

The expressive power of the backbone model is also an important factor for generalizing to unseen test data[2]. Some GNN models or techniques may improve OOD generalization performance as they may be robust to certain domain shifts on graphs. For example, ChebNet (Defferrard et al., 2016), a spectral graph convolution model, may enhance OOD generalization when the domain shift is better learned in the Fourier domain, while Virtual Node (Gilmer et al., 2017; Li et al., 2017) may improve OOD generalization performance when the domain shift is about the global context caused by the introduction of additional nodes connected to all nodes in the original graph.

The results of our comprehensive evaluation over a broad range of OOD generalization and graph augmentation algorithms combined with various GNN models and techniques support two primary conclusions (demonstrated in Tables 3 to 7):

Most OOD generalization algorithms fail to work when applied to domain shifts on graphs.
Combinations of best performing GNN models and augmentation techniques usually achieve state-of-the-art performance in OOD generalization on graphs.

These observations underscore the need for new OOD generalization algorithms that are specifically tailored for domains shifts on graphs. We carefully discuss the success and failure modes of many algorithms and models in Section 5, and we hope our work can assist the community in better understanding the distinct challenges posed by domain shifts on graphs.

2 Problem Setting

Supervised learning on graphs and graph classification. The goal of supervised graph learning is to train a featurizer $\phi : \mathcal{G} \to \mathcal{H}$ which maps a graph $G = (V, E)$ with node attributes X_v for $v \in V$ and edge attributes $e_{u,v}$ for $(u, v) \in E$ to a representation vector h_G, which can then be used to predict the label y of graph G through a classifier $\omega : \mathcal{H} \to \mathcal{Y}$ in graph classification problems.

Domain generalization on graphs. The problem of domain generalization on graphs is an extension of supervised learning where the data distribution is a mixture of D domains $\{1, \cdots, D\}$, each characterized by a dataset $S^d = \{(G_i^d, y_i^d)\}_{i=1}^{n_d}$ consisting of i.i.d. samples from distribution $P(G^d, y^d)$. We train on the training domains $\{1, \cdots, D_{\text{train}}\}$, and the goal is *out-of-distribution (OOD) generalization* to D_{test} test domains $\{D_{\text{train}} + 1, \cdots, D_{\text{train}} + D_{\text{test}}\}$. The test domains are not accessible during training, which differs from unsupervised domain adaptation, where unlabeled data from test domains are available during training. The sample space of graph structure is high dimensional and discrete, exhibiting a different nature than the Euclidean space, $\mathbb{R}^d$, of conventional vector features. An OOD generalization algorithm that learns the statistical invariances of the training distributions on graphs should also respect permutation symmetries, making the challenging OOD generalization problem even harder on graphs. See extended commentary on this distinction in Appendix A.

3 GDS Datasets

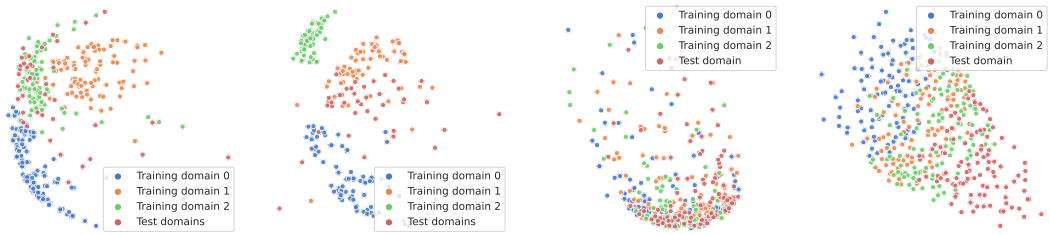
At the heart of our experiments is the design of graph datasets with rich distribution shifts; see Table 2 for more information and Fig. 1 for illustrations. For each dataset, we either prepare or recover the domain information, and as an OOD generalization benchmark, the domain labels d for each

[1] Koh et al. (2021) reports the performance of groupDRO (Sagawa et al., 2019), IRM (Arjovsky et al., 2019), and deepCORAL (Sun & Saenko, 2016) on the OGB-MOLPCBA dataset which we also consider.

[2] By using a large enough backbone model, Gulrajani & Lopez-Paz (2020) shows empirical risk minimization can achieve state-of-the-art performance in domain generalization on images.

Table 2: Statistics of the GDS datasets.

Dataset	MolHIV	MolPCBA	PPA	GossipCop	Isolation	Environment	RotatedMNIST	ColoredMNIST
Input	mol-graph	mol-graph	mol-graph	prop-networks	SBM graphs	SBM graphs	super-pixel graphs	super-pixel graphs
Prediction	mol-properties	mol-properties	mol-properties	fake news	edge prob.	compositions	digits	digits
Domain	scaffold	scaffold	species	size	compositions	compositions	rotation degree	color shift
Avg. Graph Size	25.5	26.0	243.4	57.5	58.5	147.7	66.0	67.0
# of Domains	19,089	120,084	1,581	10	10	4	6	3
# of Samples	41,127	437,929	158,100	5,464	20,000	60,000	70,000	70,000
Type	real-world Hu et al. (2020)	real-world Hu et al. (2020)	real-world Hu et al. (2020)	semi-artificial Dou et al. (2021)	artificial	artificial	semi-artificial Ghifary et al. (2015)	semi-artificial Arjovsky et al. (2019)



(a) OGB-MOLHIV **(b)** OGB-MOLPCBA **(c)** OGB-PPA **(d)** UPFD-GOSSIPCOP

Figure 2: Scatter plots visualizing the training and test domain distributions of some GDS datasets, using the embeddings learned by a Weisfeiler-Lehman (WL) subtree kernel.

sample are included in the datasets. In this section, we characterize the domains and the type of domain shifts in each dataset while relegating the details to Appendix B.

Maximum common subgraph: OGB-MOLHIV and OGB-MOLPCBA are two real-world molecular graph datasets adopted from Wu et al. (2018); Hu et al. (2020). Each graph is an abstraction of a molecule, where nodes are atoms and edges are chemical bonds. The task is to predict molecular properties, e.g., whether the given molecule inhibits HIV virus replication in OGB-MOLHIV. The domains are defined by scaffolds, the core structures of small molecules (Bemis & Murcko, 1996), which can be translated into the language of graph theory as the maximal isomorphic subgraph. In Fig. 1, we see that molecular graphs within a domain share a maximum common subgraph (i.e., the scaffold). They differ from each other only on a few atoms added to different locations of the scaffold. The graph edit distances (Sanfeliu & Fu, 1983) between graphs in the same domain are very small but can be unbounded across domains. We have many domains in the training, validation, and test splits, as most of the domains contain few samples. Each of the test domains in the two datasets has only a single sample, which imposes yet another challenge to some OOD generalization algorithms such as DANN (Ganin et al., 2016); see Section 4.

Multi-hop neighborhoods: OGB-PPA is a real-world dataset of protein association networks adopted from (Szklarczyk et al., 2019; Hu et al., 2020). The graphs in OGB-PPA are abstractions of protein-protein association relations of $1,581$ species, and the task is to predict which taxonomic group (out of 37 groups) a graph originates from. Each graph in OGB-PPA is a subgraph sampled from the 2-hop neighborhood of a protein, where the species of the center protein defines the domain. Thus, graphs within a domain share a common type of center node (center proteins are identical species). Although the center node is always removed in sub-sampling, we believe the sampled neighborhoods around proteins of a specific species exhibit sufficient similarity to be considered as a coherent domain.

Size growth: UPFD-GOSSIPCOP is a real-world dataset consisting of news propagation graphs adopted from (Dou et al., 2021; Shu et al., 2020). The propagation networks are abstractions of social engagement information (retweets between users) collected from Twitter, built according to fact-check information from Gossipcop and extracted by FakeNewsNet (Shu et al., 2020). Given this propagation network, the task is to predict the credibility of news, either real or fake. We split the entire dataset into ten domains according to graph sizes, where each domain corresponds to a decile (every 10% percentile group). The training and validation sets are randomly selected from the 80% smallest graphs with a split ratio of 6:2, while the test set is the 20% largest graphs. Domains constructed to simulate the scenario of training on propagation graphs filtered to a specific range of sizes but then deploying

Subgraph compositions: SBM-ISOLATION and SBM-ENVIRONMENT are two artificial graph datasets generated with Stochastic Block Models (SBMs) (Holland et al., 1983). A SBM defines a random graph which modulates the intra- and extra-community connection probabilities. In SBM-ISOLATION, we randomly select 3 from the 5 total communities $C_1, \cdots, C_5$ for each graph, with intra-edge probability $p_1 = 0.5, \cdots, p_5 = 0.9$ respectively, and we predict the number of communities with relatively smaller extra-edge probability $q' = 0.1$ (with default $q = 0.3$). The domain is characterized by the composition of communities selected, for example, $\{C_1, C_3, C_5\}$. While in SBM-ENVIRONMENT, we define 7 communities $\{C_1, \cdots, C_7\}$ with intra-edge probabilities $p_1 = 0.3, p_2 = 0.4, \cdots, p_7 = 0.9$ and set the extra-edge probability q to 0.1. For each graph, we select 3 communities from $\{C_1, C_3, C_5, C_7\}$ and 2 communities from $\{C_2, C_4, C_6\}$, and we use the former selections to define the domain and the later selections as the prediction task. In one sentence, SBM-ISOLATION and SBM-ENVIRONMENT are similar in their distribution shifts: graphs in different domains consist of distinct community subgraphs, although their prediction tasks differ. When generalizing to unseen test graphs, we have already seen all communities, but we have never seen them appear in the same graph.

Structural distortion: SUPERPIXEL-ROTATEDMNIST is a collection of semi-artificial graphs constructed from images in Rotated MNIST (Ghifary et al., 2015) using the super-pixel (Achanta et al., 2012) pipeline. In Rotated MNIST, handwritten-digit images are split into 6 equal folds and rotated counter-clockwisely by $0°, 15°, 30°, 45°, 60°$, and $75°$, respectively. Because super-pixels sampled near image boundaries and on digits are not significantly affected by rotations, the super-pixel graphs converted from images rotated with different angles can be thought of as graphs that undergo some "structure distortion"; see Fig. 1 for visualizations. The task is to classify digits with 1-dimensional node features as the super-pixel intensities. We train and validate on graphs converted from $0°, \cdots, 60°$-rotated images and generalize to unseen $75°$ ones.

Feature shift only: SUPERPIXEL-COLOREDMNIST is another semi-artificial graph dataset constructed using Colored MNIST images (Arjovsky et al., 2019) using the same super-pixel method as above. In Colored MNIST, the domain affects the true image-to-label correlations, fooling the algorithms that try to learn this mapping directly. When converted to graphs, the super-pixel colors are used as node features, and GNNs which only use the node features will fail on the test graphs, where the feature-to-label correlation is reversed. However, GNNs that only learn the structure-to-label mapping will generalize as if the test graphs are in-distribution. Domain shifts on SUPERPIXEL-COLOREDMNIST only affect node features, making it a feature-shift-only dataset.

In-distribution vs. out-of distribution validation set. An important fact we want to note is that, among the eight datasets we curated, MOLHIV, MOLPCBA, and PPA's validation sets are out-of-distribution (i.e., not part of the training domains), while the rest five datasets' validation set are just randomly sampled from the training domains. We understand that both setups are acceptable. Sampling the validation set from the training domains is just the standard model selection method discussed in (Gulrajani & Lopez-Paz, 2020), while using an out-of-distribution validation set can sometimes improve the OOD test performance as shown in (Koh et al., 2021). We did a simple ablation study on MOLHIVand found that using an in-distribution validation set, the OOD test performance of empirical risk minimization with a Graph Isomorphism Network (GIN) is 0.767, a bit higher than the baseline model 0.752 selected using the original out-of-distribution validation set.

Data analytics and visualization. To gain a qualitative understanding of the distribution shifts defined for some of the datasets above, we visualize the domain distributions via scatter plots of embeddings learned with a Weisfeiler-Lehman (WL) subtree kernel of 3 iterations; see Appendix D for details. We see form Section 3, the graphs (dots) from different domains (represented by colors) in MOLHIV, MOLPCBA are clearly separated. However, on PPA and GOSSIPCOP, there are still large overlaps between the distributions of graphs from different domains.

4 BASELINE ALGORITHMS AND MODELS

We implement a collection of OOD generalization algorithms, graph augmentation methods, and GNN models and techniques as shown in Table 1, see Appendix C.

OOD generalization algorithms. With Empirical Risk Minimization (ERM, (Vapnik, 2013)) (i.e., no transfer) as the baseline, we implement the following six[3] OOD generalization algorithms

- Group Distributionally Robust Optimization (**GroupDRO**, Sagawa et al. (2019)) minimizes the empirical risk on the training set while more heavily weighting domains with larger errors.
- Invariant Risk Minimization (**IRM**, (Arjovsky et al., 2019)) seeks to learn a common parameterization of the linear classifier ω that minimizes the empirical risks in every domain.
- Correlation Alignment for Deep Domain Adaptation (**DeepCORAL**, (Sun & Saenko, 2016)) matches the mean and covariance statistics of graph representations $h_G = \phi(G)$ (after graph pooling) from different domains.
- Domain-Adversarial Neural Networks: (**DANN**, (Ganin et al., 2016)) use an adversarial network to match the distributions of graph representations. As mentioned in Section 2, we also consider a modified version, **DANN-Graph**, where the featurizer ϕ is a permutation-equivariant GNN without graph pooling, and the classifier ω is a permutation-invariant GNN with graph pooling. We investigate whether performing distribution matching on the graph embeddings before pooling leads to an increase in performance.
- Meta-Learning for Domain Generalization (**MLDG**, (Li et al., 2018)) learns how to generalize across domains using the framework of MAML (Finn et al., 2017).

Augmentation methods. We implement two augmentation methods tailored for graphs:

- Adversarial Augmentation on Graphs (**FLAG**, (Kong et al., 2020)) iteratively augments node features with gradient-based adversarial perturbations.
- Structural Augmentation (**SA**) performs randomized node dropping and edge perturbation to increase the structural diversity of the graphs sampled from the training domains.

GNN models and techniques. We implement eight GNN models and techniques reflecting a variety of architectural features and inductive biases:

- Standard GNNs: Graph Isomorphism Network (**GIN**, (Xu et al., 2018)) and a 10-layer version **GIN-Deep**, along with Multi-Layer Perceptron (**MLP**) as the ablation setup without structure learning.
- Global-Context Learning: **Virtual Node** (Gilmer et al., 2017; Li et al., 2017) introduces a "virtual node" that is connected to all the nodes in the graph, which helps to learn correlations at a distance and improves the complexity of graph aggregations.
- Spectral Methods: **ChebNet** (Defferrard et al., 2016) applies spectral filtering in the Fourier domain representations of node features.
- Weisfeiler-Lehman (WL) Hierarchy: **3WL-GNN** (Maron et al., 2019) enjoys guaranteed 3-WL expressiveness, which is strictly stronger than message passing GNNs.
- Structure-Aware GNNs: **GSN** (Bouritsas et al., 2020) counts graph substructures isomorphic to some small query graph and enhances the expressive power of message passing GNNs.

5 Experiments

We run experiments for all OOD generalization and graph augmentation algorithms, combined with all GNN models and techniques (see Section 4). For the hyper-parameter search and setups as well as extended experimental details, see Appendix D.

OOD generalization performance of GNN models and techniques. Before evaluating generalization and augmentation algorithms, we want to understand the OOD generalization performance of off-the-shelf GNN models and techniques. However, if a specific model outperforms others in terms of OOD generalization performance, this could be the result of superior expressive power rather than actual robustness to the presented domain shifts. In order to disentangle these two effects, we perform a thorough ablation study. We merge the training, validation, and test sets of a dataset, randomly shuffle and then re-split using exactly the same ratios. Because training and test domains are

[3]We note the recent development of graph-specific algorithms such as those by Yehudai et al. (2021) (generalizing across sizes using a self-supervised learning task) and Wu et al. (2021) (a mix-up framework tailored for class-imbalanced node classification). We do not implement them in our initial release of GDS, because they are recent papers, and the source code has not been made public.

Table 3: Out-of-distribution generalization performance of GNN models/techniques on GDS datasets.

Types		Baseline		Global	Spectral	WL Isomorphism	Structure
Models	GIN	MLP	GIN-Deep	Virtual Node	ChebNet	3WL-GNN	GSN
Algorithm				ERM			
MolPCBA	0.247±0.002	0.090±0.000	0.253±0.002	**0.285±0.001**	0.232±0.001	OOT	0.233±0.001
MolHIV	0.752±0.016	0.684±0.010	0.758±0.016	0.762±0.007	0.755±0.020	0.720±0.021	**0.778±0.013**
PPA	0.687±0.010	0.095±0.000	0.677±0.002	**0.710±0.004**	0.603±0.007	OOT	OOT
GossipCop	0.626±0.008	0.497±0.010	0.624±0.004	0.499±0.078	**0.831±0.004**	0.784±0.031	0.614±0.011
Isolation	0.620±0.006	0.256±0.003	0.686±0.002	0.690±0.007	0.708±0.004	**0.744±0.009**	OOT
Environment	0.769±0.003	0.335±0.000	0.849±0.012	**0.861±0.006**	0.786±0.008	0.629±0.015	OOT
RotatedMNIST	0.766±0.003	0.333±0.007	0.852±0.003	0.799±0.003	**0.854±0.001**	0.602±0.039	0.780±0.005
ColoredMNIST	0.128±0.001	0.103±0.001	0.129±0.000	0.128±0.000	**0.135±0.000**	0.102±0.000	OOT

Table 4: In-distribution generalization performance of GNN models/techniques on the GDS datasets (where the training, validation, and test samples are re-shuffled).

Types		Baseline		Global	Spectral	WL Isomorphism	Structure
Models	GIN	MLP	GIN-Deep	Virtual Node	ChebNet	3WL-GNN	GSN
Algorithm				ERM			
MolPCBA	0.338±0.004	0.094±0.002	0.366±0.001	**0.386±0.001**	0.283±0.003	OOT	0.313±0.002
MolHIV	0.787±0.012	0.694±0.002	0.767±0.005	0.795±0.004	**0.805±0.008**	0.769±0.000	0.788±0.004
PPA	0.923±0.007	0.101±0.000	0.912±0.006	**0.929±0.002**	Diverged	OOT	OOT
GossipCop	0.842±0.001	0.493±0.007	0.835±0.001	0.847±0.002	0.882±0.003	**0.893±0.001**	0.838±0.003
Isolation	0.648±0.002	0.253±0.002	0.706±0.000	0.723±0.005	0.724±0.001	**0.741±0.001**	OOT
Environment	0.789±0.001	0.335±0.001	0.878±0.001	**0.896±0.004**	0.810±0.002	OOT	OOT
RotatedMNIST	0.798±0.003	0.315±0.007	0.865±0.001	0.831±0.001	**0.876±0.001**	0.614±0.002	0.815±0.002
ColoredMNIST	0.688±0.002	0.613±0.001	0.697±0.000	0.693±0.002	**0.715±0.000**	0.609±0.004	OOT

mixed in this process, the performance of this re-shuffled dataset quantifies in-distribution (ID) generalization. Generally, ID generalization performance should be higher than OOD performance. The gap between the ID and OOD generalization performance will then be an indicator of the model's robustness to the domain shift. Table 3 summarizes OOD generalization results, and Table 4 shows the corresponding ID generalization performance. We identify a series of takeaways based on the results of these experiments:

- Virtual Node, ChebNet, 3WL-GNN, and GSN outperform baselines (GIN, MLP, and GIN-Deep) in-distribution and out-of-distribution. Virtual Node excels on MOLPCBA, PPA, ENVIRONMENT, when long-range correlation is critical or graphs are large (see Table 2).
- GSN shows robustness to the domain shifts in MOLHIV: it is not the most performant in-distribution but beats other models on the OOD generalization task. Similarly, ChebNet is robust to the shifts in graph sizes on GOSSIPCOP. Considering scatter plots of the GOSSIPCOP domain distributions like in Section 3, but altered where the embeddings are extracted from the graph Laplacian spectrum instead of the WL sub-tree kernel, the comparison between Fig. 3 and Fig. 2d, we see that domain shifts involving graph size may be easier learned in the Fourier representation than the WL kernel space.
- Finally, Virtual Node fails on GOSSIPCOP where it must generalize to test domains consisting of larger graphs. The reported performance 0.499 ± 0.078 AUC is no better than random guessing (0.5 AUC); see the learning curves in Fig. 4.
- All GNN models failed to generalize OOD on COLOREDMNIST.

Performance of domain generalization and augmentation algorithms. We then evaluate the six domain generalization algorithms and the two graph augmentation methods on the eight GDS datasets; see Table 5. Here, we discuss:

- Augmentation methods, FLAG, and SA, generally work much better than domain generalization algorithms across the datasets (except for MLDG on GOSSIPCOP). This implies many of the domain generalization algorithms proposed for tensor data are not well-suited to graph data and GNNs out-of-the-box.
- On MOLPCBA, MOLHIV, and PPA, generalization algorithms (GroupDRO, IRM, DeepCORAL, and MLDG) fail consistently compared to the ERM baseline. We think this is partially due to the fact that MOLPCBA, MOLHIV, and PPA have relatively large domain counts, and this imposes a distinct challenge for these algorithms; see Section 3 and Appendix D.

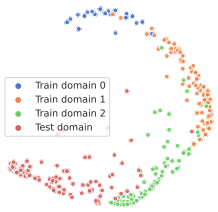


Figure 3: UPFD-GOSSIPCOP clustered using embeddings learned from Laplacian spectrums.

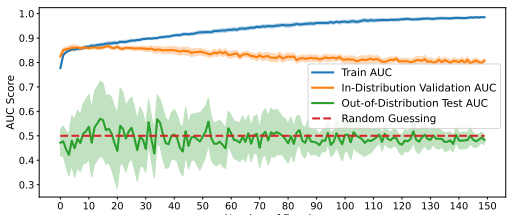


Figure 4: Virtual Node fails catastrophically to generalize OOD on UPFD-GOSSIPCOP.

Table 5: Performance of domain generalization and graph augmentation algorithms on the GDS datasets with GIN backbone models.

Type	Baseline	DRO	IRM	Distribution-Matching			Meta-Learning	Augmentation	
Algorithms	ERM	GroupDRO	IRM	DeepCORAL	DANN	DANN-G	MLDG	FLAG	SA
Model	GIN								
MolPCBA	0.247±0.002	0.211±0.004	0.145±0.002	0.152±0.002	NA	NA	OOT	**0.251±0.004**	0.241±0.001
MolHIV	0.752±0.016	0.735±0.006	0.719±0.013	0.707±0.027	NA	NA	0.650±0.014	0.752±0.003	**0.758±0.005**
PPA	0.687±0.010	0.651±0.011	0.665±0014	0.681±0.016	NA	NA	OOT	**0.712±0.004**	0.702±0.009
GossipCop	0.626±0.008	0.633±0.006	0.636±0.005	0.636±0.012	0.632±0.002	0.633±0.004	**0.643±0.002**	0.635±0.006	0.623±0.005
Isolation	0.620±0.006	0.602±0.002	0.627±0.005	0.621±0.007	0.620±0.005	0.616±0.002	0.626±0.002	**0.633±0.007**	0.621±0.007
Environment	0.769±0.003	0.759±0.001	0.765±0.003	0.763±0.006	0.759±0.003	0.764±0.003	0.757±0.007	**0.778±0.006**	0.776±0.003
RotatedMNIST	0.766±0.003	0.761±0.004	0.771±0.001	0.756±0.002	0.764±0.002	0.765±0.005	0.773±0.005	0.703±0.001	**0.789±0.003**
ColoredMNIST	0.128±0.001	0.126±0.000	0.113±0.002	0.128±0.001	0.127±0.000	0.128±0.000	0.127±0.000	0.125±0.005	**0.129±0.000**

- On GOSSIPCOP, ISOLATION, ENVIRONMENT, ROTATEDMNIST, and COLOREDMNIST, the performance of all domain generalization algorithms are relatively similar to each other and to the ERM baselines. We hypothesize that replacing the GIN backbone with more powerful models might illicit larger performance differences.
- IRM does not outperform the other algorithms on our graph SUPERPIXEL-COLOREDMNIST dataset because of the model selection criteria. We perform model selection using an in-distribution validation set on COLOREDMNIST; see Section 3. The selected model usually fails to generalize to the OOD domain on COLOREDMNIST, because on COLOREDMNIST, higher ID performance usually implies a lower OOD result. The maximum OOD test performance of IRM achieved during training was 0.718 ± 0.102, which is significantly higher than the other algorithms'.

Combinations of domain generalization or augmentation algorithms with the best performing GNN models. We expect that combinations of domain generalization algorithms and powerful GNN models can further improve the performance because (1) only domain generalization algorithms know how to make use of domain labels explicitly; and (2) sufficiently expressive featurizers are necessary for any OOD generalization algorithm to work. We then evaluate the OOD generalization performance of algorithms combined with the best performing models[4] selected from Table 3 for each dataset; see Table 6. We also test these combinations on the "re-shuffled" datasets as above to measure the performance of these combinations when tested in-distribution; see Table 7. The main results from these combination experiments are:

- The differences in performance between domain generalization algorithms in Table 6 are slightly larger than those in Table 5. And by comparing to Table 5, we see DANN-G outperforms MLDG on GOSSIPCOP, and MLDG outperforms SA on ROTATEDMNIST when using ChebNet as the backbone model. However, overall most domain generalization algorithms still perform similarly to the corresponding ERM baselines.
- If we compare the three distribution-matching generalization algorithms: deepCORAL, DANN, and DANN-Graph, their performance rarely differs by a significant margin. This means that the matching of graph embeddings (without pooling) that DANN-Graph performs does not make a difference. Theoretically, DANN and DANN-Graph could be more powerful than deepCORAL when paired with a powerful discriminator, but the adversarial optimization problem they pose is more difficult to solve in practice.

[4]We do not consider 3WL-GNN and GSN for this selection because they are not compatible with all the algorithms; see Appendix D.

Table 6: Performance of domain generalization and graph augmentation algorithms with the best performing GNN models and techniques on each of the GDS dataset.

Types		Baseline	DRO	IRM	Distribution-Matching			Meta-Learning	Augmentation	
Algorithms	Models	ERM	GroupDRO	IRM	DeepCORAL	DANN	DANN-G	MLDG	FLAG	SA
MolPCBA	Virtual Node	0.285±0.001	0.238±0.001	0.155±0.001	0.161±0.001	NA	NA	OOT	**0.293±0.003**	0.268±0.003
MolHIV	Virtual Node	0.762±0.007	0.729±0.014	0.715±0.005	0.722±0.025	NA	NA	0.693±0.019	0.769±0.017	**0.777±0.022**
PPA	Virtual Node	0.710±0.004	0.692±0.004	0.609±0.004	0.492±0.008	NA	NA	OOT	**0.724±0.002**	0.717±0.001
GossipCop	ChebNet	0.831±0.004	0.829±0.005	0.833±0.005	0.834±0.003	0.837±0.001	**0.838±0.002**	0.826±0.003	0.834±0.004	0.817±0.003
Isolation	ChebNet	0.708±0.004	0.704±0.007	0.713±0.004	0.713±0.006	0.704±0.007	0.704±0.007	0.716±0.005	0.718±0.003	**0.720±0.006**
Environment	Virtual Node	0.861±0.006	0.851±0.002	0.846±0.004	0.862±0.001	0.843±0.006	0.848±0.009	0.798±0.001	**0.887±0.003**	0.861±0.012
RotatedMNIST	ChebNet	0.854±0.001	0.851±0.001	0.857±0.002	0.855±0.002	0.852±0.001	0.852±0.001	**0.864±0.001**	0.854±0.002	0.855±0.001
ColoredMNIST	ChebNet	**0.135±0.000**	0.133±0.001	0.123±0.004	**0.135±0.000**	0.134±0.000	0.134±0.000	0.130±0.000	**0.135±0.001**	0.134±0.000

Table 7: In-distribution and out-of-distribution performance gaps of best combinations of algorithms/models.

Datasets	MolPCBA	MolHIV	PPA	GossipCop	Isolation	Environment	RotatedMNIST	ColoredMNIST
Models	Virtual Node	Virtual Node	Virtual Node	ChebNet	ChebNet	Virtual Node	ChebNet	ChebNet
Algorithms	FLAG	GCL	FLAG	DANN-G	GCL	FLAG	MLDG	FLAG
In-distribution generalization performance	0.384±0.001	0.814±0.005	0.936±0.001	0.885±0.000	0.726±0.007	0.909±0.001	0.874±0.03	0.706±0.000
Performance gaps (ID − OOD)	0.091±0.003	0.037±0.023	0.212±0.003	0.047±0.002	0.006±0.009	0.022±0.003	0.010±0.003	0.571±0.001

Observations. We summarize our observations from Tables 3 to 7 as follows:

- Most of the domain generalization algorithms fail when applied to graph distribution shift datasets; as shown in Tables 5 and 6. Their performance is usually close to the corresponding ERM baseline and can be even lower on the OGB-MOLHIV, OGB-MOLPCBA, and OGB-PPA datasets.
- We conclude that the best performance on a dataset is usually achieved by the best performing GNN model with an augmentation technique; see Table 6. This implies that graph augmentations are promising research directions to improve OOD generalization on graphs further.
- Lastly, in Table 7, we see the gaps between in-distribution and out-of-distribution generalization performance of the best-performing pairs varies across datasets. The performance gaps on ISOLATION and ROTATEDMNIST are relatively small, indicating that those best-performing pairs we identified are robust to the domain shifts considered. However, most of the gaps are large, especially on OGB-PPA, OGB-MOLPCBA, and SUPERPIXEL-COLOREDMNIST, addressing that there is still large room to improve the OOD generalization on those datasets by designing tailored algorithms to capture their specific statistical invariances.

6 CONCLUSIONS AND OUTLOOK

In this paper, we take a closer look at the out-of-domain generalization problem on graphs by first curating eight datasets reflecting diverse types of distribution shifts on graphs and then conduct extensive empirical evaluations of popular domain generalization algorithms, graph augmentation methods and GNN models/techniques on the benchmarks. Our results lead to two major conclusions: (1) Most domain generalization algorithms fail to improve the OOD generalization performance when applied to the diverse types of domain shifts on graphs; (2) The optimal combinations of powerful GNN models with strong graph augmentation techniques can achieve the state-of-the-art performance in OOD generalization on graphs. Our research address the need for either OOD generalization algorithms tailored for graphs or further graph augmentation techniques that make the predictor robust to more types of spurious correlations.

ETHICS STATEMENT

In the course of our evaluation, we did not produce any potentially harmful insights, nor become involved with any potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues. However, our work highlights the necessity for graph-specific domain generalization methods, as existing techniques are insufficient. We urge readers to keep in mind that even many of the highest performing combinations in our experiments exhibit a large gap between in-domain and out-of-domain generalization, and thus, practitioners should consider the risks associated with their applications in domain shifted settings. Moreover, graph data is heterogeneous, and rankings on our benchmarks may not reflect robustness on other problems.

REPRODUCIBILITY STATEMENT

To ensure reproducibility of our results, we ran each experiment 4 times with different random seeds and report results by mean and standard deviation. We direct readers to Appendix D for an extensive description of our hyper-parameter tuning procedures.

REFERENCES

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

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

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

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *arXiv preprint arXiv:2006.09252*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. User preference-aware fake news detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135. PMLR, 2017.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pp. 2551–2559, 2015.

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*, pp. 1263–1272. PMLR, 2017.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2020.

Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, et al. Claimbuster: The first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment*, 10 (12):1945–1948, 2017.

Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

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. *arXiv preprint arXiv:2005.00687*, 2020.

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

Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. Flag: Adversarial data augmentation for graph neural networks. *arXiv preprint arXiv:2010.09891*, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Junying Li, Deng Cai, and Xiaofei He. Learning graph-level representation for drug discovery. *arXiv preprint arXiv:1709.03741*, 2017.

Ricardo Macarron, Martyn N Banks, Dejan Bojanic, David J Burns, Dragan A Cirovic, Tina Garyantes, Darren VS Green, Robert P Hertzberg, William P Janzen, Jeff W Paslay, et al. Impact of high-throughput screening in biomedical research. *Nature reviews Drug discovery*, 10(3): 188–195, 2011.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *arXiv preprint arXiv:1905.11136*, 2019.

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.

Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (3):353–362, 1983.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big data*, 8(3):171–188, 2020.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Damian Szklarczyk, Annika L Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T Doncheva, John H Morris, Peer Bork, et al. String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic acids research*, 47(D1):D607–D613, 2019.

Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pp. 1521–1528. IEEE, 2011.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Lirong Wu, Haitao Lin, Zhangyang Gao, Cheng Tan, Stan Li, et al. Graphmixup: Improving class-imbalanced node classification on graphs by self-supervised context prediction. *arXiv preprint arXiv:2106.11133*, 2021.

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.

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

Gilad Yehudai, Ethan Fetaya, Eli Meirom, Gal Chechik, and Haggai Maron. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*, pp. 11975–11986. PMLR, 2021.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

Supplementary Material

A Details on the OOD Generalization Problem

In the supervised learning problem of graph classification, the training dataset $S = \{(G_i, y_i)\}_{i=1}^n$ contains i.i.d. samples from the joint probability distribution $P(G, y)$. We choose a predictor $f = \omega \circ \phi$ that minimizes the empirical risk $\frac{1}{n}\sum_{i=1}^n \ell(f(G_i), y_i)$ (Vapnik, 2013), where $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ is the loss function. The ubiquitous choice of hypothesis class for learning graph representations are Graph Neural Networks (GNNs), which apply permutation-equivariant operations, e.g. message passing between connected nodes, and a permutation-invariant graph-level pooling function to all nodes representations to obtain h_G.

In the OOD generalization setting, the highly dimensional and discrete nature of graph structure introduces added complexity as compared to structures like 2D grids of continuous pixel values used to represent images. If we consider connected undirected graphs without loops of size k, there are $O(2^{\binom{k}{2}})$ different representations of adjacency matrices $A \in \{0, 1\}^{\binom{n}{2}}$, while there are only $O(2^{\binom{k}{2}}/k!)$ distinct graphs. The fact that a graph structure can be encoded differently implies that the functions acting on graphs should satisfy the permutation invariance given by the permutation group $\mathfrak{G} = \Sigma_k$, i.e., the outcomes of a function on any two isomorphic graphs should be identical.

If not mentioned otherwise in this work, we assumed the featurizer ϕ was a permutation-invariant function (i.e., GNN with graph pooling), and the classifier ω a Multi-Layer Perceptron (MLP). However, in other cases, the featurizer ϕ is permutation-equivariant and the classifier ω is permutation-invariant, where embeddings $\phi(G)$ are also attribute graphs. We denote the combination of this formulation with Domain-Adversarial Neural Networks (DANN) (Ganin et al., 2016) in Sections 4 and 5 as DANN-Graph.

B Details on the GDS Datasets

Size growth: UPFD-GossipCop On the original dataset, we find a multi-layer perceptron (MLP) can achieve 0.948 accuracy which is even higher than the accuracy of a Graph Isomorphism Network (GIN) with the same number of layer and parameters, 0.930. Thus, we replace all node features (which are user profile information) prepared in Dou et al. (2021) with random integers uniformly selected from a vocabulary of size 8, i.e., $\{0, \cdots, 7\}$. We then use the ROC-AUC metric to reflect false-positives.

C Details on Algorithms and Models

C.1 Details on OOD Generalization Algorithms

C.1.1 Risk Minimization - ERM,GroupDRO,IRM,MLDG

- **ERM:** In the domain generalization setting, empirical risk minimization means simultaneously minimizing the loss over all D_{train} datasets S^d combined such that the objective becomes $\frac{1}{n_d \cdot D_{\text{train}}}\sum_{d,i}^{n_d, D_{\text{train}}} \ell(f((G_i^d), y_i^d))$ (Vapnik, 2013).
- **GroupDRO:** Let $\mathcal{R}_d = \frac{1}{n_d}\sum_i^{n_d} \ell(f((G_i^d), y_i^d))$ be the empirical risk over a single domain. Group Distributionally Robust Optimization (GroupDRO) for domain generalization then solves $\min\{\max_{d \in D_{\text{train}}} \mathcal{R}_d\}$, minimizing the worst case expected risk for all domains (Sagawa et al., 2019).
- **IRM:** Invariant Risk Minimization (IRM) (Arjovsky et al., 2019) seeks to learn a featurizer $\omega : \mathcal{G} \to \mathcal{H}$ and classifier $\phi : \mathcal{H} \to \mathcal{Y}$, composed as predictor $\phi \circ \omega$, such that a single parametrization for ϕ minimizes the empirical loss in each domain, $\mathcal{R}_d(\phi \circ \omega)$ for all domains, $\phi \in \arg\min_\Phi \mathcal{R}_d(\phi \circ \omega)$ for all $d \in D_{\text{train}}$
- **MLDG**: Meta-learning for Domain Generalization splits the D_{train} domains datasets S^d into subsets *meta-train* and *meta-test*. During training, the loss is first computed over the

meta-train sets and a parameter update determined, and then the loss is computed over the *meta-test* subsets with respect to model parameters after *meta-train* update is applied. This bi-level optimization is performed using gradient descent, and the final parameters are evaluated on the true test domains.

C.1.2 DISTRIBUTION MATCHING - DEEPCORAL,DANN

- **DeepCORAL:** which penalizes the difference between each domain. Let C_i and C_j denote the feature covariance matrices of domain i and j, that is $C_i = \frac{1}{n_i - 1}(D_i^\top D_i - \frac{1}{n_i}(\mathbf{1}^\top D_i)^\top(\mathbf{1}^\top D_i))$, where n_i is the number of samples and D_i indicates the data example. The CORAL loss is defined as $l = \frac{1}{4d^2}\|C_i - C_j\|_F^2$, d is the number of dimension of data and F is Frobenius norm.
- **DANN:** which trains on labeled data from the source domain and unlabeled data from the target domain. During the training procedure, DANN encourages the appearance of features, which are discriminative for the main learning task on the source domains and indiscriminate with respect to the transfer between the domains.
- **DANN-GRAPH:** which differs from DANN as it use the graph before pooling as the embedding. In DANN-GRAPH, the featurizer is a permutation-equivalent function on graph, while the classifier is then a permutation-invariant function, typically a GNN with graph pooling.

C.2 DETAILS ON GRAPH AUGMENTATION ALGORITHMS

C.2.1 FLAG

With image data, rotations and crops are standard examples of semantics preserving, transformations, and these types of augmentations are not considered to be domain specific. However, the graphs we consider are abstracted from diverse fields and individual nodes, edges and features are much more semantically relevant than individual pixels in an image - i.e. nodes might represent atoms and edges represent chemical bonds. Thus, it is unclear what types of augmentation (in features and connective structure) work well for which types of graphs and the domain shifts they are subject to. FLAG is one type of graph adversarial-based augmentation algorithms which happen in the node feature space. Our experimental results show that FLAG is effective especially when the input node features are discrete categorical features. In this work we follow the original paper to do 3 steps of adversarial training.

C.2.2 ERM+SA VERSUS CONTRASTIVE PRETRAINING

You et al. (2020) proposes a graph contrastive learning framework (GCL) for self-supervised pretraining of GNNs based on minimizing the distance in latent space between two augmented views of the same graph according to a contrastive loss (a formulation of the loss proposed in SimCLR (Chen et al., 2020), but for graph data). The self-supervised pretraining plus supervised finetuning scheme they implement for solving transfer learning tasks was the closest application to the domain generalization present in their evaluation. Inspired by the graph contrastive learning framework (GCL) for self-supervised pre-training of GNNs proposed by You et al. (2020), we select two of the *structural augmentation* (SA) types that they use to generate contrastive pairs and simply apply them directly the training data while performing ERM. We consider *Node dropping* - randomly discarding a certain portion of vertices V along with their connections, and *Edge perturbation* - perturbing the structure of G by randomly adding or dropping a certain ratio of edges, under the assumption that the graph semantics are preserved under these transformations and that a model should be robust to variance in the edge connectivity. The choice described in section Section 4 to omit this contrastive pretraining in the main evaluation is motivated by empirical results suggesting that ERM+SA alone is competitive, and that GCL does not provide a performance benefit in the domain generalization setting compared to the added computational cost of processing $2N$ augmented graphs per epoch of pretraining.

Table 8: The results for IRM and Meta-Learning type algorithms with different hyperparameters. Top results are boldfaced.

	IRM		Meta-Learning		
Dataset	IRM-1.0	IRM-100.0	MLDG-0.1	MLDG-1.0	MLDG-10.0
GIN					
MolPCBA	**0.112**	0.066	NA	NA	NA
MolHIV	**0.689**	0.643	**0.666**	0.615	0.653
PPA	**0.581**	0.443	NA	NA	NA
GossipCop	**0.642**	0.599	**0.64**	0.639	0.638
Isolation	**0.616**	0.547	0.604	**0.638**	0.625
Environment	**0.757**	0.599	0.759	0.751	**0.767**
RotatedMNIST	**0.766**	0.377	**0.779**	0.761	0.768
ColoredMNIST	**0.126**	0.106	**0.128**	0.127	0.128

D MORE EXPERIMENTAL DETAILS

We consider the following hyperparameters for different algorithms:

- IRM: the weight for IRM penalty loss is chosen from $\{1.0, 100.0\}$, IRM penalty anneals per 500 iterations.
- MLDG: the hyperparamter beta is chosen from $\{0, 1, 1.0, 10.0\}$.
- FLAG: the inner gradient ascent step size for FLAG is chosen from $\{0, 001, 0.001\}$.
- GCL: the data augmentation ratio for GCL is chosen from $\{0, 1, 0.2, 0.3, 0.4\}$, the type of data augmentations are randomly chosen from node drop and edge permutation.
- DeepCORAL: Coral penalty weight is chosen from $\{1, 0, 10.0\}$.
- DANN: the DANN lambda value is chosen from $\{0, 1, 1.0, 10.0\}$.
- DANN-G: similarly the DANN lambda is chosen from $\{0, 1, 1.0, 10.0\}$.
- GSN: the type of subgraph is cycle and the maximum of substructure size is 6.

Table (8), (9) and (10) present the results on the validation set for different types of algorithms across different hyperparameters. We select the best hyperparameter from Table (8), (9) and (10) and summary all the important hyperparameters in Table (11).

For all the implementations, we use the Adam optimizer, with weight decay 0. We also show the batch size, the number of epochs, learning rate, the number of groups for each batch for every dataset in Table (12). We use GNN base model GIN with five convolutional layers, dropout of 0, and the RELU activation function. The dimension of hidden layer is 300, number of training epochs depends on different datasets.

Table 9: The results for Data Augmentation type algorithms with different hyperparameters. Top results are boldfaced.

	Augmentation					
Algorithms	FLAG-0.001	FLAG-0.01	GCL-0.1	GCL-0.2	GCL-0.3	GCL-0.4
	GIN					
MolPCBA	0.251	**0.257**	**0.244**	0.232	0.227	0.227
MolHIV	**0.76**	0.747	0.732	0.78	**0.786**	0.766
PPA	**0.699**	0.695	**0.704**	0.699	0.675	0.688
GossipCop	0.63	**0.632**	**0.62**	0.613	**0.62**	0.614
Isolation	0.616	**0.633**	0.623	0.631	**0.637**	0.616
Environment	0.77	**0.794**	0.772	**0.774**	0.771	0.671
RotatedMNIST	0.695	**0.698**	**0.787**	0.781	0.784	0.782
ColoredMNIST	0.127	**0.126**	**0.129**	0.129	0.129	0.13

Table 10: The results for Distribution Matching type algorithms with different hyperparameters. Top results are boldfaced.

	Distribution-Matching							
Dataset	DeepCORAL-1.0	DeepCORAL-10.0	DANN-0.1	DANN-1.0	DANN-10.0	DANN-G-0.1	DANN-G-1.0	DANN-G-10.0
	GIN							
MolPCBA	**0.152**	0.145	NA	NA	NA	NA	NA	NA
MolHIV	**0.722**	0.674	NA	NA	NA	NA	NA	NA
PPA	**0.704**	0.694	NA	NA	NA	NA	NA	NA
GossipCop	**0.639**	0.627	**0.635**	0.631	0.637	**0.638**	0.631	0.578
Isolation	**0.639**	0.613	0.608	**0.611**	0.602	0.605	**0.606**	0.576
Environment	**0.766**	0.761	0.75	**0.755**	0.754	0.755	**0.759**	0.74
RotatedMNIST	**0.755**	0.759	**0.745**	0.74	0.681	**0.742**	0.729	0.684
ColoredMNIST	**0.128**	0.129	**0.126**	0.125	0.125	**0.127**	0.125	0.125

Table 11: The summary of the hyperparameters for different algorithms.

Datasets	MolPCBA	MolHIV	PPA	GossipCop	Isolation	Environment	RotatedMNIST	ColoredMNIST
IRM	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
MLDG	NA	0.1	NA	0.1	1.0	10.0	0.1	0.1
FLAG	0.01	0.001	0.001	0.01	0.01	0.01	0.01	0.01
GCL	0.1	0.3	0.1	0.1	0.3	0.2	0.1	0.1
DeepCORAL	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
DANN	NA	NA	NA	0.1	1.0	1.0	0.1	0.1
DANN-G	NA	NA	NA	0.1	1.0	1.0	0.1	0.1

Table 12: More hyperparameters for different datasets.

Datasets	MolPCBA	MolHIV	PPA	GossipCop	Isolation	Environment	RotatedMNIST	ColoredMNIST
Batch Size	128	128	32	128	128	128	128	128
# Epochs	250	200	150	150	150	200	150	100
Learning Rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
# Groups Per Batch	4	4	4	4	4	2	4	2