CAUSAL-AWARE GRAPH NEURAL ARCHITECTURE SEARCH UNDER DISTRIBUTION SHIFTS (SUPPLEMENTARY MATERIALS)

Anonymous authors
Paper under double-blind review

A ALGORITHM

The overall framework and optimization procedure of the proposed CARNAS are summarized in Figure 1 and Algorithm 1, respectively.

Algorithm 1 The overall algorithm of **CARNAS**

Require: Training Dataset $\mathcal{G}_{tr}$,
Hyper-parameters t in Eq. (6), μ in Eq. (10), θ_1, θ_2 in Eq. (16)
Initialize all trainable parameters
for $p = 1, \ldots, P$ **do**
  Set σ_p as Eq. (17)
  Derive causal and non-causal subgraphs as Eq. (4) (5) (6)
  Calculate graph representations of causal and non-causal subgraphs as Eq. (7) (8)
  Calculate $\mathcal{L}_{cpred}$ using Eq. (9)
  Sample N_s non-causal subgraphs as candidates
  for causal subgraph G_c of graph G in $\mathcal{G}_{tr}$ **do**
    Do interventions on G_c in latent space as Eq. (10)
    Calculate architecture matrix $\mathbf{A_c}$ and $\{\mathbf{A_v}_j\}$ from causal subgraph and their intervention graphs as Eq. (12)
  end for
  Calculate $\mathcal{L}_{op}$ using Eq. (13)
  Calculate $\mathcal{L}_{pred}$ using Eq. (11) (14)
  Calculate $\mathcal{L}_{arch}$ using Eq. (15)
  Calculate the overall loss $\mathcal{L}_{all}$ using Eq. (16)
  Update parameters using gradient descends
end for

B REPRODUCIBILITY DETAILS

B.1 DEFINITION OF SEARCH SPACE

The number of layers in our model is predetermined before training, and the type of operator for each layer can be selected from our defined operator search space $\mathcal{O}$. We incorporate widely recognized architectures GCN, GAT, GIN, SAGE, GraphConv, and MLP into our search space as candidate operators in our experiments. This allows for the combination of various sub-architectures within a single model, such as using GCN in the first layer and GAT in the second layer. Furthermore, we consistently use standard global mean pooling at the end of the GNN architecture to generate a global embedding.

B.2 DATASETS DETAILS

We utilize synthetic SPMotif datasets, which are characterized by three distinct degrees of distribution shifts, and three different real-world datasets, each with varied components, following previous works (21; 32; 29). Based on the statistics of each dataset as shown in Table 1, we conducted a

comprehensive comparison across *various scales and graph sizes*. This approach has empirically validated the scalability of our model.

Table 1: Statistics for different datasets.

	Graphs	Avg. Nodes	Avg. Edges
ogbg-molhiv	41127	25.5	27.5
ogbg-molsider	1427	33.6	35.4
ogbg-molbace	1513	34.1	36.9
SPMotif-0.7/0.8/0.9	18000	26.1	36.3

Detailed description for real-world datasets The real-world datasets are 3 molecular property prediction datasets in OGB (9), and are adopted from the MoleculeNet (31). Each graph represents a molecule, where nodes are atoms, and edges are chemical bonds.

- The HIV dataset was introduced by the Drug Therapeutics Program (DTP) AIDS Antiviral Screen, which tested the ability to inhibit HIV replication for over 40000 compounds. Screening results were evaluated and placed into 2 categories: inactive (confirmed inactive CI) and active (confirmed active CA and confirmed moderately active CM).
- The Side Effect Resource (SIDER) is a database of marketed drugs and adverse drug reactions (ADR). The version of the SIDER dataset in DeepChem has grouped drug side-effects into 27 system organ classes following MedDRA classifications measured for 1427 approved drugs (following previous usage).
- The BACE dataset provides quantitative (IC_{50}) and qualitative (binary label) binding results for a set of inhibitors of human β-secretase 1 (BACE-1). It merged a collection of 1522 compounds with their 2D structures and binary labels in MoleculeNet, built as a classification task.

The division of the datasets is based on scaffold values, designed to segregate molecules according to their structural frameworks, thus introducing a significant challenge to the prediction of graph properties.

B.3 Detailed Hyper-parameter Settings

We fix the number of latent features $Q = 4$ in Eq. (4), number of intervention candidates N_s as batch size in Eq. (10), $\sigma_{min} = 0.1$, $\sigma_{max} = 0.7$, $P = 100$ in Eq. (17), and the tuned hyper-parameters for each dataset are as in Table 4.

Table 2: Hyper-parameter settings

Dataset	t in Eq. (6)	μ in Eq. (10)	θ_1 in Eq. (16)	θ_2 in Eq. (16)
SPMotif-0.7/0.8/0.9	0.85	0.26	0.36	0.010
ogbg-molhiv	0.46	0.68	0.94	0.007
ogbg-molsider	0.40	0.60	0.85	0.005
ogbg-molbace	0.49	0.54	0.80	0.003

C Deeper Analysis

C.1 Supplementary Analysis of the Experimental Results

Sythetic datasets. We notice that the performance of CARNAS is way better than DIR (29), which also introduces causality in their method, on synthetic datasets. We provide an explanation as follows: Our approach differs from and enhances upon DIR in several key points. Firstly, unlike DIR, which uses normal GNN layers for embedding nodes and edges to derive a causal subgraph, we employ disentangled GNN. This allows for more effective capture of latent features when extracting causal subgraphs. Secondly, while DIR focuses on the causal relationship between a graph instance and its label, our study delves into the causal relationship between a graph instance and its optimal architecture, subsequently using this architecture to predict the label. Additionally, we incorporate NAS method, introducing an invariant architecture customization module, which considers the impact of architecture on performance. Based on these advancements, our method may outperform DIR.

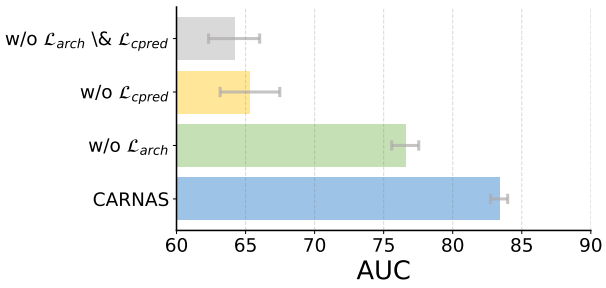

Figure 1: Results of ablation studies on SIDER, where 'w/o $\mathcal{L}_{arch}$' removes $\mathcal{L}_{arch}$ from the overall loss in Eq. (**??**), 'w/o $\mathcal{L}_{cpred}$' removes $\mathcal{L}_{cpred}$, and 'w/o $\mathcal{L}_{arch}$ & $\mathcal{L}_{cpred}$' removes both of them. The error bars report the standard deviations. Besides, the average and standard deviations of the best-performed baseline on each dataset are denoted as the dark and light thick dash lines respectively.

Real-world datasets. We also notice that our methods improves a lot on the performance for the second real-world dataset SIDER. We further conduct an ablation study on SIDER to confirm that each proposed component contributes to its performance, as present in Figure 1. The model 'w/o $Larch$' shows a slight decrease in performance, while 'w/o $Lcpred$' exhibits a substantial decline. This indicates that both restricting the invariance of the influence of the causal subgraph on the architecture via $Larch$, and ensuring that the causal subgraph retains crucial information from the input graph via $Lcpred$, are vital for achieving high performance on SIDER, especially the latter which empirically proves to be exceptionally effective.

C.2 Dynamic Training Process and Convergence

For a deeper understanding of our model training process, and further remark the impact of the dynamic σ_p in Eq.(17), we conduct experiments and compare the training process in the following settings:

- 'with Dynamic σ' means we use the dynamic σ_p in Eq.(17) to adjust the training key point in each epoch.
- 'w/o Dynamic σ' means we fix the σ in Eq.(16) as a constant value $\frac{\sigma_{max}+\sigma_{min}}{2}$.

According to Figure 2, *our method can converge rapidly in 10 epochs*. Figure 2 also obviously reflects that after 10 epochs the validation loss with dynamic σ keeps declining and its *accuracy continuously rising*. However, in the setting without dynamic σ, the validation loss may rise again, and accuracy cannot continue to improve.

These results verify our aim to adopt this σ_p to *elevate the efficiency of model training* in the way of dynamically adjusting the training key point in each epoch by focusing more on the causal-aware part (i.e. identifying suitable causal subgraph and learning vectors of operators) in the early stages and focusing more on the performance of the customized super-network in the later stages. We also empirically confirm that our method is not complex to train.

C.3 Complexity Analysis

In this section, we analyze the complexity of our proposed method in terms of its computational time and the quantity of parameters that require optimization. Let's denote by $|V|$ the number of nodes in a graph, by $|E|$ the number of edges, by $|\mathcal{O}|$ the size of search space, and by d the dimension of hidden representations within a traditional graph neural network (GNN) framework. In our approach, d_0 represents the dimension of the hidden representations within the identification network GNN_0, d_1 represents the dimension of the hidden representations within the shared graph encoder GNN_1, and d_s denotes the dimension within the tailored super-network. Notably, d_0 encapsulates the combined dimension of Q chunks, meaning the dimension per chunk is d_0/Q.

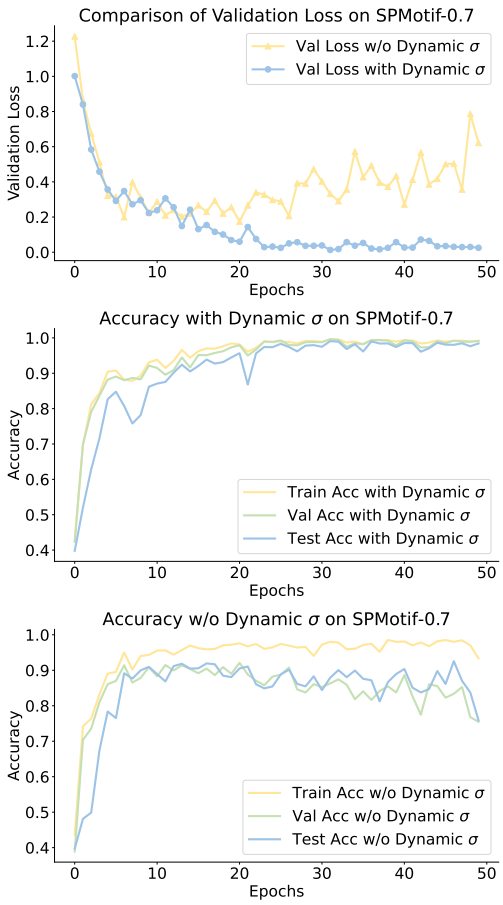

Figure 2: Training process of synthetic datasets.

C.3.1 Time Complexity Analysis

For most message-passing GNNs, the computational time complexity is traditionally $O(|E|d + |V|d^2)$. Following this framework, the GNN_0 in our model exhibits a time complexity of $O(|E|d_0 + |V|d_0^2)$, and the GNN_1 in our model exhibits a time complexity of $O(|E|d_1 + |V|d_1^2)$. The most computationally intensive operation in the invariant architecture customization module, which involves the computation of $\mathcal{L}_{op}$, leads to a time complexity of $O(|\mathcal{O}|^2 d_1)$. The time complexity attributed to the customized super-network is $O(|\mathcal{O}|(|E|d_s + |V|d_s^2))$. Consequently, the aggregate time complexity of our method can be summarized as $O(|E|(d_0 + d_1 + |\mathcal{O}|d_s) + |V|(d_0^2 + d_1^2 + |\mathcal{O}|d_s^2) + |\mathcal{O}|^2 d_1)$.

C.3.2 Parameter Complexity Analysis

A typical message-passing GNN has a parameter complexity of $O(d^2)$. In our architecture, the disentangled causal subgraph identification network GNN_0 possesses $O(d_0^2)$ parameters, the shared GNN encoder GNN_1 possesses $O(d_1^2)$, the invariant architecture customization module contains $O(|\mathcal{O}|d_1)$ parameters and the customized super-network is characterized by $O(|\mathcal{O}|d_s^2)$ parameters. Therefore, the total parameter complexity in our framework is expressed as $O(d_0^2 + d_1^2 + |\mathcal{O}|d_1 + |\mathcal{O}|d_s^2)$.

The analyses underscore that the proposed method scales linearly with the number of nodes and edges in the graph and maintains a constant number of learnable parameters, aligning it with the efficiency of prior GNN and graph NAS methodologies. Moreover, given that $|\mathcal{O}|$ typically represents a modest constant (for example, $|\mathcal{O}| = 6$ in our search space) and that d_0 and d_1 is generally much less than d_s, the computational and parameter complexities are predominantly influenced by d_s. To ensure equitable comparisons with existing GNN baselines, we calibrate d_s within our model such that the parameter count, specifically $|\mathcal{O}|d_s^2$, approximates d^2, thereby achieving a balance between efficiency and performance.

C.4 Training Efficiency

To further illustrate the efficiency of CARNAS, we provide a direct comparison with the best-performed NAS baseline, DCGAS, based on the total runtime for 100 epochs. As shown in Table 3,

CARNAS consistently requires less time across different datasets while achieving superior best performance, demonstrating its enhanced efficiency and effectiveness.

Table 3: Comparison of runtime

Method	SPMotif	HIV	BACE	SIDER
DCGAS	104 min	270 min	12 min	11 min
CARNAS	76 min	220 min	8 min	8 min

C.5 Hyper-parameters Sensitivity

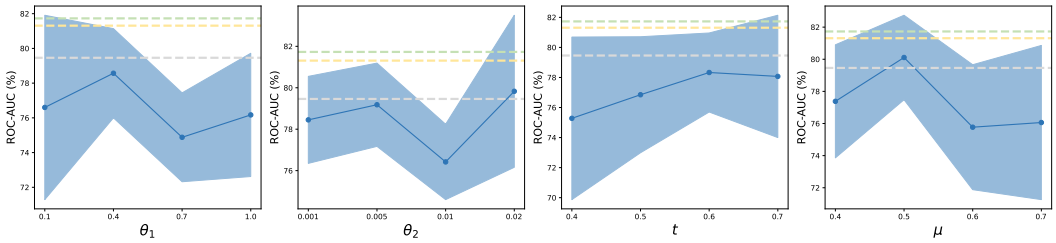

Figure 3: Hyper-parameters sensitivity analysis. The area shows the average ROC-AUC and standard deviations. The green, yellow, grey dashed lines represent the average performance corresponding to the fine-tuned hyper-parameters of CARNAS, best performed baseline DCGAS, 2nd best performed baseline GraphConv, respectively.

We empirically observe that our model is insensitive to most hyper-parameters, which remain fixed throughout our experiments. Consequently, the number of parameters requiring tuning in practice is relatively small. t, μ, θ_1 and θ_2 have shown more sensitivity, prompting us to focus our tuning efforts on these 4 hyper-parameters.

Therefore, we conduct sensitivity analysis (on BACE) for the 4 important hyper-parameters, as shown in Figure 4. The value selection for these parameters were deliberately varied evenly within a defined range to assess sensitivity thoroughly. The specific hyper-parameter settings used for the CARNAS reported in Table 2 (in main paper) are more finely tuned and demonstrate superior performance to the also finely tuned other baselines. The sensitivity *allows for potential performance improvements* through careful parameter tuning, and our results in sensitivity analysis outperform most baseline methods, indicating a degree of *stability and robustness in response to these hyper-parameters*.

Mention that, the best performance of the fine-tuned DCGAS may exceed the performance of our method without fine-tuning sometimes. This is because, DCGAS addresses the challenge of out-of-distribution generalization through data augmentation, generating a sufficient quantity of graphs for training. In contrast, CARNAS focuses on capturing and utilizing causal and stable subparts to guide the architecture search process. The methodological differences and the resulting disparity in the volume of data used could also contribute to the performance variations observed.

Limitation. Although the training time and search efficiency of our method is comparable to most of the Graph NAS methods, we admit that it is less efficient than standard GNNs. At the same time, in order to obtain the best performance for a certain application scenario, our method does need to fine-tune four sensitive hyper-parameters.

D More comparision with OOD GNN

In our initial experiment, we compared our model with two non-NAS-based graph OOD methods, ASAP and DIR. We expanded our evaluation to include 13 well-known non-NAS-based graph OOD methods (covering all the methods you mentioned), providing a comprehensive comparison. The results, presented as below, demonstrate CARNAS not only performs well among NAS-based methods but also significantly outperforms non-NAS graph OOD methods. This superior performance is attributed to CARNAS's ability to effectively discover and leverage the stable causal graph-architecture relationships during the neural architecture search process.

Table 4: Performance Comparison (ROC-AUC) '-' denotes CIGA is not suitable for multi-task dataset

Class	Method	SIDER	BACE	HIV
Vanilla GNN	GCN	59.84 ± 1.54	68.93 ± 6.95	75.99 ± 1.19
	GAT	57.40 ± 2.01	75.34 ± 2.36	76.80 ± 0.58
	GIN	57.57 ± 1.56	73.46 ± 5.24	77.07 ± 1.49
	SAGE	56.36 ± 1.32	74.85 ± 2.74	75.58 ± 1.40
	GraphConv	56.09 ± 1.06	78.87 ± 1.74	74.46 ± 0.86
	MLP	58.16 ± 1.41	71.60 ± 2.30	70.88 ± 0.83
OOD GNN	ASAP	55.77 ± 1.18	71.55 ± 2.74	73.81 ± 1.17
	DIR	57.34 ± 0.36	76.03 ± 2.20	77.05 ± 0.57
	MoleOOD	57.12 ± 0.82	76.65 ± 2.71	76.57 ± 1.11
	CIGA	-	77.53 ± 2.53	76.89 ± 0.85
	iMoLD	60.76 ± 0.65	78.72 ± 1.75	77.17 ± 0.93
	Coral	60.32 ± 1.04	78.65 ± 1.55	76.88 ± 1.75
	DANN	59.52 ± 1.02	78.84 ± 1.11	76.98 ± 1.32
	GIL	59.67 ± 0.32	75.72 ± 1.93	73.70 ± 1.14
	GSAT	60.06 ± 1.11	78.47 ± 1.70	76.70 ± 0.98
	Mixup	60.83 ± 0.74	78.16 ± 2.54	76.81 ± 1.31
	GroupDRO	61.15 ± 1.06	79.24 ± 1.30	76.97 ± 1.36
	IRM	59.50 ± 0.52	78.87 ± 1.50	76.77 ± 1.01
	VREx	54.60 ± 0.91	75.77 ± 3.35	71.60 ± 1.56
NAS	DARTS	60.64 ± 1.37	76.71 ± 1.83	74.04 ± 1.75
	PAS	59.31 ± 1.48	76.59 ± 1.87	71.19 ± 2.28
	GRACES	61.85 ± 2.58	79.46 ± 3.04	77.31 ± 1.00
	DCGAS	63.46 ± 1.42	81.31 ± 1.94	78.04 ± 0.71
	CARNAS	**83.36 ± 0.62**	**81.73 ± 2.92**	**78.33 ± 0.64**

Table 5: Comparison of Time and Memory Cost between OOD GNN and CARNAS

Method	SIDER		BACE		HIV	
	Time (Mins)	Mem. (MiB)	Time	Mem.	Time	Mem.
DIR	5	4328	5	4323	103	4769
MoleOOD	5	4317	5	4315	96	4650
CIGA	-	-	4	4309	86	4510
iMoLD	3	4184	3	4182	65	4377
Coral	3	4323	2	4323	70	4795
DANN	2	4309	2	4314	47	4505
GIL	26	4386	33	4373	412	6225
GSAT	4	4318	4	4310	49	4600
GroupDRO	4	4311	10	4309	50	4509
IRM	4	975	3	978	80	1301
VREx	6	4313	16	4314	51	4516
CARNAS	8	2556	8	2547	220	2672

Regarding time and memory costs, Table below shows that CARNAS is competitive with non-NAS-based graph OOD methods, as we search the architecture and learn its weights simultaneously. The time and memory efficiency of CARNAS make it a practical choice. Thus, we experimentally verify that the proposed CARNAS does make sense, for addressing the graph OOD problem by diving into the NAS process from causal perspective.

E CASE STUDY

For graphs with different motif shapes (causal subparts), we present the learned operation probabilities for each layer (in expectation) in the table below. The values that are notably higher than others for each layer are highlighted in bold, and the most preferred operators for each layer are listed in the last row.

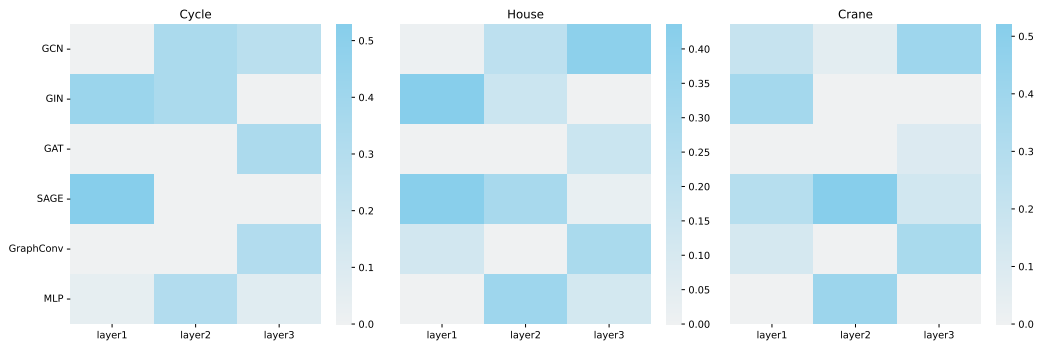

Figure 4: Comparison of operation probabilities for graphs with different motif shapes.

We observe that different motif shapes indeed prefer different architectures, e.g., graphs with cycle prefer GAT in the third layer, while this operator is seldomly chosen in neither layer of the other two types of graphs; the operator distributions are similar for graphs with cycle and house in the first layer, but differ in other layers. To be specific, Motif-Cycle is characterized by a closed-loop structure where each node is connected to two neighbors, displaying both symmetry and periodicity. For graphs with this motif, CARNAS identifies SAGE-GCN-GAT as the most suitable architecture. Motif-House, on the other hand, features a combination of triangular and quadrilateral structures, introducing a certain level of hierarchy and asymmetry. For graphs with this shape, CARNAS determines that GIN-MLP-GCN is the optimal configuration. Lastly, Motif-Crane presents more complex cross-connections between nodes compared to the previous two motifs, and CARNAS optimally configures graphs with it with a GIN-SAGE-GCN architecture.

By effectively integrating various operations and customizing specific architectures for different causal subparts (motifs) with diverse features, our NAS-based CARNAS can further improve the OOD generalization.

F Related Work

F.1 Graph Neural Architecture Search

In the rapidly evolving domain of automatic machine learning, Neural Architecture Search (NAS) represents a groundbreaking shift towards automating the discovery of optimal neural network architectures. This shift is significant, moving away from the traditional approach that heavily relies on manual expertise to craft models. NAS stands out by its capacity to autonomously identify architectures that are finely tuned for specific tasks, demonstrating superior performance over manually engineered counterparts. The exploration of NAS has led to the development of diverse strategies, including reinforcement learning (RL)-based approaches (36; 10), evolutionary algorithms-based techniques (22; 17), and methods that leverage gradient information (16; 33). Among these, graph neural architecture search has garnered considerable attention.

The pioneering work of GraphNAS (6) introduced the use of RL for navigating the search space of graph neural network (GNN) architectures, incorporating successful designs from the GNN literature such as GCN, GAT, etc. This initiative has sparked a wave of research (6; 26; 21; 3; 8; 35; 7), leading to the discovery of innovative and effective architectures. Recent years have seen a broadening of focus within Graph NAS towards tackling graph classification tasks, which are particularly relevant for datasets comprised of graphs, such as those found in protein molecule studies. This research area has been enriched by investigations into graph classification on datasets that are either independently identically distributed (26) or non-independently identically distributed, with GRACES (21) and DCGAS (32) being notable examples of the latter. Through these efforts, the field of NAS continues to expand its impact, offering tailored solutions across a wide range of applications and datasets.

F.2 Graph Out-of-Distribution Generalization

In the realm of machine learning, a pervasive assumption posits the existence of identical distributions between training and testing data. However, real-world scenarios frequently challenge this assumption with inevitable shifts in distribution, presenting significant hurdles to model performance in out-of-distribution (OOD) scenarios (23). The drastic deterioration in performance becomes evident when models lack robust OOD generalization capabilities, a concern particularly pertinent in the domain of Graph Neural Networks (GNNs), which have gained prominence within the graph community (12). Several noteworthy studies (28; 27; 13; 5; 24; 15; 25) have tackled this challenge by focusing on identifying environment-invariant subgraphs to mitigate distribution shifts. These approaches typically rely on pre-defined or dynamically generated environment labels from various training scenarios to discern variant information and facilitate the learning of invariant subgraphs. Moreover, the existing methods usually adopt a fixed GNN encoder in the whole optimization process, neglecting the role of graph architectures in out-of-distribution generalization. In this paper, we focus on automating the design of generalized graph architectures by discovering causal relationships between graphs and architectures, and thus handle distribution shifts on graphs.

F.3 Causal Learning on Graphs

The field of causal learning investigates the intricate connections between variables (20; 19), offering profound insights that have significantly enhanced deep learning methodologies. Leveraging causal relationships, numerous techniques have made remarkable strides across diverse computer vision applications (34; 18). Additionally, recent research has delved into the realm of graphs. For instance, (30) implements interventions on non-causal components to generate representations, facilitating the discovery of underlying graph rationales. (5) decomposes graphs into causal and bias subgraphs, mitigating dataset biases. (14) introduces invariance into self-supervised learning, preserving stable semantic information. (4) ensures out-of-distribution generalization by capturing graph invariance. (11) tackled the challenge of learning causal graphs involving latent variables, which are derived from a mixture of observational and interventional distributions with unknown interventional objectives. To mitigate this issue, the study proposed an approach leveraging a Ψ-Markov property. (1) introduced a randomized algorithm, featuring p-colliders, for recovering the complete causal graph while minimizing intervention costs. Additionally, (2) presented an adaptable method for causality detection, which notably benefits from various types of interventional data and incorporates sophisticated neural architectures such as normalizing flows, operating under continuous constraints. However, these methods adopt a fixed GNN architecture in the optimization process, neglecting the role of architectures in causal learning on graphs. In contrast, in this paper, we focus on handling distribution shifts in the graph architecture search process from the causal perspective by discovering the causal relationship between graphs and architectures.

REFERENCES

[1] Raghavendra Addanki, Shiva Kasiviswanathan, Andrew McGregor, and Cameron Musco. Efficient intervention design for causal discovery with latents. In *International Conference on Machine Learning*, pages 63–73. PMLR, 2020.

[2] Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. *Advances in Neural Information Processing Systems*, 33:21865–21877, 2020.

[3] Shaofei Cai, Liang Li, Jincan Deng, Beichen Zhang, Zheng-Jun Zha, Li Su, and Qingming Huang. Rethinking graph neural architecture search from message-passing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6657–6666, 2021.

[4] 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.

[5] Shaohua Fan, Xiao Wang, Yanhu Mo, Chuan Shi, and Jian Tang. Debiasing graph neural networks via learning disentangled causal substructure. In *NeurIPS*, 2022.

[6] Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, and Yue Hu. Graph neural architecture search. In *International joint conference on artificial intelligence*. International Joint Conference on Artificial Intelligence, 2021.

[7] Chaoyu Guan, Xin Wang, Hong Chen, Ziwei Zhang, and Wenwu Zhu. Large-scale graph neural architecture search. In *International Conference on Machine Learning*, pages 7968–7981. PMLR, 2022.

[8] Chaoyu Guan, Xin Wang, and Wenwu Zhu. Autoattend: Automated attention representation search. In *International conference on machine learning*, pages 3864–3874. PMLR, 2021.

[9] 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.

[10] Yesmina Jaafra, Jean Luc Laurent, Aline Deruyver, and Mohamed Saber Naceur. Reinforcement learning for neural architecture search: A review. *Image and Vision Computing*, 89:57–66, 2019.

[11] Amin Jaber, Murat Kocaoglu, Karthikeyan Shanmugam, and Elias Bareinboim. Causal discovery from soft interventions with unknown targets: Characterization and learning. *Advances in neural information processing systems*, 33:9551–9561, 2020.

[12] Haoyang Li, Xin Wang, Ziwei Zhang, and Wenwu Zhu. Out-of-distribution generalization on graphs: A survey. *arXiv preprint arXiv:2202.07987*, 2022.

[13] Haoyang Li, Ziwei Zhang, Xin Wang, and Wenwu Zhu. Learning invariant graph representations for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 35:11828–11841, 2022.

[14] Sihang Li, Xiang Wang, An Zhang, Yingxin Wu, Xiangnan He, and Tat-Seng Chua. Let invariant rationale discovery inspire graph contrastive learning. In *ICML*, pages 13052–13065, 2022.

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

[16] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2018.

[17] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 2021.

[18] Jovana Mitrovic, Brian McWilliams, Jacob C Walker, Lars Holger Buesing, and Charles Blundell. Representation learning via invariant causal mechanisms. In *ICLR*, 2020.

[19] Judea Pearl. *Causality*. Cambridge university press, 2009.

[20] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[21] Yijian Qin, Xin Wang, Ziwei Zhang, Pengtao Xie, and Wenwu Zhu. Graph neural architecture search under distribution shifts. In *International Conference on Machine Learning*, pages 18083–18095. PMLR, 2022.

[22] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *International conference on machine learning*, pages 2902–2911. PMLR, 2017.

[23] Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv:2108.13624*, 2021.

[24] Yongduo Sui, Xiang Wang, Jiancan Wu, Min Lin, Xiangnan He, and Tat-Seng Chua. Causal attention for interpretable and generalizable graph classification. *Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2022.

[25] 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. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[26] Lanning Wei, Huan Zhao, Quanming Yao, and Zhiqiang He. Pooling architecture search for graph classification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2091–2100, 2021.

[27] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. Handling distribution shifts on graphs: An invariance perspective. *International Conference on Learning Representations*, 2022.

[28] Ying-Xin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat seng Chua. Discovering invariant rationales for graph neural networks. In *ICLR*, 2022.

[29] 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*, 2021.

[30] Yingxin Wu, Xiang Wang, An Zhang, Xiangnan He, and Tat-Seng Chua. Discovering invariant rationales for graph neural networks. In *ICLR*, 2022.

[31] 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.

[32] Yang Yao, Xin Wang, Yijian Qin, Ziwei Zhang, Wenwu Zhu, and Hong Mei. Data-augmented curriculum graph neural architecture search under distribution shifts. 2024.

[33] Peng Ye, Baopu Li, Yikang Li, Tao Chen, Jiayuan Fan, and Wanli Ouyang. beta-darts: Beta-decay regularization for differentiable architecture search. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10864–10873. IEEE, 2022.

[34] Dong Zhang, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, and Qianru Sun. Causal intervention for weakly-supervised semantic segmentation. *NeurIPS*, pages 655–666, 2020.

[35] Zeyang Zhang, Ziwei Zhang, Xin Wang, Yijian Qin, Zhou Qin, and Wenwu Zhu. Dynamic heterogeneous graph attention neural architecture search. In *Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023.

[36] Barret Zoph and Quoc Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2016.