Investigating Out-of-Distribution Generalization of GNNs: An Architecture Perspective

Kai Guo*
School of Artificial Intelligence, Jilin University
Changchun, Jilin, China
guokai20@mails.jlu.edu.cn

Hongzhi Wen
Department of Computer Science and Engineering, Michigan State University
East Lansing, Michigan, USA
wenhongz@msu.edu

Wei Jin
Department of Computer Science, Emory University
Atlanta, Georgia, USA
wei.jin@emory.edu

Yaming Guo
School of Artificial Intelligence, Jilin University
Changchun, Jilin, China
guoym21@mails.jlu.edu.cn

Jiliang Tang
Department of Computer Science and Engineering, Michigan State University
East Lansing, Michigan, USA
tangjili@msu.edu

Yi Chang†
School of Artificial Intelligence, Jilin University
Changchun, Jilin, China
yichang@jlu.edu.cn

Abstract

Graph neural networks (GNNs) have exhibited remarkable performance under the assumption that test data comes from the same distribution of training data. However, in real-world scenarios, this assumption may not always be valid. Consequently, there is a growing focus on exploring the Out-of-Distribution (OOD) problem in the context of graphs. Most existing efforts have primarily concentrated on improving graph OOD generalization from two **model-agnostic** perspectives: data-driven methods and strategy-based learning. However, there has been limited attention dedicated to investigating the impact of well-known **GNN model architectures** on graph OOD generalization, which is orthogonal to existing research. In this work, we provide the first comprehensive investigation of OOD generalization on graphs from an architecture perspective, by examining the common building blocks of modern GNNs. Through extensive experiments, we reveal that both the graph self-attention mechanism and the decoupled architecture contribute positively to graph OOD generalization. In contrast, we observe that the linear classification layer tends to compromise graph OOD generalization capability. Furthermore, we provide in-depth theoretical insights and discussions to underpin these discoveries. These insights have empowered us to develop a novel GNN backbone model, DGAT, designed to harness the robust properties of both graph self-attention mechanism and the decoupled architecture. Extensive experimental results demonstrate the effectiveness of our model under graph OOD, exhibiting substantial and consistent enhancements across various training strategies. Our codes are available at https://github.com/KaiGuo20/DGAT.

CCS Concepts

• **Computing methodologies** → **Neural networks**.

Keywords

out-of-distribution generalization, graph neural networks, self-attention, decoupled architecture

ACM Reference Format:
Kai Guo, Hongzhi Wen, Wei Jin, Yaming Guo, Jiliang Tang, and Yi Chang. 2024. Investigating Out-of-Distribution Generalization of GNNs: An Architecture Perspective. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671792

1 Introduction

Graph Neural Networks (GNNs) have emerged as a powerful tool for representation learning on various graph-structured data, such as social networks [20, 49, 50], citation networks [5, 13, 16, 26, 53], biological networks [38, 42], epidemiological networks [27] and product graphs [6, 8, 40]. The representations learned by GNNs can tremendously facilitate diverse downstream tasks, including node classification [28, 30, 35], graph classification [18, 33, 41], and link prediction [54, 56]. In many of these tasks, it is conventionally assumed that training and test datasets are drawn from an identical distribution. Nonetheless, this assumption is often contravened in practical scenarios [44, 45]. For instance, for paper classification on citation graphs, models may be trained on papers from a specific timeframe but later be required to make predictions for more recent publications [45]. Such discrepancies between training and test data distributions outline the out-of-distribution (OOD) challenge.

Recent studies on GNNs have pointed out potential vulnerabilities when these models face distributional shifts [3, 22, 23, 44]. To counteract this, existing techniques aim to enhance graph OOD

*Work done while author was visiting Michigan State University.
†Corresponding author

KDD '24, August 25–29, 2024, Barcelona, Spain.

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671792

generalization for node classification tasks majorly from two perspectives: data-based and learning-strategy-based methods. Data-based methods focus on manipulating the input graph data to boost OOD generalization. Examples of such strategies include graph data augmentation [11, 51] and graph transformation [15]. On the other hand, learning-strategy-based methods emphasize modifying training approaches by introducing specialized optimization objectives and constraints. Notable methods encompass graph invariant learning [45, 47] and regularization techniques [55]. When integrated with existing GNN backbone models [10, 16, 37, 43], these methods can enhance their OOD generalization capabilities.

While these techniques have made strides in addressing graph OOD challenges, they primarily focus on external techniques for improving OOD generalization. Moreover, there is a phenomenon where different GNN models exhibit varying performance in graph OOD scenarios[15]. **Hence, there remains a significant gap in our understanding of the inherent OOD generalization capabilities of different GNN backbone architectures.** This lack of insight raises critical questions: *Which GNN architecture is best suited when dealing with OOD scenarios? Are some models naturally more robust than others?* There is a pressing need to delve deeper into these architectures, comprehensively assess their innate capabilities, and provide clearer guidelines for their deployment in OOD situations.

Research Questions. To bridge the gap, this paper presents the first systematic examination on OOD generalization abilities of popular GNN architectures, specifically targeting the node classification task. Our analysis is framed around three key questions:

Q1: How do distinct GNN architectures perform when exposed to OOD data?

Q2: If there are performance differences as indicated in Q1, can we identify specific designs within GNN architectures responsible for these variations? What could be the underlying causes?

Q3: Informed by the insights from the previous questions, can we develop new GNN designs that enhance OOD generalization?

Contributions. By addressing the above questions, our contributions are summarized as follows:

A1: We rigorously assess a set of common building modules of GNNs, including attention mechanism, decoupled architecture and linear classification layer. Our empirical investigation reveals that both the attention mechanism and the decoupled architecture contribute positively to OOD generalization. Conversely, we observe that the linear classification layer tends to impair the OOD generalization capability.

A2: For a deeper understanding, we delve into the reasons why certain building modules enhance OOD generalization and provide corresponding analysis. We demonstrate that the graph self-attention mechanism in graphs adhering to the information bottleneck principle is beneficial for OOD generalization.

A3: Based on our findings, we introduce a novel GNN design, Decoupled Graph Attention Network (DGAT), which combines the attention mechanism with the decoupled architecture, enabling dynamical adjustments of the propagation weights and separating propagation from feature transformation.

Our major contribution lies in **the systematic investigation of the impact of GNN architectural modules** on OOD scenarios. Remarkably, **our study is orthogonal to existing research efforts of model-agnostic solutions**. Indeed, our findings can complement existing external strategies. DGAT achieves superior performance against other GNN backbone models when trained using various OOD algorithms.

2 Preliminaries

2.1 Graph OOD Generalization Problem

The aim of our study is to investigate the out-of-distribution (OOD) generalization problem in graph domain from an underexplored perspective, the GNN backbone models. As a preliminary, we first introduce the graph OOD problem, and then discuss representative backbones for graph OOD generalization.

The OOD problem originates in the distribution shifts between training and test data. In a supervised learning setting, such distribution shifts can be defined as two types: i.e., covariate shift and concept shift [11].

In the realm of graph OOD, the input X is specified as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with N nodes $v_i \in \mathcal{V}$, edges $(v_i, v_j) \in \mathcal{E}$, an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$. Therefore, the covariate variable consists of an input pair of $(\mathbf{X}, \mathbf{A})$, where $\mathbf{X} \in \mathbb{R}^{N\times d}$ now denotes the node feature matrix. Consequently, graph OOD problems involve distribution shifts with both $\mathbf{X}$ and $\mathbf{A}$, which is more intricate than the general OOD problem. Graph distribution shifts can also be defined as two types: i.e., covariate shift and concept shift. Covariate shift refers to the distribution shift in input variables between training and test data. Formally, $P_{\text{tr}}(\mathbf{X}, \mathbf{A}) \neq P_{\text{te}}(\mathbf{X}, \mathbf{A})$. On the other hand, concept shift depicts the shift in conditional distribution $P(Y \mid \mathbf{X}, \mathbf{A})$ between training and test data, i.e., $P_{\text{tr}}(Y \mid \mathbf{X}, \mathbf{A}) \neq P_{\text{te}}(Y \mid \mathbf{X}, \mathbf{A})$.

The aim of graph OOD generalization is to bolster model performance in OOD scenarios. To solve an OOD problem, we deploy a GNN backbone model, symbolized as a mapping function $f_\theta(\mathbf{X}, \mathbf{A})$ with learnable parameter θ. The graph OOD generalization problem can be articulated as:

$$\min_{\theta} \max_{e\in\mathcal{E}} \mathbb{E}_{(\mathbf{X},\mathbf{A},Y)\sim p(\mathbf{X},\mathbf{A},Y|e=e)} [\mathcal{L}(f_\theta(\mathbf{X}, \mathbf{A}), Y) \mid e] \tag{1}$$

where $\mathcal{L}(\cdot)$ represents a loss function, and e denotes the environment. In the graph machine learning research community, much attention has been dedicated [29, 45] to refining loss function $\mathcal{L}$, optimizing θ, or augmenting on $\mathbf{X}$ and $\mathbf{A}$. However, our approach diverges from these methods. Instead, we focus on examining the impact of the design choices in the backbone model f. To the best of our knowledge, our study offers the first systematic analysis of GNN backbone architectures' effects in OOD contexts.

2.2 Graph Neural Network Architectures

Next, we briefly compare the architectures of classic GNN models that are investigated in our analysis in Section 3.

GCN [16]. The Graph Convolutional Network (GCN) stands as the most representative model, and is chosen as the standard baseline model in our study. A graph convolutional layer comprises a pair of aggregation and transformation operators. At the l-th layer, the mathematical representation of the graph convolutional

layer is as follows:

$$\mathbf{Z}^{(l)} = \sigma(\hat{\mathbf{A}}\mathbf{Z}^{(l-1)}\mathbf{W}^{(l)}), \tag{2}$$

where $\mathbf{Z}^{(l)}$ denotes the node representation of $l-th$ layer. $\mathbf{W}^{(l)}$ represents the linear transformation matrix. $\sigma(\cdot)$ is the nonlinear activation function. Note that $\hat{\mathbf{A}}$ is the normalized adjacency matrix, calculated from $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$, where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with added self-loops, $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$ is the degree matrix.

When applying the GCN model, there are two prevalent implementations. The first [11] adds a linear prediction layer after the final graph convolutional layer (noted as layer L), projecting the dimension of $\mathbf{Z}^{(L)}$ to that of the label Y. The second [16], noted as **GCN−**, omits this prediction layer, adjusting dimensions directly in the last convolutional layer by altering the dimension of $\mathbf{W}^{(L)}$. While the distinction between these methods might be overlooked, it can influence OOD performance. We have compared both implementations in Section 3, and our findings and analysis are detailed in Section 3.4.

GAT [37]. In contrast to the fixed aggregation in GCN, Graph Attention Networks (GAT) employ a self-attention mechanism to assign distinct weights to neighboring nodes during aggregation. The attention mechanism can be expressed as:

$$\alpha_{ij} = \frac{\exp\left(\sigma\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{Z}_i \| \mathbf{W}\mathbf{Z}_j\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\sigma\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{Z}_i \| \mathbf{W}\mathbf{Z}_k\right]\right)\right)}, \tag{3}$$

where $\mathbf{a}$ is a learnable weight vector, $\mathbf{Z}_i$ is the representation vector of node i, and $\mathcal{N}_i$ is the neighborhood of node i in the graph. In light of its connection with GCN, GAT can be studied as an object to investigate the **attention mechanism** under the controlled variable of GCN. We conduct the experiments in Section 3 and our findings are presented in Section 3.2.

SGC [43]. The Simplifying Graph Convolutional Network (SGC) is a decoupled GNN model, where "**decoupled**" refers to a GNN model that separates the neural network transformation operators from the propagation (a.k.a, aggregation) operators [7]. Formally, SGC can be defined as:

$$\mathbf{Z} = \hat{\mathbf{A}}^K\mathbf{X}\mathbf{W}, \tag{4}$$

where $\hat{\mathbf{A}}^K$ can be seen as a composition of K propagation layers, and $\mathbf{W}$ is a simple transformation layer. SGC can serve as the experimental group to study the influence of the **decoupled architecture** on GCN performance, while a K layer GCN acts as the control group. The results of the control experiment are presented in Section 3.3.

APPNP [10]. APPNP is another decoupled model, which starts with transformation and subsequently proceeds to propagation. It can be expressed as:

$$\begin{aligned}\mathbf{Z}^{(0)} &= \mathbf{H} = f_\theta(\mathbf{X}), \mathbf{Z}^{(k+1)} = (1-\beta)\hat{\mathbf{A}}\mathbf{Z}^{(k)} + \beta\mathbf{H},\\ \mathbf{Z}^{(K)} &= \text{softmax}\left((1-\beta)\hat{\mathbf{A}}\mathbf{Z}^{(K-1)} + \beta\mathbf{H}\right),\end{aligned} \tag{5}$$

where f_θ represents a composition of multiple transformation layers (i.e., an MLP), $\mathbf{H}$ is the node representation of MLP, and β specifies the teleport probability of personalized PageRank propagation. Only when β is set to 0, the propagation is equivalent to GCN and SGC. Another noticeable distinction between APPNP and SGC is the order of propagation and transformation layers. Therefore, in Section 3, we have two settings for APPNP, with and without controlling $\beta = 0$. These settings aim to distinguish between the effects of **decoupled architecture** and teleport in propagation. Detailed analyses are provided in Section 3.3.

3 Investigating OOD Generalization of GNN Architectures

Graph out-of-distribution (OOD) generalization has primarily been addressed through learning-strategy-based and data-based methods. While these model-agnostic approaches provide flexibility, the potential influence of backbone GNN architectures on OOD generalization remains less explored. To delve deeper into this, we initiated a systematic analysis of various GNN architectures in OOD scenarios. When designing the study, we note that GNN architectures comprise multiple optional components, such as attention mechanisms and decoupled propagation. To evaluate their individual impacts, we adopted the most classic GCN model as our baseline and embarked on a series of ablation studies. Our ablation studies focus on three primary modifications: (1) substituting graph convolution with an attention mechanism, (2) decoupling feature transformation and feature propagation, and (3) removing the GCN's linear prediction layer. By controlling external variables, we were able to distinguish the contributions of these individual factors. The following sections provide detailed settings of our experiments and the consequential findings regarding these architectural components.

3.1 Experimental Setup

Evaluation Metric. Our study aims to assess the impact of various backbone architectures on graph OOD generalization. In previous literature, researchers primarily use OOD test performance as their main metric. Yet, models display performance variations both in in-distribution (IID) and OOD settings. This suggests that OOD test performance alone is insufficient for a holistic evaluation of OOD generalization. To address this, we adopt the IID/OOD generalization gap metric from the NLP and CV domains [14, 48] for graph OOD analysis. The IID/OOD generalization gap depicts the difference between IID and OOD performance, offering a measure of models' sensitivity and robustness to distribution shifts. It is defined as: $\underline{\textbf{GAP} = \textbf{IID}_{\textbf{test}} - \textbf{OOD}_{\textbf{test}}}$, where IID_{test} and OOD_{test} are the test performance on IID and OOD test datasets, respectively.

To rigorously determine the component's influence on graph OOD generalization, we performed a paired T-Test between GCN baseline and other models. Evaluating across 10 runs for each model on every dataset, a p-value < 0.05 indicates a significant difference, with the t-value highlighting the superior model.

Dataset. We utilized the GOOD benchmark [11] for evaluating graph OOD generalization on node classification task. This benchmark offers a unique capability to simultaneously measure both IID and OOD performance, which is a feature not present in prevalent datasets from other studies, e.g., EERM [45]. Such simultaneous measurement is vital as calculating the GAP depends on both performance. The GOOD benchmark comprises citation networks, gamer networks, university webpage networks, and synthetic datasets, and delineates shifts as either covariate or concept shifts. From this collection, we selected 11 datasets, excluding CBAS due to its

limited node size and feature dimension, and WebKB-university-covariate as all models exhibited high variance.

Implementation of GNN models. First, we follow the implementation of GOOD benchmark and use GCN [16] as our baseline model, which concludes with a linear prediction layer. In the GOOD paper [11], models and hyperparameters are selected based on an OOD validation set. In contrast, our study emphasizes the inherent robustness of backbone models to unanticipated distribution shifts common in real-world scenarios. Consequently, we determine optimal hyperparameters using the IID validation set (Appendix A.4).

Next, to establish a comparative analysis framework, we implement other GNN models: GCN−, GAT [37], SGC [43], and APPNP [10], as delineated in Section 2.2. Each implementation ensures consistency by maintaining all factors constant, except for the specific design under consideration. For instance, while deploying SGC, we align its hidden dimensions and propagation layers with those of the GCN baseline, even adjusting the transformation layer size to make it slightly different from the original SGC (i.e., the linear transformation in Eqn. 4 is replaced by an MLP). This adjustment isolates the "decoupled architecture" as the sole variation between SGC and GCN. By comparing these models with GCN, we discern how their distinct architectures influence graph OOD generalization.

In the following, we examine the impact of common GNN building blocks, i.e., **attention mechanism**, **coupled/decoupled architecture**, and **linear prediction layer**, respectively.

3.2 Impact of Attention Mechanism

In Table 1, we assess the impact of attention mechanism by comparing OOD performance of GAT against GCN. GAT surpasses GCN on 10 out of 11 datasets, and shows a lower GAP value on 9 out of 11 datasets. Particularly, on the Arxiv-time-degree dataset, GAT improves OOD performance by 5.1% relative to GCN and decreases the GAP by 36.4%, **underscoring the advantages of attention mechanism for graph OOD generalization**.

To statistically validate these observations, we further applied T-Tests to the OOD results of both models on each dataset. In Table 1, red numbers denote that GAT significantly outperforms GCN, while blue signifies the opposite. The data reveals GAT's significant advantage on 7 datasets, further emphasizing the potency of attention in graph OOD generalization. The detailed T-Test results are reported in Figure 4.(a) in Appendix A.1.

Theoretical Insights. Next, we present a theoretical analysis that delves into the success of GAT, elucidating why graph attention yields advantages for OOD generalization. Our analysis comprises two key components: (1) We establish a compelling link between the graph attention mechanism and the fundamental concept of information bottleneck; and (2) We discuss that optimizing the information bottleneck can notably enhance OOD generalization capabilities. At the start of our analysis, we introduce the concept of the information bottleneck (IB).

We denote the variables of input node features as X, and the variables of the output representations as Z. Thus, the mapping function of GNN $f_\theta(\cdot)$ can be expressed as $f(Z \mid X)$. Consider a distribution $X \sim \text{Gaussian}(X', \epsilon)$ with X as the noisy input variable, X' as the clean target variable, and ϵ as the variance for the Gaussian noise. Following Kirsch et al. [17], the information bottleneck principle involves minimizing the mutual information between the input X and its latent representation Z while still accurately predicting X' from Z, and can be formulated as:

$$f^*_{\text{IB}}(Z \mid X) = \arg\min_{f(Z|X)} I(X, Z) - I\left(Z, X'\right) \tag{6}$$

where $I(\cdot, \cdot)$ stands for the mutual information. With the aforementioned notations and concepts, we now introduce our proposition.

Proposition 1. *Given a node i with its feature vector x_i and its neighborhood $\mathcal{N}(i)$, the following aggregation scheme for obtaining its hidden representation $\mathbf{z}_i$,*

$$\mathbf{z}_i = \sum_{j \in \mathcal{N}(i)} \frac{\eta_i \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)}{\sum_{j \in \mathcal{N}(i)} \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)} \mathbf{x}_j, \tag{7}$$

with $\eta_i, \mathbf{W}_Q, \mathbf{W}_K$ being the learnable parameters, can be understood as the iterative process to optimize the objective in Eq. (6).

The detailed proof of the proposition mentioned above is available in Appendix A.2. Proposition 1 unveils an intriguing relationship between the aggregation scheme in Eq. (7) and the information bottleneck principle in Eq. (6): it demonstrates that the information bottleneck principle can be approached by adaptively aggregating similar node features into a learnable representation vector. It is worth noting that the aggregation scheme employed in Graph Attention Networks (GAT) (Eq. (3)) can be viewed as a specific instance of Eq. (7) under certain conditions: (1) GAT sets η_i to a constant value; and (2) GAT computes attention using a weight matrix multiplication on the concatenated node pair vector. Given the insights provided by the proposition, it is reasonable to infer that graph self-attention mechanism updates the node embeddings following an IB principle.

Furthermore, we highlight that the information bottleneck principle plays a pivotal role in enhancing the OOD generalization of neural networks. Notably, Yang et al. [46] suggests that the information bottleneck can help GNNs simultaneously discard spurious features and learn invariant features, thereby achieving OOD generalization. Li et al. [21] and Ahuja et al. [1] have also demonstrated the effectiveness of the information bottleneck for OOD generalization. Consequently, we postulate that the reason the graph attention mechanism contributes to the OOD generalization of GNNs is intricately tied to its connection with the information bottleneck principle.

3.3 Impact of Coupled/Decoupled Structure

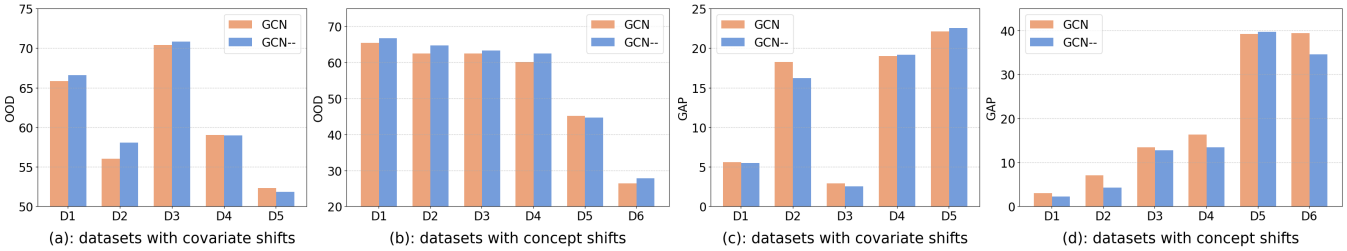
In order to compare coupled and decoupled structures, we evaluate the performance of various decoupled GNNs, presented in Table 2. For both OOD tests and GAP values, SGC surpasses GCN on merely 5 out of 11 datasets. In contrast, when the order of propagation and transformation is reversed, APPNP ($\beta = 0$) exceeds GCN's performance on 9 out of 11 datasets and demonstrates a lower GAP value on 9 out of 11 datasets. This reveals the significance of the transformation-propagation order in OOD generalization, **suggesting a preference for transformation prior to propagation in graph OOD contexts**. The T-Test results are presented in the same color scheme as in Section 3.2, which further confirms our initial observation.

Table 1: OOD and GAP performances for investigating the impact of self-attention. All numerical results are averages across 10 random runs. Red color indicates the statistically significant improvement(i.e., $P_{value} < 0.05$ and $T_{value} > 0$) over the GCN. Blue color indicates the statistically significant worse(i.e., $P_{value} < 0.05$ and $T_{value} < 0$) over the GCN. The best performance in each dataset is highlighted in bold. OOD indicates the OOD performance on OOD test data.

		G-Cora-Word		G-Cora-Degree		G-Arxiv-Time		G-Arxiv-Degree		G-Twitch-Language		G-WebKB-University	
		OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓
Covariate	GCN	65.85	**5.62**	56.05	18.26	70.38	2.90	59.05	19.01	52.32	22.13	-	-
	GAT	**66.23**	5.77	**56.12**	**17.99**	**70.95**	**2.08**	**59.32**	**18.78**	**52.83**	**21.17**	-	-
Concept	GCN	65.44	3.05	62.48	7.05	62.50	13.42	60.13	16.36	**45.12**	**39.25**	26.42	39.41
	GAT	**65.86**	**2.21**	**63.85**	**4.83**	**64.96**	**10.89**	**63.07**	**11.99**	44.14	40.81	**29.27**	**35.23**

Table 2: OOD and GAP performances for investigating the impact of decoupled architecture. All numerical results are averages across 10 random runs. Red color indicates the statistically significant improvement(i.e., $P_{value} < 0.05$ and $T_{value} > 0$) over the GCN. Blue color indicates the statistically significant worse(i.e., $P_{value} < 0.05$ and $T_{value} < 0$) over the GCN. The best performance in each dataset is highlighted in bold.

		G-Cora-Word		G-Cora-Degree		G-Arxiv-Time		G-Arxiv-Degree		G-Twitch-Language		G-WebKB-University	
		OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓
covariate	GCN	65.85	5.62	56.05	18.26	70.38	2.90	59.05	19.01	52.32	22.13	-	-
	SGC	66.19	5.61	55.43	18.46	70.54	**2.27**	**59.66**	**18.10**	54.03	20.40	-	-
	APPNP($\beta = 0$)	66.66	**5.16**	56.14	17.83	**70.69**	2.84	59.33	18.67	51.66	22.18	-	-
	APPNP	**67.48**	6.37	**58.33**	**17.16**	69.22	3.38	55.40	22.10	**56.47**	**16.75**	-	-
concept	GCN	65.44	3.05	62.48	7.05	62.50	13.42	60.13	16.36	45.12	39.25	26.42	39.41
	SGC	65.21	3.16	62.41	6.77	63.88	11.94	56.66	20.26	44.22	40.52	25.78	40.38
	APPNP($\beta = 0$)	66.21	**2.81**	64.09	**4.85**	**65.10**	**10.94**	**65.44**	9.86	44.10	39.87	29.17	35.83
	APPNP	**66.46**	4.24	**64.81**	5.55	63.46	11.51	64.28	**9.51**	**46.81**	**35.99**	**30.28**	**35.22**



(a): datasets with covariate shifts (b): datasets with concept shifts (c): datasets with covariate shifts (d): datasets with concept shifts

Figure 1: Comparision of OOD and GAP between GCN and GCN– for investagating the impact of linear classifier. GCN– means GCN without linear classifier. D1, D2, D3, D4, D5, D6 represent G-Cora-Word, G-Cora-Degree, G-Arxiv-Time, G-Arxiv-Degree, G-Twitch-Language and G-WebKB-University respectively.

Such observation can be potentially explained from the theory that decoupled graph neural networks are equivalent to label propagation, proposed by Dong et al. [7]. From a label propagation perspective, the models propagate known labels across the graph to generate pseudo-labels for unlabeled nodes. These pseudo-labels then optimize the model predictor. The augmentation with pseudo-labels may curb overfitting and the architecture's training approach can adaptively assign structure-aware weights to these pseudo-labels [7]. This might account for the enhanced OOD generalization performance seen in the decoupled architecture.

3.4 Impact of Linear Prediction Layer

Lastly, we evaluate the impact of the linear prediction layer on a GCN. As shown in Figure 1, GCN– surpasses GCN on 8 out of 11 OOD datasets and achieves a lower GAP value on 8 out of 11 datasets. This indicates that **removing the last linear prediction layer can enhance graph OOD performance**. The T-Test results are reported in Figure 4.(b) in Appendix A.1.

The performance dip of the linear prediction layer might be attributed to two factors. First, introducing an extra linear prediction layer might lead to surplus parameters and higher model complexity, amplifying the overfitting risk on IID. Second, the propagation process is non-parametric and has a lower risk of overfitting. It may be advantageous in both IID and OOD contexts. In contrast, the additional linear prediction layer is fit to the training data's label distribution, potentially hindering performance when faced with OOD distribution shifts. By omitting this layer, we amplify the graph's intrinsic structure, leading to improved OOD generalization. Additionally, Tang and Liu [36] argue that the Lipschitz continuity constant of APPNP can be smaller than that of GCN, suggesting that APPNP may achieve lower generalization error than GCN.

This also provides a favorable explanation for the ability of decoupled architectures to enhance out-of-distribution generalization capabilities.

4 New GNN Design for Enhanced OOD Generalization

In the previous section, we provided both empirical results and theoretical analysis that show the beneficial roles of the attention mechanism and the decoupled architecture in enhancing the OOD generalization of GNNs. On the other hand, we noted that the linear prediction layer detracts from OOD generalization. **Motivated by these observations, we propose to merge the attention mechanism with the decoupled architecture, opting to omit the linear prediction layer.** Specifically, we calculate attention scores from transformed features and employ these scores throughout each propagation layer. In the following, we delve into the details of our proposed model, Decoupled Graph Attention Network (DGAT).

4.1 A New GNN Design

Our proposed DGAT model integrates the principles of decoupled architecture and the attention mechanism, both of which have demonstrated positive contributions to OOD performance in previous sections. To adopt a decoupled architecture, we separate the linear transformation and propagation operations and conduct the linear transformation prior to propagation. Furthermore, we inject the attention mechanism into the propagation operations. As a result, our model includes three components: linear transformation, attention score computation, and adaptive propagation. The framework is illustrated by Figure 2.

Linear Transformation. Our DGAT model decouples the GNN into transformation and propagation. This design can enhance OOD performance as discussed in Section 3.3. Therefore, DGAT first applies two linear transformation layers to the input data.

$$\begin{aligned} \mathbf{Z}^{(\text{init})} &= \sigma(\boldsymbol{W}^{(\text{init})}\mathbf{X} + \boldsymbol{b}^{(\text{init})}) \quad \in \mathbb{R}^{N\times d} \\ \mathbf{Z}^{(0)} &= \boldsymbol{H} = \boldsymbol{W}^{(0)}\mathbf{Z}^{(\text{init})} + \boldsymbol{b}^{(0)} \quad \in \mathbb{R}^{N\times c} \end{aligned} \tag{8}$$

where d is the hidden dimension and c indicates the number of classes. The second linear layer maps features to the label space. These pseudo-labels help mitigate overfitting and dynamically assign structure-aware weights to them.

Attention Score Computation. Our experimental findings in Section 3.2 highlight the advantages of the attention mechanism for handling graph OOD, supported by the theoretical evidence that the graph self-attention mechanism aligns with the information bottleneck principle. To incorporate the attention mechanism into our model, we calculate attention scores based on the node representation $\mathbf{Z}^{(\text{init})}$, formulated as:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{Z}_i^{(\text{init})} \| \mathbf{W}\mathbf{Z}_j^{(\text{init})}\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{Z}_i^{(\text{init})} \| \mathbf{W}\mathbf{Z}_k^{(\text{init})}\right]\right)\right)}, \tag{9}$$

where α_{ij} represents the attention score. We use $\boldsymbol{P}$ to denote the attention matrix, where $\boldsymbol{P}_{ij} = \alpha_{ij}$. We compute α_{ij} using $\mathbf{Z}^{(\text{init})}$ instead of $\mathbf{Z}^{(0)}$, because $\mathbf{Z}^{(0)}$ usually has a low dimensionality which can hamper the identification of important nodes.

Adaptive Propagation. The last step in DGAT is propagation. Instead of a traditional fixed propagation, DGAT achieves adaptive propagation by combining attention score matrices and adjacency matrices. We define $\hat{\tilde{\mathbf{A}}} := (1-\gamma)\boldsymbol{P} + \gamma\hat{\boldsymbol{A}}$. The adaptive propagation function is expressed as follows:

$$\begin{aligned} \mathbf{Z}^{(k+1)} &= (1-\beta)\hat{\tilde{\mathbf{A}}}\mathbf{Z}^{(k)} + \beta\mathbf{H}, \\ \mathbf{Z}^{(K)} &= \text{softmax}\left((1-\beta)\hat{\tilde{\mathbf{A}}}\mathbf{Z}^{(K-1)} + \beta\mathbf{H}\right), \end{aligned} \tag{10}$$

where β is a hyperparameter to control the trade-off of the initial connection.

Note that we do not employ a linear layer at the end of the model architecture as it can negatively impact the OOD generalization (Section 3.4). As a consequence, our model combines both the decoupled architecture and attention mechanism, creating an elegant fusion of the strengths of the two preceding components.

Complexity analysis. In the following, we demonstrate that the computational complexity of our model is comparable to that of the GCN. We define the adjacency matrix as $\mathbf{A} \in \mathbb{R}^{N\times N}$, the input matrix as $\mathbf{X} \in \mathbb{R}^{N\times d}$, and the transformation matrix as $\mathbf{W} \in \mathbb{R}^{d\times d}$. The operations within a GCN layer result in the following time complexity: $O\left(Nd^2 + N^2d\right)$. Considering that each multiplication with $\mathbf{A}$ is a sparse multiplication, the complexity can be rewritten as $O\left(L|E|d^2 + LN^2d\right)$, where $|E|$ represents the number of edges and L represents the number of layers. Similarly, APPNP has the same computational complexity as GCN.

For self-attention, we need to compute $\mathbf{Q} = \mathbf{X}\mathbf{W}_q$, $\mathbf{K} = \mathbf{X}\mathbf{W}_k$, and $\mathbf{V} = \mathbf{X}\mathbf{W}_v$, each requiring $\mathrm{O}(Nd^2)$ time. Unlike in the GCN case, we also need to compute $\mathbf{Q}\mathbf{K}^\top$ to obtain the attention score $\boldsymbol{\alpha}$, which takes $\mathrm{O}(N^2d)$ time. Finally, computing $\boldsymbol{\alpha}\mathbf{V}$ also takes $\mathrm{O}(N^2d)$ time. These computations result in a total time complexity of $\mathrm{O}(N^2d+Nd^2)$. All of these operations are computed at each layer, leading to the final time complexity of $\mathrm{O}\left(LN^2d + LNd^2\right)$. Therefore, the computational complexity of our model is $O(L|E|d^2 + LN^2d)$, which is comparable to that of GCN.

Advantages. Despite its simplicity, our model stands out by offering several compelling advantages:

(a) *Simple yet robust.* DGAT is grounded in the findings of our prior experimental study and theoretical analysis outlined in Section 3. It enjoys the strengths of essential components that positively contribute to OOD generalization.

(b) *Favorable computational efficiency.* The efficiency analysis above reveals that the computational complexity of DGAT is comparable to that of traditional GNNs, yet it demonstrates superior OOD generalization.

(c) *Compatible with diverse training strategies.* DGAT is orthogonal to external OOD techniques and can achieve further OOD generalization from OOD training strategies. In the following sections, we will empirically verify that this model can function as a powerful backbone for various popular OOD algorithms.

4.2 Experiment

To assess the OOD generalization capabilities of the proposed DGAT, we conduct experiments under various training strategies on node classification tasks. Through experiments, we aimed to answer the following questions: **Q1**: Can DGAT outperform existing GNN

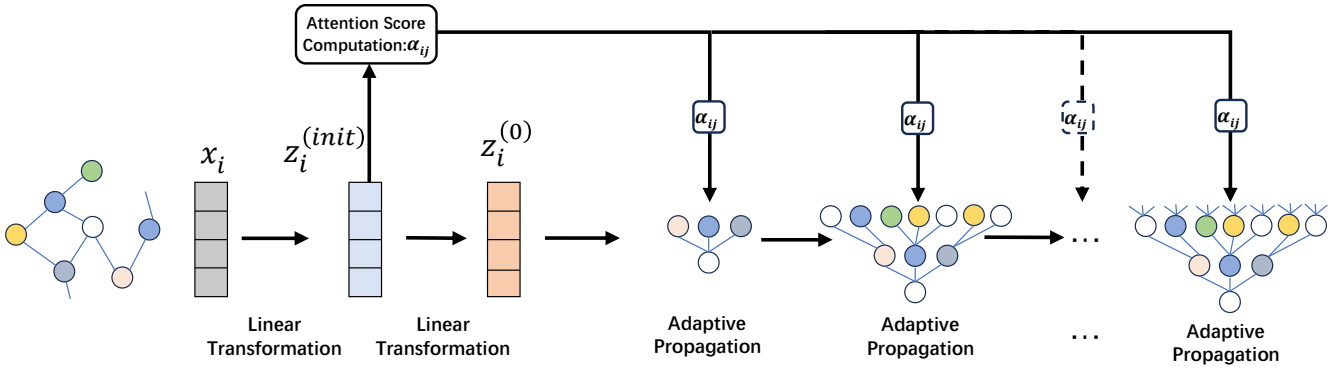


Figure 2: An illustration of our proposed model DGAT. In this decoupled architecture, we calculate attention scores from transformed features and employ these scores throughout each propagation layer.

Table 3: OOD and GAP performances under ERM setting on datasets from GOOD. All results are averages over 10 random runs.

		G-Cora-Word		G-Cora-Degree		G-Arxiv-Time		G-Arxiv-Degree		G-Twitch-Language		G-WebKB-University	
		OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓	OOD↑	GAP↓
Covariate	GCN–	66.56	**5.47**	58.09	16.22	70.82	2.52	58.95	19.15	51.86	22.55	-	-
	SGC	66.23	5.51	55.86	17.98	70.27	1.87	59.36	17.96	53.19	16.26	-	-
	APPNP	67.62	6.28	58.87	16.66	69.2	2.43	56.25	20.74	56.85	16.67	-	-
	GAT	65.84	5.62	58.25	15.66	70.65	2.38	58.45	19.78	53.3	20.63	-	-
	GraphSAGE	65.59	7.67	56.54	19.24	69.36	3.00	57.59	19.67	55.88	17.75	-	-
	GPRGNN	67.59	6.44	59.46	15.75	67.74	2.66	56.68	19.02	56.37	16.05	-	-
	DGAT	**67.67**	6.09	**59.68**	**15.02**	**71.33**	**1.65**	**60.12**	**17.56**	**57.37**	**15.31**	-	-
Concept	GCN–	66.70	2.31	64.72	4.27	63.27	12.78	62.46	13.47	44.72	39.65	27.80	34.54
	SGC	66.28	**1.96**	62.58	7.12	63.33	11.61	55.06	21.28	47.43	35.31	29.63	33.37
	APPNP	67.31	4.12	66.3	4.83	63.64	10.76	63.92	8.83	48.1	34.08	26.88	44.45
	GAT	66.32	2.05	64.41	**4.10**	65.02	10.86	64.75	9.00	43.95	40.89	28.35	**32.65**
	GraphSAGE	65.42	5.46	65.06	5.43	62.85	11.77	60.92	12.96	45.51	39.93	**34.50**	40.67
	GPRGNN	66.95	3.59	65.97	4.96	61.89	10.96	63.05	8.06	**49.07**	**33.08**	27.06	40.27
	DGAT	**67.50**	3.57	**66.36**	4.45	**65.11**	**10.17**	**65.86**	**7.73**	45.10	40.20	33.57	37.92

Table 4: OOD performances under ERM and EERM on datasets from EERM paper. All results are averages over 5 random runs.

Dataset	Method	GCN–	SGC	APPNP	GAT	GraphSAGE	GPRGNN	**DGAT**
Amz-Photo	ERM	93.79±0.97	93.83±2.30	94.44±0.29	96.30±0.79	95.09±0.60	91.87±0.65	**96.56±0.85**
Cora	ERM	91.59±1.44	92.17±2.38	95.16±1.06	94.81±1.28	99.67±0.14	93.00±2.17	**99.68±0.06**
Elliptic	ERM	50.90±1.51	49.19±1.89	62.17±1.78	65.36±2.70	56.12±4.47	64.59±3.52	**73.09±2.14**
OGB-Arxiv	ERM	38.59±1.35	41.44±1.49	44.84±1.43	40.63±1.57	39.56±1.66	44.38±0.59	**45.95±0.65**
Twitch-E	ERM	59.89±0.50	59.61±0.68	61.05±0.89	58.53±1.00	62.06±0.09	59.72±0.40	**62.14±0.23**
Amz-Photo	EERM	94.05±0.40	92.21±1.10	92.47±1.04	95.57±1.32	**95.57±0.13**	90.78±0.52	92.54±0.77
Cora	EERM	87.21±0.53	79.15±6.55	94.21±0.38	85.00±0.96	98.77±0.14	88.82±3.10	**98.85±0.26**
Elliptic	EERM	53.96±0.65	45.37±0.82	58.80±0.67	58.14±4.71	58.20±3.55	67.27±0.98	**68.74±1.12**
OGB-Arxiv	EERM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
Twitch-E	EERM	59.85±0.85	54.48±3.07	62.28±0.14	59.84±0.71	62.11±0.12	61.57±0.12	**62.52±0.09**

architectures on OOD test data? **Q2**: Is DGAT a better backbone model in different OOD generalization methods?

4.2.1 OOD performance of DGAT. To answer **Q1**, we evaluate the performance of our proposed DGAT on ERM setting.

Baselines. We evaluate the performance of our DGAT by comparing it with several state-of-the-art models, including GCN, APPNP, GAT, SGC, GPRGNN [4], and GraphSAGE [12].

Datasets. In this experiment, we selected 11 datasets from the GOOD benchmark, the same as introduced in Section 3. In addition, we conducted experiments on 5 new datasets that are used in EERM [45] paper. These datasets exhibit diverse distribution shifts: Cora and Amz-Photo involve synthetic spurious features; Twitch-E exhibits cross-domain transfer with distinct domains for each graph; Elliptic and OGB-Arxiv represent temporal evolution datasets, showcasing dynamic changes and temporal distribution shifts. The details of these datasets are shown in Appendix A.3

Implementation Details. In Section 3.4, we observed that using linear transformation as the final prediction layer degrades the OOD performance. Consequently, we replace the linear prediction layer with an individual graph convolutional layer for all

the models. Similar to our prior experiments, we train and select model hyperparameters on IID distribution and test the model on OOD distribution. The number of layers is chosen from $\{2, 3\}$. The hidden size is chosen from $\{100, 200, 300\}$. We tune the following hyper-parameters: $\gamma \in \{0, 0.2, 0.5\}$, $\beta \in \{0, 0.1, 0.2, 0.5\}$. The details of parameters are shown in Appendix A.4.

Experimental Results. The results on GOOD datasets are reported in Table 3. From this table, we find that DGAT outperforms baselines on 9/11 datasets for the OOD test. Meanwhile, DGAT exhibits a lower GAP value compared to baselines on 6 out of 11 datasets. For example, DGAT delivers an improvement of 1.7% over baselines for OOD test while achieving a decline of 4.1% over baselines for GAP on GOODArxiv-degree-concept, which indicates the effectiveness of our model for Graph OOD generalization. The results on datasets from EERM paper under ERM setting are reported in Table 4. Remarkably, our proposed DGAT outperforms baseline GNN backbones on all of these datasets.

To further validate the contribution of each component in DGAT, we conduct additional ablation study. The results confirm the effectiveness of our proposed method and are shown in Appendix A.5.

4.2.2 DGAT Performance as a Backbone. In order to answer the **Q2**, we conduct experiments to evaluate our model and baselines across various strategies proposed for OOD. Specifically, we choose GCN– and APPNP that perform well under ERM as the backbones. Representative OOD algorithms such as IRM [2], VREx [19], GroupDRO [29], Graph-Mixup [39] and EERM [45] are considered as baseline methods. Among them, Graph-Mixup and EERM are graph-specific methods. It's worth noting that within the GOOD framework, Graph-Mixup is equipped with GCN incorporating a linear classifier, and Graph-Mixup is more suitable for this framework in APPNP, as it is better suited for enhancement at the hidden dimension level rather than the class dimension. Hence, for Graph-Mixup, we have added a linear classifier on top of the models.

The experimental results illustrating the performances related to OOD and GAP across various training strategies (i.e, IRM, VRex, GroupDRO, Graph-Mixup) on datasets from GOOD are shown in Figure 3. The results of OOD performance under EERM setting on datasets from EERM are reported in Table 4. We have the following observations.

(a) First, **compared to the baselines, our DGAT model consistently demonstrates better OOD generalization when combined with external OOD algorithms.** We find that DGAT outperforms baselines on 9/11 datasets under IRM training strategy. For example, DGAT delivers an improvement of 3.6% over baselines for OOD test on G-Cora-Degree-Covariate under IRM. Meanwhile, DGAT demonstrates superior OOD performance in comparison to baselines across 9 out of 11 datasets under Graph-Mixup training setting. The results in Table 4 shows that our model also achieves better performance on most datasets under EERM training setting.

(b) Second, **OOD training algorithms do not always improve the OOD performance of backbone models**. For example, on G-Cora-Word and G-Cora-Degree datasets, all backbone models suffer from OOD performance degradation when trained with EERM algorithm. This observation is consistent with the results in GOOD paper [11] and highlights the limitation of existing OOD algorithms.

(c) Meanwhile, **we indeed notice a significant improvement in certain cases.** For example, on G-Twitch-language-concept, DGAT equipped with VREx achieves an improvement of 7.0% over ERM. This confirms that combining with an effective training algorithm can further enhance the OOD generalization of our DGAT backbone model.

5 Related work

5.1 OOD generalization

In the real world, when the distribution of training data differs from that of testing data, denoted as $P_{tr}(X, Y) \neq P_{te}(X, Y)$, this distribution shift is referred to as an OOD problem. Common types of distribution shifts including covariate shift, concept shift, and prior shift [24]. To effectively achieve better OOD generalization, several methods have been proposed [2, 9, 19, 29, 34]. For instance, Invariant Risk Minimization (IRM) [2] is a representative method designed to identify invariant features. This approach ensures that the optimal classifier performs consistently across all environments. GroupDRO [29] enhances the model's out-of-distribution (OOD) generalization capabilities by focusing on optimizing for the worst-case scenario across a set of predefined groups and introducing a strong regularization. VREx [19] considers there exists variation among different training domains, and this variation can be extrapolated to test domains. Based on this assumption, the method aims to make the risks across different training domains as consistent as possible, reducing the model's reliance on spurious features.

5.2 OOD generalization on graphs

In graph-structured data, the OOD problem exists as well, but the research on graph OOD is currently in its early stages. The covariate shift and concept shift also exit in the graph domain. Unlike the general OOD problem, in graph-based OOD, shifts can occur not only in features but may also occur implicitly in the graph structure. Some efforts have been proposed to solve the graph OOD problem in node classification tasks from two perspectives: data-based methods and learning-strategy-based methods. Data-based methods focus on manipulating the input graph data to boost OOD generalization [15, 24, 39]. For example, GTrans [15] provides a data-centric view to solve the graph OOD problem and propose to transform the graph data at test time to enhance the ability of graph OOD generalization. On the other hand, learning-strategy-based methods emphasize modifying training approaches by introducing specialized optimization objectives and constraints [25, 45, 47]. For example, EERM [45] seeks to leverage the invariant associations between features and labels across diverse distributions, thereby achieving demonstrably satisfactory OOD generalization in the face of distribution shifts. However, current methods predominantly emphasize external techniques to enhance OOD generalization, yet they do not provide insights into the inherent performance of the underlying backbone models themselves. Therefore, in this work, we investigate the impact of the GNN architecture on graph OOD.

6 Conclusion

GNNs tend to yield suboptimal performance on out-of-distribution (OOD) data. While prior efforts have predominantly focused on enhancing graph OOD generalization through data-driven and

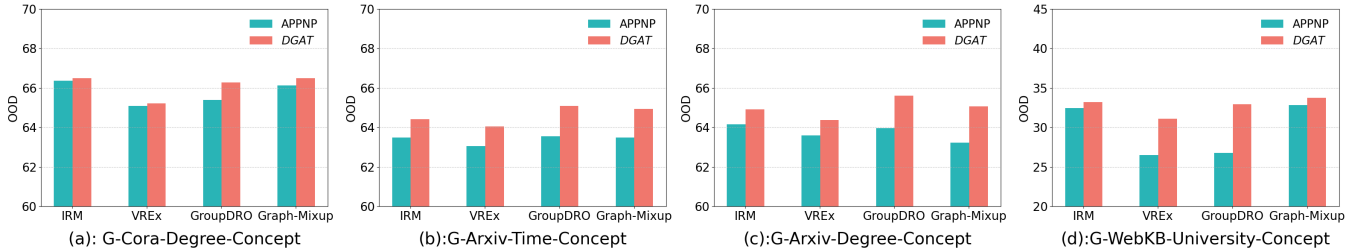


Figure 3: Comparision of OOD performance between DGAT and APPNP equipped with various OOD algorithms.

strategy-based methods, relatively little attention has been devoted to assessing the influence of GNN backbone architectures on OOD generalization. To bridge this gap, we undertake the first comprehensive examination of the OOD generalization capabilities of well-known GNN architectures. Our investigation unveils that both the attention mechanism and the decoupled architecture positively impact OOD generalization. Conversely, we observe that the linear classification layer tends to compromise OOD generalization ability. To deepen our insights, we provide theoretical analysis and discussions. Building upon our findings, we introduce a novel GNN design, denoted as DGAT, which combines the self-attention mechanism and the decoupled architecture. Our comprehensive experiments across a variety of training strategies show that the GNN backbone architecture is indeed important, and that combining useful architectural components can lead to a superior GNN backbone architecture for OOD generalization.

7 Acknowledgments

Kai Guo, Yaming Guo and Yi Chang are supported by the National Key R&D Program of China under Grant No.2023YFF0905400 and the National Natural Science Foundation of China (No.U2341229).

References

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

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

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

[4] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations*. https://openreview.net/forum?id=n6jl7fLxrP

[5] Mauro Conti, Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. 2009. (old) Distributed data source verification in wireless sensor networks. *Inf. Fusion* 10, 4 (2009), 342–353. https://doi.org/10.1016/j.inffus.2009.01.002

[6] Leonardo Cotta, Carlos HC Teixeira, Ananthram Swami, and Bruno Ribeiro. 2020. Unsupervised joint k-node graph representations with compositional energy-based models. *Advances in Neural Information Processing Systems* 33 (2020), 17536–17547.

[7] Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. 2021. On the equivalence of decoupled graph convolution network and label propagation. In *Proceedings of the Web Conference 2021*. 3651–3662.

[8] Wenqi Fan, Xiaorui Liu, Wei Jin, Xiangyu Zhao, Jiliang Tang, and Qing Li. 2022. Graph trend filtering networks for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 112–121.

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

[10] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997* (2018).

[11] Shurui Gui, Xiner Li, Limei Wang, and Shuiwang Ji. 2022. Good: A graph out-of-distribution benchmark. *Advances in Neural Information Processing Systems* 35 (2022), 2059–2073.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[13] Haoyu Han, Xiaorui Liu, Haitao Mao, MohamadAli Torkamani, Feng Shi, Victor Lee, and Jiliang Tang. 2023. Alternately optimized graph neural networks. In *International Conference on Machine Learning*. PMLR, 12411–12429.

[14] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained Transformers Improve Out-of-Distribution Robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2744–2751.

[15] Wei Jin, Tong Zhao, Jiayuan Ding, Yozen Liu, Jiliang Tang, and Neil Shah. 2023. Empowering Graph Representation Learning with Test-Time Graph Transformation. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=Lnxl5pr018

[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[17] Andreas Kirsch, Clare Lyle, and Yarin Gal. 2020. Unpacking information bottlenecks: Surrogate objectives for deep learning. (2020).

[18] Lecheng Kong, Yixin Chen, and Muhan Zhang. 2022. Geodesic graph neural network for efficient graph representation learning. *Advances in Neural Information Processing Systems* 35 (2022), 5896–5909.

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

[20] Chetan Kumar, Riazat Ryan, and Ming Shao. 2020. Adversary for social good: Protecting familial privacy through joint adversarial attacks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 11304–11311.

[21] Bo Li, Yifei Shen, Yezhen Wang, Wenzhen Zhu, Dongsheng Li, Kurt Keutzer, and Han Zhao. 2022. Invariant information bottleneck for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7399–7407.

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

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

[24] Xiner Li, Shurui Gui, Youzhi Luo, and Shuiwang Ji. 2023. Graph Structure and Feature Extrapolation for Out-of-Distribution Generalization. *arXiv preprint arXiv:2306.08076* (2023).

[25] Yang Liu, Xiang Ao, Fuli Feng, Yunshan Ma, Kuan Li, Tat-Seng Chua, and Qing He. 2023. 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*. 1548–1558.

[26] Zhining Liu, Ruizhong Qiu, Zhichen Zeng, Hyunsik Yoo, David Zhou, Zhe Xu, Yada Zhu, Kommy Weldemariam, Jingrui He, and Hanghang Tong. 2024. Class-Imbalanced Graph Learning without Class Rebalancing. In *Forty-first International Conference on Machine Learning*.

[27] Zewen Liu, Guancheng Wan, B Aditya Prakash, Max SY Lau, and Wei Jin. 2024. A Review of Graph Neural Networks in Epidemic Modeling. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)* (2024).

[28] Haitao Mao, Lun Du, Yujia Zheng, Qiang Fu, Zelin Li, Xu Chen, Shi Han, and Dongmei Zhang. 2021. Source free unsupervised graph domain adaptation. *arXiv preprint arXiv:2112.00955* (2021).

[29] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *ICLR* (2020).

[30] Zixing Song, Yifei Zhang, and Irwin King. 2022. Towards an optimal asymmetric graph structure for robust semi-supervised node classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1656–1665.
[31] Susanne Still, William Bialek, and Léon Bottou. 2003. Geometric clustering using the information bottleneck method. *Advances in neural information processing systems* 16 (2003).
[32] DJ Strouse and David J Schwab. 2019. The information bottleneck and geometric clustering. *Neural computation* 31, 3 (2019), 596–612.
[33] Yongduo Sui, Xiang Wang, Jiancan Wu, Min Lin, Xiangnan He, and Tat-Seng Chua. 2022. Causal attention for interpretable and generalizable graph classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1696–1705.
[34] Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*. Springer, 443–450.
[35] Zeyu Sun, Wenjie Zhang, Lili Mou, Qihao Zhu, Yingfei Xiong, and Lu Zhang. 2022. Generalized equivariance and preferential labeling for gnn node classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 8395–8403.
[36] Huayi Tang and Yong Liu. 2023. Towards understanding generalization of graph neural networks. In *International Conference on Machine Learning*. PMLR, 33674–33719.
[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[38] Hao Wang, Jiaxin Yang, and Jianrong Wang. 2021. Leverage large-scale biological networks to decipher the genetic basis of human diseases using machine learning. *Artificial Neural Networks* (2021), 229–248.
[39] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, and Bryan Hooi. 2021. Mixup for node and graph classification. In *Proceedings of the Web Conference 2021*. 3663–3674.
[40] Yu Wang, Yuying Zhao, Yi Zhang, and Tyler Derr. 2023. Collaboration-Aware Graph Convolutional Network for Recommender Systems. In *Proceedings of the ACM Web Conference 2023*.
[41] Dana Warmsley, Alex Waagen, Jiejun Xu, Zhining Liu, and Hanghang Tong. 2022. A survey of explainable graph neural networks for cyber malware analysis. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2932–2939.
[42] Hongzhi Wen, Jiayuan Ding, Wei Jin, Yiqi Wang, Yuying Xie, and Jiliang Tang. 2022. Graph neural networks for multimodal single-cell data integration. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 4153–4163.
[43] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. PMLR, 6861–6871.
[44] Qitian Wu, Yiting Chen, Chenxiao Yang, and Junchi Yan. 2023. Energy-based out-of-distribution detection for graph neural networks. *arXiv preprint arXiv:2302.02914* (2023).
[45] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. 2022. Handling Distribution Shifts on Graphs: An Invariance Perspective. In *International Conference on Learning Representations (ICLR)*.
[46] Ling Yang, Jiayi Zheng, Heyuan Wang, Zhongyi Liu, Zhilin Huang, Shenda Hong, Wentao Zhang, and Bin Cui. 2023. Individual and Structural Graph Information Bottlenecks for Out-of-Distribution Generalization. *IEEE Transactions on Knowledge and Data Engineering* (2023).
[47] Junchi Yu, Jian Liang, and Ran He. 2023. Mind the Label Shift of Augmentation-based Graph OOD Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11620–11630.
[48] Chongzhi Zhang, Mingyuan Zhang, Shanghang Zhang, Daisheng Jin, Qiang Zhou, Zhongang Cai, Haiyu Zhao, Xianglong Liu, and Ziwei Liu. 2022. Delving deep into the generalization of vision transformers under distribution shifts. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*. 7277–7286.
[49] Yanfu Zhang, Hongchang Gao, Jian Pei, and Heng Huang. 2022. Robust Self-Supervised Structural Graph Neural Network for Social Network Prediction. In *Proceedings of the ACM Web Conference 2022*. 1352–1361.
[50] Yanfu Zhang, Shangqian Gao, Jian Pei, and Heng Huang. 2022. Improving social network embedding via new second-order continuous graph neural networks. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 2515–2523.
[51] Tong Zhao, Wei Jin, Yozen Liu, Yingheng Wang, Gang Liu, Stephan Günnemann, Neil Shah, and Meng Jiang. 2022. Graph data augmentation for graph machine learning: A survey. *arXiv preprint arXiv:2202.08871* (2022).
[52] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. 2022. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*. PMLR, 27378–27394.
[53] Kaixiong Zhou, Xiao Huang, Daochen Zha, Rui Chen, Li Li, Soo-Hyun Choi, and Xia Hu. 2021. Dirichlet energy constrained learning for deep graph neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 21834–21846.
[54] Yangze Zhou, Gitta Kutyniok, and Bruno Ribeiro. 2022. OOD link prediction generalization capabilities of message-passing GNNs in larger test graphs. *Advances in Neural Information Processing Systems* 35 (2022), 20257–20272.
[55] Qi Zhu, Natalia Ponomareva, Jiawei Han, and Bryan Perozzi. 2021. Shift-robust gnns: Overcoming the limitations of localized graph training data. *Advances in Neural Information Processing Systems* 34 (2021), 27965–27977.
[56] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. 2021. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems* 34 (2021), 29476–29490.

A Appendices

A.1 Results of T-Test

To statistically validate these observations, we further applied T-Tests to the OOD results of both models on each dataset. The results are shown in Figure 4 and Figure 5.

A.2 Proof of Proposition 1

Proposition 1. *Given a node i with its feature vector x_i and its neighborhood $\mathcal{N}(i)$, the following aggregation scheme for obtaining its hidden representation $\mathbf{z}_i$,*

$$\mathbf{z}_i = \sum_{j \in \mathcal{N}(i)} \frac{\eta_i \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)}{\sum_{j \in \mathcal{N}(i)} \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)} \mathbf{x}_j,$$

with $\eta_i, \mathbf{W}_Q, \mathbf{W}_K$ being the learnable parameters, can be understood as an iterative process to optimize the objective in Eq. (6).

Proof. Given a distribution $X \sim \text{Gaussian}(X', \epsilon)$ with X as the observed input variable and X' as the clean target variable. Following [17], the information bottleneck principle involves minimizing the mutual information between the input X and its latent representation Z while still accurately predicting X' from Z. In the context of deep learning, the information bottleneck principle can be formulated as the following optimization objective:

$$f^*_{\text{IB}}(Z \mid X) = \arg \min_{f(Z|X)} I(X, Z) - I\left(Z, X'\right) \tag{11}$$

As demonstrated by Still et al. [31], we can utilize the information bottleneck to solve the clustering problems, in which nodes will be clustered into clusters with indices c. For simplicity, we take the 1-hop graph of node u to illustrate where the node indices are $1, 2, \ldots, |\mathcal{N}(u)|$. Following Strouse and Schwab [32], we assume that $p(i) = \frac{1}{n}$ with $n = |\mathcal{N}(u)|$ and $p(\mathbf{x}|i) \propto \exp\left[-\frac{1}{2\epsilon^2}||\mathbf{x} - \mathbf{x}_i||^2\right]$ with the introduction of a smoothing parameter ϵ.

We denote p_t as the probability distribution after the t-th iteration, and the iterative equation is given by [52]:

$$p_t(c|i) = \frac{\log p_{t-1}(c)}{Z(i)} \exp\left[-D_{\text{KL}}\left[p(\mathbf{x}|i)|p_{t-1}(\mathbf{x}|c)\right]\right] \tag{12}$$

$$p_t(c) = \frac{n_t^{(c)}}{n} \tag{13}$$

$$p_t(\mathbf{x}|c) = \frac{1}{n_t^{(c)}} \sum_{i \in S_t^{(c)}} p(\mathbf{x}|i), \tag{14}$$

where $Z(i)$ ensures normalization, $S_t^{(c)}$ represents the set of node indices in cluster c, and $n_t^{(c)} = |S_t^{(c)}|$ is the number of nodes assigned to cluster c. Then, we can approximate $p_{t-1}(\mathbf{x}|c)$ using a Gaussian distribution $q_{t-1}(\mathbf{x}|c) \sim \text{Gaussian}(\mu_{t-1}^{(c)}, \Sigma_{t-1}^{(c)})$. When the value of ϵ w.r.t. $p(\mathbf{x}|i)$ is sufficiently small, we have:

$$D_{\text{KL}}\left[p(\mathbf{x}|i)|q_{t-1}(\mathbf{x}|c)\right] \propto [\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i] + \log\det \Sigma_{t-1}^{(c)} + B, \tag{15}$$

where B represents terms that are independent of the assignment of data points to clusters and are consequently irrelevant to the objective. By substituting Eq. (15) back into Eq. (12), we can reformulate the cluster update as follows:

$$\begin{aligned} p_t(c|i) &= \frac{\log p_{t-1}(c)}{Z(i)} \frac{\exp\left[-[\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i]\right]}{\det \Sigma_{t-1}^{(c)}} \\ &= \frac{\log p_{t-1}(c)}{\det \Sigma_{t-1}^{(c)}} \frac{\exp\left[-[\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i]\right]}{\sum_c \exp\left[-[\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i]\right]}. \end{aligned} \tag{16}$$

To minimize the KL-divergence between $p_{t-1}(\mathbf{x}|c)$ and $q_{t-1}(\mathbf{x}|c)$, $\mu_c^{(t)}$ will be updated as:

$$\begin{aligned} \mu_c^{(t)} &= \frac{1}{n}\sum_{i=1}^{n} p_t(c|i)\mathbf{x}_i \\ &= \frac{1}{n}\sum_{i=1}^{n} \frac{\log p_{t-1}(c)}{\det \Sigma_{t-1}^{(c)}} \frac{\exp\left[-[\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i]\right]}{\sum_c \exp\left[-[\mu_{t-1}^{(c)} - \mathbf{x}_i]^\top [\Sigma_{t-1}^{(c)}]^{-1} [\mu_{t-1}^{(c)} - \mathbf{x}_i]\right]} \mathbf{x}_i \\ &= \sum_{i=1}^{n} \frac{\log p_{t-1}(c)}{n \det \Sigma_{t-1}} \frac{\exp\left[2 \cdot [\mu_{t-1}^{(c)}]^\top \Sigma_{t-1}^{-1} \mathbf{x}_i\right]}{\sum_c \exp\left[2 \cdot [\mu_{t-1}^{(c)}]^\top \Sigma_{t-1}^{-1} \mathbf{x}_i\right]} \mathbf{x}_i, \end{aligned} \tag{17}$$

where the last equation follows from the assumption that Σ_{t-1}^c is shared among all clusters and μ_c are normalized w.r.t. Σ_{t-1}^{-1}. Let $\mathbf{z}_c = \mu_c^{(t)}$,$\eta_c = \frac{\log p_{t-1}(c)}{n \det \Sigma_{t-1}}$, $2 \cdot \mu_{t-1}^{(c)} = \mathbf{W}_K \mathbf{x}_c$, $\mathbf{W}_Q = \Sigma_{t-1}^{-1}$ and rewrite the subscripts appropriately to obtain:

$$\mathbf{z}_i = \sum_{j \in \mathcal{N}(i)} \frac{\eta_i \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)}{\sum_{j \in \mathcal{N}(i)} \exp([\mathbf{W}_K \mathbf{x}_i]^\top \mathbf{W}_Q \mathbf{x}_j)} \mathbf{x}_j,$$

where η indicates attention correction weighting parameters, $\mathbf{W}_Q$ and $\mathbf{W}_K$ are transformation parameter about input features. □

This indicates that the graph self-attention mechanism follows the information bottleneck principle. Specifically, $\mu_c^{(t)}$ refers to the data distribution learned by the information bottleneck, and $\mathbf{z}_c$ is learned by the self-attention mechanism.

A.3 Dataset Details

We leverage the datasets from GOOD [11] and EERM datasets to evaluate our model. The statistics of EERM datasets are summarized in Table 5. These datasets exhibit various types of distribution shifts: Cora and Amz-Photo undergo artificial transformation shifts,

Table 5: Statistic information for datasets from EERM

Dataset	#Nodes	#Edges	#Classes
Cora	2,703	5,278	10
Amz-Photo	7,650	119,081	10
Twitch-E	$1,912 - 9,498$	$31,299 - 153,138$	2
Elliptic	203,769	$2,34,355$	2
OGB-Arxiv	169,343	$1,166,243$	40

Twitch-E involves Cross-Domain transfers, and Elliptic and OGB-Arxiv display temporal evolution.

A.4 Hyperparameter Selection

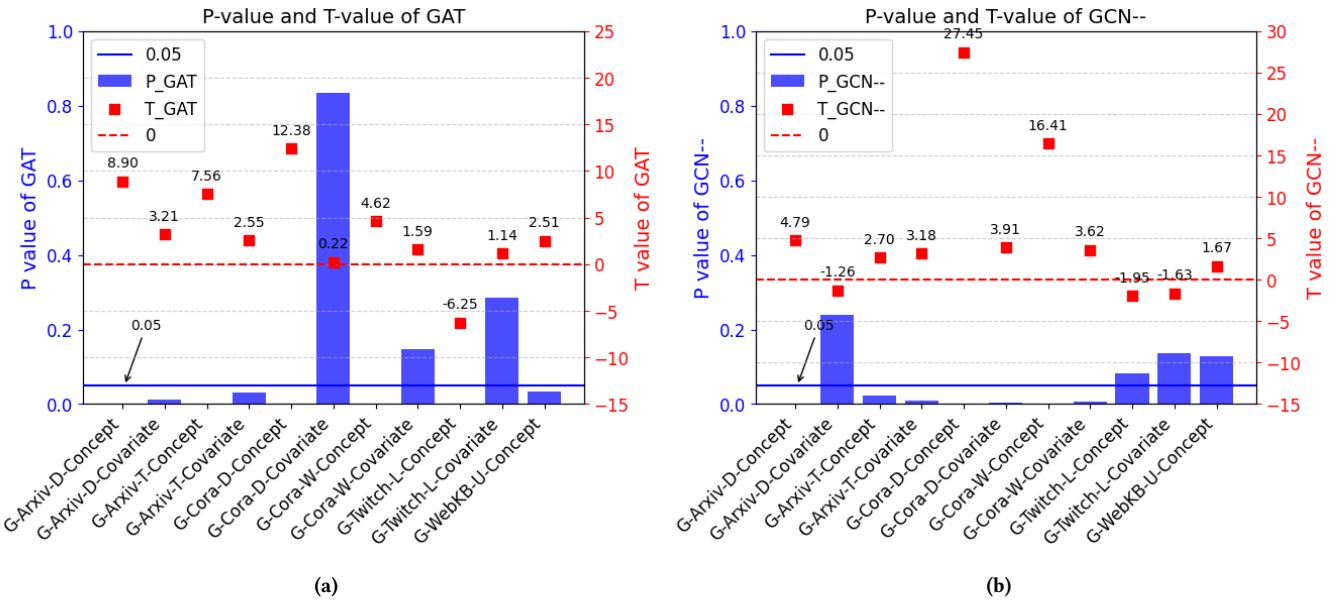
The fine-tuned parameters for each dataset in Section 3 are determined as follows:

- GOODCora-degree-covariate: lr=1e-3, dropout=0.5, hidden=200, model_layer=2
- GOODCora-degree-concept: lr=1e-3, dropout=0.5, hidden=200, model_layer=2
- GOODCora-word-covariate: lr=1e-3, dropout=0.5, hidden=300, model_layer=2
- GOODCora-word-concept: lr=1e-3, dropout=0.5, hidden=300, model_layer=1
- GOODArxiv-degree-covariate: lr=1e-3, dropout=0.2, hidden=300, model_layer=3
- GOODArxiv-degree-concept: lr=1e-3, dropout=0.2, hidden=300, model_layer=3
- GOODArxiv-time-covariate: lr=1e-3, dropout=0.2, hidden=300, model_layer=3
- GOODArxiv-time-concept: lr=1e-3, dropout=0.2, hidden=300, model_layer=3
- GOODTwitch-language-covariate: lr=1e-3, dropout=0.5, hidden=200, model_layer=2
- GOODTwitch-language-concept: lr=1e-3, dropout=0.5, hidden=300, model_layer=3
- GOODWebKB-university-concept: lr=5e-3, dropout=0.5, hidden=300, model_layer=1

The other experiment involves comparing our proposed model, DGAT, with various baseline models. We fine-tune the parameters within the following search space: layers (2, 3), dropout (0, 0.1, 0.2, 0.5), hidden (100, 200, 300), γ (0, 0.2, 0.5), β (0, 0.1, 0.2, 0.5), learning rate (1e-3, 5e-2, 5e-3), heads (2, 4). Taking the GOODCora-degree-Concept dataset as an example, the parameters used for GCN are: layers: 2, learning rate: 5e-3, dropout: 0.2, hidden: 300; the parameters used for GAT are: layers: 2, learning rate: 5e-3, dropout: 0.2, hidden: 300, heads: 2; the parameters used for APPNP are: layers: 2, learning rate: 5e-3, dropout: 0.2, hidden: 300, β: 0.2; the parameters used for DGAT are: layers: 2, learning rate: 5e-3, dropout: 0.2, hidden: 300, heads: 2, γ: 0.5, β: 0.2.

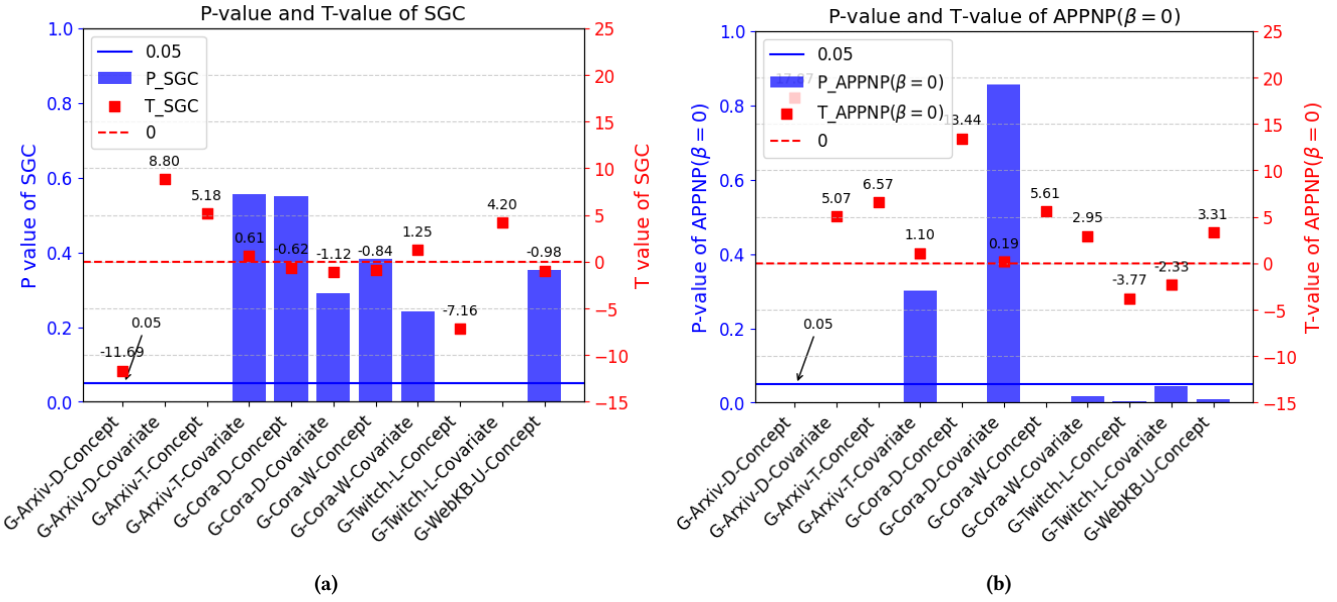
A.5 Ablation Study

We conducted the ablation study to analyze the impact of each component in DGAT. From Table 6, we find that each component contributes positively to the model.



(a) (b)

Figure 4: P_{value} and T_{value} of GAT and GCN–



(a) (b)

Figure 5: P_{value} and T_{value} of SGC and APPNP($\beta = 0$)

Table 6: Ablation study of DGAT on representative datasets on ERM setting. All numerical results are averages across 10 random runs.

	GOOD-Cora-D-Covariate	elliptic	OGB-Arxiv
DGAT	**59.68 ± 0.46**	**73.09 ± 2.14**	**45.51 ± 0.67**
DGAT w/o self-attention	59.19 ± 0.58	69.58 ± 2.08	45.29 ± 0.83
DGAT w/o decouple	58.25 ± 0.58	70.26 ± 1.46	40.44 ± 1.36
DGAT w/o remove linear classifier	57.52 ± 1.04	67.94 ± 2.63	44.79 ± 0.63