Enhancing Size Generalization in Graph Neural Networks through Disentangled Representation Learning

Zheng Huang [1] **Qihui Yang** [2] **Dawei Zhou** [3] **Yujun Yan** [1]

Abstract

Although most graph neural networks (GNNs) can operate on graphs of any size, their classification performance often declines on graphs larger than those encountered during training. Existing methods insufficiently address the removal of size information from graph representations, resulting in sub-optimal performance and reliance on backbone models. In response, we propose DISGEN, a novel and model-agnostic framework designed to disentangle size factors from graph representations. DISGEN employs size- and task-invariant augmentations and introduces a decoupling loss that minimizes shared information in hidden representations, with theoretical guarantees for its effectiveness. Our empirical results show that DISGEN outperforms the state-of-the-art models by up to 6% on real-world datasets, underscoring its effectiveness in enhancing the size generalizability of GNNs. Our codes are available at: https://github.com/GraphmindDartmouth/DISGEN.

1. Introduction

Graphs can exhibit significant variations in size, ranging from small molecular structures to large protein networks (Kearnes et al., 2016; Fout et al., 2017), and from concise code snippets with tens of nodes to extensive programs with thousands of nodes (Yan et al., 2020; Guo et al., 2020). Training Graph Neural Networks (GNNs) on these datasets is often limited to small graphs due to computational constraints or data availability (Zhang, 2019; Li et al., 2021a). However, these models are intended for application on larger graphs, necessitating effective size generalization to handle larger test graphs.

Despite the inherent capability of GNNs to process graphs of any size, performance declines are observed when models trained on smaller graphs are applied to substantially larger ones (Buffelli et al., 2022; Chen et al., 2022). Prior work to address this challenge involves strategies such as accessing test domain graphs (Yehudai et al., 2021), employing causal modeling for the graph generative process (Chen et al., 2022; Bevilacqua et al., 2021), and simulating size shifts within the training data through graph coarsening approaches (Buffelli et al., 2022). However, these methods inadequately remove the size information from the graph representations, resulting in sub-optimal performance and reliance on backbone models.

Motivated by recent studies (Yan et al., 2023; Bevilacqua et al., 2021) that reveal the correlation between learned graph representations and their size, we aim to tackle the size generalization problem through disentangled representation learning, a technique known for separating fundamental factors in observed data. However, directly applying existing disentangled representation learning methods (Sarhan et al., 2020; Creager et al., 2019) to our problem poses several challenges. Firstly, these methods typically rely on supervision to disentangle different information, necessitating supervision of graph size in our case. The discrete and unbounded nature of graph sizes, however, complicates their use as direct supervision labels. Secondly, it remains unclear how to minimize the shared information between size-related and task-related representations with theoretical guarantees. Current practices often use correlation loss (Mo et al., 2023) or enforce orthogonality of the representations (Sarhan et al., 2020) to segregate distinct information. However, there is limited theoretical analysis to substantiate their effectiveness, and in practice, these methods have proven suboptimal for our problem (Section 4.3).

To address these challenges, we propose a general **Dis**entangled representation learning framework for size **Gen**eralization (DISGEN) of GNNs. To tackle the first challenge, we introduce new augmentation strategies to guide the model in learning relative size information. Specifically, we create two views—size- and task-invariant views—for a given input graph and facilitate the learning of their rel-

[1]Department of Computer Science, Dartmouth College, Hanover, NH, USA. [2]Electrical and Computer Engineering, UCSD, San Diego, USA. [3]Department of Computer Science, Virginia Tech, Blacksburg, VA, USA. Correspondence to: Yujun Yan <yujun.yan@darmouth.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

ative size through a contrastive loss. To tackle the second challenge, we propose a decoupling loss to minimize the shared information between the hidden representations optimized for size- and task-related information, respectively. We further provide theoretical guarantees to justify the effectiveness of the decoupling loss.

Our contributions can be summarized as follows:

- **Novel model-agnostic framework:** To the best of our knowledge, DISGEN is the first disentangled representation learning framework to tackle the size generalization problem for GNNs.
- **Novel designs with theoretical guarantees:** We propose new augmentation strategies and novel decoupling loss to segment size- and task-related information. We also provide theoretical guarantees to justify the effectiveness of our proposed loss.
- **Extensive experiments:** Our empirical results show that DISGEN outperforms the state-of-the-art models by up to 6% on real-world datasets, highlighting its enhanced size generalizability for GNNs.

2. Preliminary

Notations. Consider a set of n graphs denoted by $\{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_n\}$, where each $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ represents the i-th graph with $N = |\mathcal{V}_i|$ nodes and $E = |\mathcal{E}_i|$ edges. We denote the neighborhood of a node v as $\mathcal{N}_v$, defined as $\mathcal{N}_v = \{u \mid (u, v) \in \mathcal{E}_i\}$. Furthermore, we denote the size-invariant and task-invariant views augmented from graph $\mathcal{G}_i$ as $\mathcal{G}_i^{(1)}$ and $\mathcal{G}_i^{(2)}$, respectively. As to the matrix representations, we use $\mathbf{X}_i \in \mathbb{R}^{N \times d_f}$ and $\mathbf{A}_i \in \mathbb{R}^{N \times N}$ to represent the feature matrix and adjacency matrix of $\mathcal{G}_i$, respectively, where d_f is the dimension of node features. Moreover, we use $\mathbf{h}_{g_i} \in \mathbb{R}^{d_g}$ to denote the representation of graph $\mathcal{G}_i$, where d_g is the dimension size. Additional matrix notations will be introduced as the paper progresses. Generally, we use a bold lowercase letter to denote a vector and a bold uppercase letter to denote a matrix used in our framework. In addition, superscripts are employed to denote matrices associated with augmented graphs, while subscripts are utilized to indicate matrices specific to a particular graph. We denote the entry at the (p, q) position of matrix $\mathbf{M}$ as $\mathbf{M}[p, q]$. Furthermore, for any multivariate function $\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) : (\mathbb{R}^c, \mathbb{R}^c) \mapsto \mathbb{R}^{d_g}$, we use the expression $\partial \mathbf{f} / \partial \tilde{\mathbf{s}} \equiv \mathbf{0}$ to indicate that for every component function f_i of the multivariate function $\mathbf{f}$, its partial derivative with respect to every component of the input vector $\tilde{\mathbf{s}}$ is always 0, i.e. $\partial f_i / \partial s_j \equiv 0$ for all i and j.

Problem Setup. In this paper, we define graph size as the number of nodes in the graph and study the size generalizability of GNNs (Chami et al., 2022; Wu et al., 2020; Maron et al., 2018). We define a GNN model as size generalizable if it demonstrates generalizability to test graphs with sizes larger than those in the training set. We focus on a supervised graph classification task, where each graph $\mathcal{G}_i$ is assigned a label y_i, and the goal is to learn a GNN model $f_\theta : (\mathbf{A}_i, \mathbf{X}_i) \mapsto y_i$ that maps each graph $\mathcal{G}_i$ to y_i. The graph classification objective is given by a `CrossEntropy` loss. Our goal is to design a framework, denoted by g, where $g \circ f_\theta(\mathbf{A}_i, \mathbf{X}_i)$ yields a more accurate estimation of y_i for graphs with sizes larger than those encountered during training.

Disentangled Representation Learning. Disentangled representation learning aims to separate and isolate the fundamental factors within observed data. By incorporating supervision or prior knowledge, this approach promotes the independence of different factors. This independence can be achieved through various methods, such as optimizing the Pearson's correlation coefficient to zero (Mo et al., 2023), enforcing orthogonality (Sarhan et al., 2020), or minimizing cosine similarity (Li et al.) of hidden representations.

Explainable GNN Model. Explainable GNN models (Ying et al., 2019; Luo et al., 2020; Li et al., 2023; Yan et al., 2019) are effective tools for understanding the predictions made by GNN models. Our focus is on perturbation-based methods (Yuan et al., 2022; Luo et al., 2020), which assess the importance scores of input graphs by monitoring changes in predictions resulting from different input perturbations. The intuition behind these approaches is that retaining task-relevant information in the inputs should result in predictions similar to the original ones. The explainable model takes as input a graph's adjacency matrix $\mathbf{A}_i$, feature matrix $\mathbf{X}_i$, its label y_i, and a trained GNN f_θ. The output is a weighted matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ that indicates the importance of edges, serving as the explanation for the given label.

3. Methodology

In this section, we present our framework DISGEN that utilizes disentangled representation learning to distinguish size- and task-related information in graph representations learned by GNN backbones.

To address the first challenge posed by the discrete and unbounded nature of graph sizes, we propose new augmentation strategies to guide the model in learning relative size information. Specifically, we create two views—size- and task-invariant views—for a given input graph and facilitate the learning of their relative sizes through a contrastive loss. To address the second challenge of minimizing the shared information to disentangle size- and task-related representations, we introduce a decoupling loss with theoretical guarantees. In this section, we first present an overview of the framework in Section 3.1, followed by a discussion on

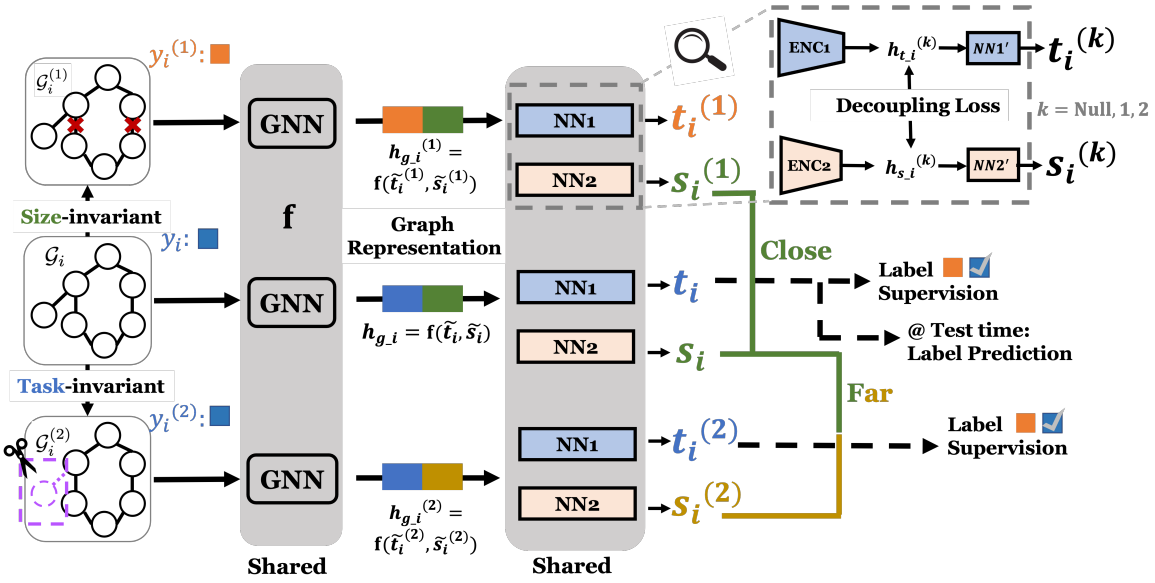

Figure 1. Framework overview: our model augments each graph $\mathcal{G}_i$ with size- and task-invariant views ($\mathcal{G}_i^{(1)}$ and $\mathcal{G}_i^{(2)}$), which, along with the original graph, are processed by a shared GNN backbone. Two encoders then generate size- ($\mathbf{s}_i$) and task-related ($\mathbf{t}_i$) representations, respectively. A contrastive loss on size-related representations guides relative size learning, while a decoupling loss ensures the separation of size- and task-related information.

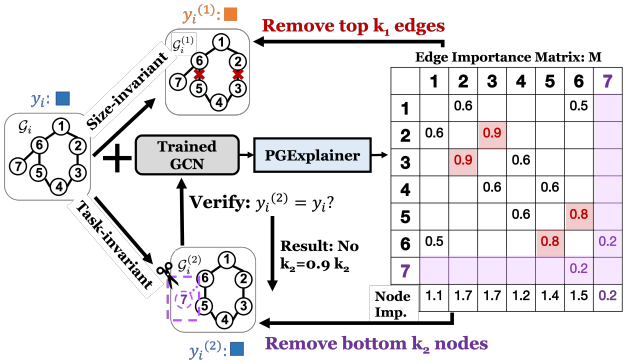

Figure 2. Augmentation overview: view $\mathcal{G}_i^{(1)}$ is generated by removing edges that most significantly change the label information, while $\mathcal{G}_i^{(2)}$ results from eliminating nodes that have little impact on the model predictions.

learning size-related information through graph augmentations in Section 3.2. Lastly, we introduce the decoupling loss and discuss its theoretical guarantees in Section 3.3

3.1. Framework Overview

As illustrated in Figure 1, DISGEN initially augments each graph $\mathcal{G}_i$ with two views—size- and task-invariant views ($\mathcal{G}_i^{(1)}$ and $\mathcal{G}_i^{(2)}$). These augmented views, along with the original graph, are then fed into the shared GNN backbone. Subsequently, two encoders are employed to encode the output graph representations into size- and task-related representations ($\mathbf{s}_i$ and $\mathbf{t}_i$), respectively. The task-related representations of both the original and task-invariant graphs ($\mathbf{t}_i$ and $\mathbf{t}_i^{(2)}$) are supervised by their shared labels. A contrastive loss is then applied to the size-related representations of the original graphs and their augmented views ($\mathbf{s}_i, \mathbf{s}_i^{(1)}$ and $\mathbf{s}_i^{(2)}$), aiming to guide the model in learning their relative sizes. Additionally, to ensure the separation of size- and task-related information, a decoupling loss is applied to the hidden representations responsible for generating these representations. In Figure 1, the hidden representations for size- and task-related information are denoted by $\mathbf{h}_{t_i}^{(k)}$ and $\mathbf{h}_{s_i}^{(k)}$, respectively, where i is the graph index and k specifies different views. Finally, during test time, we use the task-related representations $\mathbf{t}_i$ to predict the label for the graph $\mathcal{G}_i$.

3.2. Augmentation

In this subsection, we present the augmentation strategies employed to generate the size- and task-invariant views, as shown in Figure 2. The size-invariant view $\mathcal{G}_i^{(1)}$ maintains the same number of nodes as $\mathcal{G}$ but possesses distinct labels. Conversely, the task-invariant view $\mathcal{G}_i^{(2)}$ shares the same label as $\mathcal{G}$, but differs in the number of nodes. In more detail, view $\mathcal{G}_i^{(1)}$ is generated by selectively removing edges that most significantly change the label information, whereas view $\mathcal{G}_i^{(2)}$ is formed by eliminating nodes that have little impact on the label information. One key challenge in the augmentation process is determining whether the label

information has been altered. To address this, we employ a pre-trained GNN f_θ to monitor the change. We first pre-train a backbone GNN on the same training graphs, and consider the augmented graph as changing the label information if and only if the predictions from the pre-trained GNN for the augmented graph change.

To identify critical edges that influence label information and insignificant nodes with minimal impact, we utilize the edge importance matrix derived from graph explainable models (Luo et al., 2020; Ying et al., 2019). In more detail, the explainable model takes a graph $\mathcal{G}_i$, its corresponding label y_i, and a trained GNN model f_θ, as the inputs, and outputs an edge importance matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$, where the entry $\mathbf{M}[i,j]$ indicates the importance score of edge $e_{ij} \in \mathcal{E}_i$. This score measures the influence on the graph label prediction when removing the edge e_{ij}—the higher the score, the stronger the influence. As we focus on the undirected graphs, we symmetrize the matrix by: $\hat{\mathbf{M}} = \frac{1}{2}(\mathbf{M}+\mathbf{M}^T)$. To generate the size-invariant view, we remove k_1 edges with highest importance scores from the original graph $\mathcal{G}_i$, where k_1 is a predefined hyperparameter. To generate the task-invariant view, we compute the node importance scores and remove k_2 nodes with the lowest scores from the original graph, along with the edges connecting them, where k_2 is a predefined hyperparameter. We define the node importance scores based on the importance scores of the edges incident to the node. Mathematically, the importance score m_{v_j} of a node v_j is defined as: $m_{v_j} = \sum_{k \in \mathcal{N}_{v_j}} \hat{\mathbf{M}}[j,k]$. To verify that the task-invariant view $\mathcal{G}_i^{(2)}$ maintains the same label as the original graph, we feed it to the pre-trained GNN f_θ, and test if the following holds: $f_\theta(\mathcal{G}) = f_\theta(\mathcal{G}_i^{(2)})$. If the condition is not met, we reduce k_2 to $0.9k_2$ and repeat the above process.

The views $\mathcal{G}_i^{(1)}$ and $\mathcal{G}_i^{(2)}$ are then utilized to guide our framework to learn the relative size information. Inspired by contrastive learning (Mo et al., 2023; Li et al., 2021b), we modify the contrastive loss to convey to the model that the size-invariant view maintains the same size as the original graph while the task-invariant view has a different size. Specifically, we aim for the learned size representations of $\mathcal{G}_i$ and $\mathcal{G}_i^{(1)}$ ($\mathbf{s}_i$ and $\mathbf{s}_i^{(1)}$) to be close to each other as the two views share the same graph size. Conversely, we expect the size representations of $\mathcal{G}_i$ and $\mathcal{G}_i^{(2)}$ ($\mathbf{s}_i$ and $\mathbf{s}_i^{(2)}$) to be far away from each other, reflecting their distinct graph sizes. If we denote c_1 (c_2) as the cosine similarity between $\mathbf{s}_i$ and $\mathbf{s}_i^{(1)}$ ($\mathbf{s}_i^{(2)}$), the contrastive loss is given by:

$$\mathcal{L}_s = -log\left(\frac{exp(c_1)/\tau}{exp(c_1)/\tau + exp(c_2)/\tau}\right),$$

where τ is a hyperparameter.

In addition to the contrastive loss, we also provide supervision on the task-related representations $\mathbf{t}_i$ and $\mathbf{t}_i^{(2)}$ to encourage the learning of task information, where $\mathbf{t}_i$ and $\mathbf{t}_i^{(2)}$ share the same label. The supervision loss is given by:

$$\mathcal{L}_t = -\alpha_1 \texttt{CE}(y_i, \mathbf{t}_i) - \alpha_2 \texttt{CE}(y_i, \mathbf{t}_i^{(2)}),$$

where CE denotes the `CrossEntropy` loss, α_1 and α_2 are hyperparameters for regularization.

3.3. Decoupling Loss

3.3.1. Design

In this subsection, we introduce the decoupling loss, which aims to minimize the shared information between the hidden representations optimized for size- and task-related information. We denote the size- and task-related hidden representations of a graph as $\mathbf{h_s} \in \mathbb{R}^{d_h}$ and $\mathbf{h_t} \in \mathbb{R}^{d_h}$, respectively. The size-related hidden representations of all original and augmented graphs in a batch of size b are stacked into a matrix $\mathbf{H}_s = \left[\mathbf{h}_{s_1}, \mathbf{h}_{s_1}^{(1)}, \mathbf{h}_{s_1}^{(2)}, ..., \mathbf{h}_{s_b}, \mathbf{h}_{s_b}^{(1)}, \mathbf{h}_{s_b}^{(2)}\right]^T$. Similarly, the matrix of stacked task-related hidden representations in a batch is denoted by $\mathbf{H}_t = \left[\mathbf{h}_{t_1}, \mathbf{h}_{t_1}^{(1)}, \mathbf{h}_{t_1}^{(2)}, ..., \mathbf{h}_{t_b}, \mathbf{h}_{t_b}^{(1)}, \mathbf{h}_{t_b}^{(2)}\right]^T$. Our rationale is that if $\mathbf{H}_s$ and $\mathbf{H}_t$ contain the same information, there exists a function $f_p \in \mathcal{F}$ that can transform one representation to another: $\mathbf{H}_s = f_p(\mathbf{H}_t)$. When we restrict the function class $\mathcal{F}$ to linear projection functions, the substantial similarity in information between $\mathbf{H}_s$ and $\mathbf{H}_t$ suggests the existence of a linear projection $\mathbf{P}$ such that the difference $\mathbf{H}_t\mathbf{P} - \mathbf{H}_s$ is sufficiently small. For any given $\mathbf{H}_s$ and $\mathbf{H}_t$, we can find an optimal projection plane $\mathbf{P}_{opt}$ that minimizes the mapping residual from $\mathbf{H}_t$ to $\mathbf{H}_s$:

$$\mathbf{P}_{opt} = \min_{\mathbf{P}} \left\| \mathbf{H}_t\mathbf{P} - \mathbf{H}_s \right\|_F, \tag{1}$$

where $\| \cdot \|_F$ denotes the Frobenius norm of a matrix. Let D denote the residual under optimal linear projection: $D = \|\mathbf{H}_t\mathbf{P}_{opt} - \mathbf{H}_s\|_F$. We can use D to quantify the shared information between $\mathbf{H}_s$ and $\mathbf{H}_t$. A small value of D indicates a substantial overlap in information between $\mathbf{H}_s$ and $\mathbf{H}_t$, while a large D suggests minimal shared information. Recall that our goal is to minimize the shared information, thus we aim to train our framework such that $\mathbf{H}_s$ and $\mathbf{H}_t$ satisfy:

$$\max_{\mathbf{H}_t, \mathbf{H}_s} \min_{\mathbf{P}} \left\| \mathbf{H}_t\mathbf{P} - \mathbf{H}_s \right\|_F, \tag{2}$$

The best linear projection plane $\mathbf{P}_{opt}$ can be obtained by setting the derivative of Equation (1) with respect to $\mathbf{P}$ to zero. Thus $\mathbf{P}_{opt}$ is given by:

$$\mathbf{P}_{opt} = (\mathbf{H}_t^T\mathbf{H}_t)^{-1}\mathbf{H}_t^T\mathbf{H}_s. \tag{3}$$

As a result, we denote our decoupling loss as:

$$\mathcal{L}_d = \frac{1}{D^2}, \quad D = \left\| \mathbf{H}_t \mathbf{P}_{opt} - \mathbf{H}_s \right\|_F. \tag{4}$$

While the computation of $\mathbf{P}_{opt}$ requires inversion, it is noteworthy that in practice, the matrix size for inversion is small, resulting in negligible slowdowns in training time.

After decoupling $\mathbf{h}_t$ and $\mathbf{h}_s$, a prediction $\mathbf{t}$ is made by a neural network, as well as $\mathbf{s}$. As a result, the objective function of our framework is: $\mathcal{L} = \beta_1 \mathcal{L}_s + \beta_2 \mathcal{L}_t + \beta_3 \mathcal{L}_d$, where β_1, β_2 and β_3 are weights of different loss functions.

3.3.2. Theoretical Analysis

In this subsection, we provide a theoretical analysis to show the effectiveness of the decoupling loss. We begin with the assumptions and definitions. We assume that $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{t}} \in \mathbb{R}^c$ are unknown ground truth vectors for graph $\mathcal{G}$. The parameter c denotes the minimum dimensionality required for $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{t}}$ to exclusively encode size-related and task-related information, respectively. We further assume that the graph representation $\mathbf{h}_g$ is a function of $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{t}}$, i.e. $\mathbf{h}_g = \mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}})$. Next, we formally define decoupling as follows:

Definition 3.1. We say that $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ can be decoupled from $\mathbf{f}$ by $\mathbf{g}_1$ and $\mathbf{g}_2$, if given $\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) : (\mathbb{R}^c, \mathbb{R}^c) \mapsto \mathbb{R}^{d_g}$, $\exists\, \mathbf{g}_1, \mathbf{g}_2 : \mathbb{R}^{d_g} \mapsto \mathbb{R}^{d_h}$, satisfying that

$$\frac{\partial \mathbf{g}_1(\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}))}{\partial \tilde{\mathbf{s}}} \equiv \mathbf{0},$$

and

$$\frac{\partial \mathbf{g}_2(\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}))}{\partial \tilde{\mathbf{t}}} \equiv \mathbf{0}.$$

Note that if $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ can be decoupled from $\mathbf{f}$, then the following equations hold:

$$\mathbf{g}_1(\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}})) = \mathbf{w}_1(\tilde{\mathbf{t}}),$$

$$\mathbf{g}_2(\mathbf{f}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}})) = \mathbf{w}_2(\tilde{\mathbf{s}}),$$

for some functions $\mathbf{w}_1$ and $\mathbf{w}_2$.

Our goal is to show the connection between maximizing D (Equation (4)) and decoupling $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ from $\mathbf{f}$. Before presenting the rationale, we first show that $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ cannot be decoupled from $\mathbf{f}$ if D is sufficiently small. For simplicity, we may express the composite function $\text{ENC}_i \circ \mathbf{f}(\cdot, \cdot)$ as a function of $\mathbf{r} \in \mathbb{R}^{2c}$, i.e., $\text{ENC}_i \circ \mathbf{f}(\cdot) : \mathbb{R}^{2c} \mapsto \mathbb{R}^{d_h}$, where $\mathbf{r}$ is obtained by stacking $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$.

Theorem 3.2. *Consider the composite functions* $\text{ENC}_i \circ \mathbf{f}(\cdot)$, $i \in 1, 2$, *defined on a closed set* $S \in \mathbb{R}^{2c}$. *Assume that these composite functions are twice differentiable at some point* $\mathbf{r}_0$, *and the gradients* $\nabla \mathbf{h}_t$ *and* $\nabla \mathbf{h}_s$ *at* $\mathbf{r}_0$ *are nonzero matrices. Furthermore, assume that the dimension* c *of* $\tilde{\mathbf{t}}$ $(\tilde{\mathbf{s}})$ *and the dimension* d_h *of* $\mathbf{h}_t$ $(\mathbf{h}_s)$ *satisfy the condition:* $d_h \geq 2c + 1$. *Then* $\forall \mathbf{r} \in S$, $\exists \mathbf{P}$ *of full rank, and some constants* ι_1, ι_2, *such that:*

$$\mathbf{h}_t(\mathbf{r})\mathbf{P} = \mathbf{h}_s(\mathbf{r}) + O(\|\mathbf{r} - \mathbf{r}_0\|^2),\ \iota_1 < \lim_{\mathbf{r} \to \mathbf{r}_0} \left\| \frac{O(\|\mathbf{r} - \mathbf{r}_0\|^2)}{\|\mathbf{r} - \mathbf{r}_0\|^2} \right\| < \iota_2 \tag{5}$$

$\Rightarrow \tilde{\mathbf{t}}$ *and* $\tilde{\mathbf{s}}$ *can not be decoupled from* $\mathbf{f}(\cdot, \cdot)$ *by the* $\mathbf{ENC}_i(\cdot)$.

Proof. The hidden representations $\mathbf{h}_t$ and $\mathbf{h}_s$ are outputs of the encoders, which are functions of $\mathbf{r}$:

$$\begin{aligned} \mathbf{h}_t(\mathbf{r}) =& \mathbf{h}_t(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) = \mathbf{ENC}_1 \circ \mathbf{f}(\mathbf{r}), \\ \mathbf{h}_s(\mathbf{r}) =& \mathbf{h}_s(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) = \mathbf{ENC}_2 \circ \mathbf{f}(\mathbf{r}). \end{aligned} \tag{6}$$

We further expand each composite function into a set of functions $h_{t_i}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}})$ and $h_{s_i}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) : \mathbb{R}^{2c} \mapsto \mathbb{R}$

$$\begin{aligned} \mathbf{h}_t(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) &= \begin{bmatrix} h_{t_1}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) & \cdots & h_{t_d_h}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) \end{bmatrix} \\ &= \begin{bmatrix} h_{t_1}(\mathbf{r}) & \cdots & h_{t_d_h}(\mathbf{r}) \end{bmatrix} = \mathbf{h}_t(\mathbf{r}), \\ \mathbf{h}_s(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) &= \begin{bmatrix} h_{s_1}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) & \cdots & h_{s_d_h}(\tilde{\mathbf{t}}, \tilde{\mathbf{s}}) \end{bmatrix} \\ &= \begin{bmatrix} h_{s_1}(\mathbf{r}) & \cdots & h_{s_d_h}(\mathbf{r}) \end{bmatrix} = \mathbf{h}_s(\mathbf{r}). \end{aligned} \tag{7}$$

We give the proof by contradiction.

We assume that $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ can be decoupled from $\mathbf{f}$ by the $\mathbf{ENC}_i(\cdot)$ function, then:

$$\begin{aligned} \mathbf{h}_s(\tilde{\mathbf{s}}) &= \begin{bmatrix} h_{s_1}(\tilde{\mathbf{s}}) & \cdots & h_{s_d_h}(\tilde{\mathbf{s}}) \end{bmatrix}, \\ \mathbf{h}_t(\tilde{\mathbf{t}}) &= \begin{bmatrix} h_{t_1}(\tilde{\mathbf{t}}) & \cdots & h_{t_d_h}(\tilde{\mathbf{t}}) \end{bmatrix}. \end{aligned} \tag{8}$$

Using Equation (5), we obtain the following expression:

$$\begin{bmatrix} h_{t_1}(\tilde{\mathbf{t}}) & \cdots & h_{t_d_h}(\tilde{\mathbf{t}}) \end{bmatrix} \mathbf{P}_i = h_{s_i}(\tilde{\mathbf{s}}) + O(\|\mathbf{r} - \mathbf{r}_0\|^2), \tag{9}$$

where $\mathbf{P}_i$ stands for the i-th column of $\mathbf{P}$. Taking the partial derivatives with respect to every component $\tilde{s}_j$ of $\tilde{\mathbf{s}}$ on both sides of the equation, we get:

$$\begin{aligned} \frac{\partial h_{s_i}(\tilde{\mathbf{s}})}{\partial \tilde{s}_j} &= \begin{bmatrix} \frac{\partial h_{t_1}(\tilde{\mathbf{t}})}{\partial \tilde{s}_j} & \cdots & \frac{\partial h_{t_d_h}(\tilde{\mathbf{t}})}{\partial \tilde{s}_j} \end{bmatrix} \mathbf{P}_i - O(\|\mathbf{r} - \mathbf{r}_0\|) \\ &= \begin{bmatrix} 0 & \cdots & 0 \end{bmatrix} \mathbf{P}_i - O(\|\mathbf{r} - \mathbf{r}_0\|). \end{aligned} \tag{10}$$

When $\mathbf{r} \to \mathbf{r}_0$, $O(\|\mathbf{r} - \mathbf{r}_0\|) \to \mathbf{0}$. Thus:

$$\frac{\partial h_{s_i}(\tilde{\mathbf{s}})}{\partial \tilde{s}_j}\Big|_{\mathbf{r}_0} = 0, \text{ where } i \in \{1, \ldots, d_h\}, j \in \{1, \ldots, c\}. \tag{11}$$

Similarly, taking the partial derivatives w.r.t. any $\tilde{t}_j$ on both sides of the equation yields:

$$\begin{bmatrix} \frac{\partial h_{s_1}(\tilde{\mathbf{s}})}{\partial \tilde{t}_1} & \cdots & \frac{\partial h_{s_d_h}(\tilde{\mathbf{s}})}{\partial \tilde{t}_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial h_{s_1}(\tilde{\mathbf{s}})}{\partial \tilde{t}_c} & \cdots & \frac{\partial h_{s_d_h}(\tilde{\mathbf{s}})}{\partial \tilde{t}_c} \end{bmatrix} = \begin{bmatrix} \frac{\partial h_{t_1}(\tilde{\mathbf{t}})}{\partial \tilde{t}_1} & \cdots & \frac{\partial h_{t_d_h}(\tilde{\mathbf{t}})}{\partial \tilde{t}_1} \\ \cdots & \cdots & \cdots \\ \frac{\partial h_{t_1}(\tilde{\mathbf{t}})}{\partial \tilde{t}_c} & \cdots & \frac{\partial h_{t_d_h}(\tilde{\mathbf{t}})}{\partial \tilde{t}_c} \end{bmatrix} \mathbf{P} - O(\|\mathbf{r} - \mathbf{r}_0\|) = \mathbf{0}. \tag{12}$$

Since $\mathbf{P}$ is a matrix of full rank, when $\mathbf{r} \to \mathbf{r}_0$, $O(\|\mathbf{r} - \mathbf{r}_0\|) \to \mathbf{0}$. Thus we have:

$$\frac{\partial h_{t_i}(\tilde{\mathbf{t}})}{\partial t_j}|_{\mathbf{r}_0} = 0, \text{ where } i \in \{1, \dots, d_h\}, j \in \{1, \dots, c\} \tag{13}$$

Considering Equation (11) and Equation (13), it can be concluded that $\nabla \mathbf{h}_t$ and $\nabla \mathbf{h}_s$ at $\mathbf{r}_0$ are zero matrices. This result contradicts our initial assumptions. Thus, $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ can not be decoupled from $\mathbf{f}$ by the $\mathbf{ENC}_i(\cdot)$ functions. □

Theorem 3.2 provides a sufficient condition for $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ not to be decoupled from $\mathbf{f}$ by $\mathbf{ENC}_i(\cdot)$. Next, we will present a necessary condition with an additional constraint, beyond those specified in Theorem 3.2.We begin by defining two matrices at $\boldsymbol{r}_0$:

$$\mathbf{B} = \begin{bmatrix} h_{s_1}(\boldsymbol{r}_0) & \cdots & h_{s_d_h}(\boldsymbol{r}_0) \\ \frac{\partial h_{s_1}(\boldsymbol{r}_0)}{\partial \tilde{t}_1} & \cdots & \frac{\partial h_{s_d_h}(\boldsymbol{r}_0)}{\partial \tilde{t}_1} \\ \cdots & & \cdots \\ \frac{\partial h_{s_1}(\boldsymbol{r}_0)}{\partial \tilde{t}_c} & \cdots & \frac{\partial h_{s_d_h}(\boldsymbol{r}_0)}{\partial \tilde{t}_c} \\ \frac{\partial h_{s_1}(\boldsymbol{r}_0)}{\partial \tilde{s}_1} & \cdots & \frac{\partial h_{s_d_h}(\boldsymbol{r}_0)}{\partial \tilde{s}_1} \\ \cdots & & \cdots \\ \frac{\partial h_{s_1}(\boldsymbol{r}_0)}{\partial \tilde{s}_c} & \cdots & \frac{\partial h_{s_d_h}(\boldsymbol{r}_0)}{\partial \tilde{s}_c} \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} h_{t_1}(\boldsymbol{r}_0) & \cdots & h_{t_d_h}(\boldsymbol{r}_0) \\ \frac{\partial h_{t_1}(\boldsymbol{r}_0)}{\partial \tilde{t}_1} & \cdots & \frac{\partial h_{t_d_h}(\boldsymbol{r}_0)}{\partial \tilde{t}_1} \\ \cdots & & \cdots \\ \frac{\partial h_{t_1}(\boldsymbol{r}_0)}{\partial \tilde{t}_c} & \cdots & \frac{\partial h_{t_d_h}(\boldsymbol{r}_0)}{\partial \tilde{t}_c} \\ \frac{\partial h_{t_1}(\boldsymbol{r}_0)}{\partial \tilde{s}_1} & \cdots & \frac{\partial h_{t_d_h}(\boldsymbol{r}_0)}{\partial \tilde{s}_1} \\ \cdots & & \cdots \\ \frac{\partial h_{t_1}(\boldsymbol{r}_0)}{\partial \tilde{s}_c} & \cdots & \frac{\partial h_{t_d_h}(\boldsymbol{r}_0)}{\partial \tilde{s}_c} \end{bmatrix}.$$

Theorem 3.3. *Given the conditions in Theorem 3.2, we assume that the matrices $\boldsymbol{B}$ and $\boldsymbol{C}$, defined at $\mathbf{r}_0$ satisfy that $\exists \boldsymbol{C}^+$, such that $\boldsymbol{C}\boldsymbol{C}^+\boldsymbol{B} = \boldsymbol{B}$. Then $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ can not be decoupled from $\mathbf{f}(\cdot,\cdot)$ by the* $\mathbf{ENC}_i(\cdot)$ functions
$\Rightarrow \forall \mathbf{r} \in S$, $\exists \mathbf{P}$ and some constants ι_1, ι_2,

$$\mathbf{h}_t(\mathbf{r})\mathbf{P} = \mathbf{h}_s(\mathbf{r}) + O(\|\mathbf{r} - \mathbf{r}_0\|^2),\ \iota_1 < \lim_{\mathbf{r}\to\mathbf{r}_0} \|\frac{O(\|\mathbf{r}-\mathbf{r}_0\|^2)}{\|\mathbf{r}-\mathbf{r}_0\|^2}\| < \iota_2.$$

Proof. Since $\mathbf{h}_t(\mathbf{r})$ and $\mathbf{h}_s(\mathbf{r})$ are twice differentiable at $\mathbf{r}_0$, we can expand $\mathbf{h}_t(\mathbf{r})$ and $\mathbf{h}_s(\mathbf{r})$ into Taylor series around $\mathbf{r}_0 \in S$. For $i, j \in \{1, \dots, d_h\}$,

$$h_{s_i}(\mathbf{r}) = a_{i_0} + (\mathbf{r} - \mathbf{r}_0)^\top \mathbf{a}_{i_1} + O(\|\mathbf{r} - \mathbf{r}_0\|^2),$$

$$h_{t_j}(\mathbf{r}) = b_{j_0} + (\mathbf{r} - \mathbf{r}_0)^\top \mathbf{b}_{j_1} + O(\|\mathbf{r} - \mathbf{r}_0\|^2),$$

where $a_{i_0} = h_{s_i}(\mathbf{r}_0)$, $b_{j_0} = h_{t_j}(\mathbf{r}_0)$, and $\mathbf{a}_{i_1}$ and $\mathbf{b}_{j_1}$ are gradient vectors.

We find the matrix $\mathbf{P}$ by solving the following equations:

$$\begin{bmatrix} a_{1_0} & \cdots & a_{d_h_0} \\ \mathbf{a}_{1_1} & \cdots & \mathbf{a}_{d_h_1} \end{bmatrix} = \begin{bmatrix} b_{1_0} & \cdots & b_{d_h_0} \\ \mathbf{b}_{1_1} & \cdots & \mathbf{b}_{d_h_1} \end{bmatrix} \mathbf{P} \tag{14}$$
$$\Leftrightarrow \boldsymbol{B} = \boldsymbol{C}\mathbf{P}$$

Note that there are $d_h + 2cd_h$ constraints, whereas $\mathbf{P}$ has d_h^2 degrees of freedom. Using Lemma A.1 and the condition $\boldsymbol{C}\boldsymbol{C}^+\boldsymbol{B} = \boldsymbol{B}$ and $d_h \geq 2c + 1$, it follows that the Equation (14) has a feasible solution. When the requirements in Equation (14) are met, the first two terms in the Taylor series of $\mathbf{h}_t(\mathbf{r})\mathbf{P}$ and $\mathbf{h}_s(\mathbf{r})$ are equal. Thus, $\mathbf{h}_t(\mathbf{r})\mathbf{P} = \mathbf{h}_s(\mathbf{r}) + O(\|\mathbf{r} - \mathbf{r}_0\|^2)$. □

Next we provide a theoretic rationale for maximizing D in Equation (4). We assume that $\tilde{\mathbf{t}}$ and $\tilde{\mathbf{s}}$ are uniformly sampled from their domain S. Thus, we have:

$$\left\|\mathbf{H}_t\mathbf{P}_{opt} - \mathbf{H}_s\right\|_F = 3b\int_S \frac{\|\mathbf{y}(\mathbf{r})\|}{V_S} d\mathbf{r},\ \mathbf{y}(\mathbf{r}) = \mathbf{h}_t(\mathbf{r})\mathbf{P}_{opt} - \mathbf{h}_s(\mathbf{r}), \tag{15}$$

where $V_S = \int_S d\mathbf{r}$, b is the batch size and three is the number of views. In the following, we prove that if $\mathbf{h}_t$, $\mathbf{h}_s$ and $\mathbf{P}_{opt}$ satisfy Equation (5), there exists a universal upper bound for Equation (15) applicable to all $O(\|\mathbf{r} - \mathbf{r}_0\|^2)$ that meet certain criteria. Since our framework maximizes D, the decoupling of $\mathbf{h}_t$ and $\mathbf{h}_s$ from $\mathbf{f}$ may be achieved when the maximized D surpasses the upper bound.

Definition 3.4. Consider a Jordan-measurable set S' and arbitrary constants ρ_1 and ρ_2, where $0 < \rho_1 < \rho_2$. We define $\Omega^{S'}_{\rho_1,\rho_2,M_0}$ as the set of functions $\mathbf{y}(\mathbf{r}) = O(\|\mathbf{r} - \mathbf{r}_0\|^2)$ that satisfy the following conditions: $\exists M_0 > 0$, $3b\int_{S\setminus S'} \frac{\|\mathbf{y}(\mathbf{r})\|}{V_S} d\mathbf{r} < M_0$ and $\rho_1\|\mathbf{r} - \mathbf{r}_0\|^2 < \|\mathbf{y}(\mathbf{r})\| < \rho_2\|\mathbf{r} - \mathbf{r}_0\|^2$ on S'.

Theorem 3.5. *If $\exists\, \mathbf{g}_1, \mathbf{g}_2 : \mathbb{R}^{d_g} \mapsto \mathbb{R}^{d_h}$ that can decouple the graph representation function $\mathbf{f}$, $\exists\, U^{S'}_{\rho_1,\rho_2,M_0} \in \mathbb{R}$, which is the universal upper bound for any $\mathbf{y}(\mathbf{r}) \in \Omega^{S'}_{\rho_1,\rho_2,M_0}$.*

Proof. $\forall \varepsilon_0 > 0$, we define $S_{\varepsilon_0} = \{\mathbf{r} \mid \|\mathbf{r} - \mathbf{r}_0\| < \frac{\varepsilon_0}{\rho_2}\}$. $\forall \mathbf{r} \in S_{\varepsilon_0}$, we have:

$$\begin{aligned} \rho_2\|\mathbf{r} - \mathbf{r}_0\| &< \varepsilon_0 \\ \|\mathbf{y}(\mathbf{r})\| < \rho_2\|\mathbf{r} - \mathbf{r}_0\|^2 &< \varepsilon_0\|\mathbf{r} - \mathbf{r}_0\|. \end{aligned} \tag{16}$$

Thus,

$$\frac{\|\mathbf{y}(\mathbf{r})\|}{\|\mathbf{r} - \mathbf{r}_0\|} < \varepsilon_0 \tag{17}$$

Since S' is a Jordan-measurable set, $\exists \varepsilon_1$, such that $S_{\varepsilon_1} \supseteq S'$. Assume that $\varepsilon_0 < \varepsilon_1$. Then $\forall \mathbf{r} \in S' \setminus S_{\varepsilon_0}$, we have:

$$\frac{\|\mathbf{y}(\mathbf{r})\|}{\|\mathbf{r} - \mathbf{r}_0\|} < \varepsilon_1 \tag{18}$$

Referring to 15, we have:

$$
\begin{aligned}
&\left\|\mathbf{H}_t\mathbf{P}_{opt}-\mathbf{H}_s\right\|_F \\
=&3b\int_{S_{\varepsilon_0}}\frac{\|\mathbf{y}(\mathbf{r})\|}{V_S}d\mathbf{r}+3b\int_{S'\setminus S_{\varepsilon_0}}\frac{\|\mathbf{y}(\mathbf{r})\|}{V_S}d\mathbf{r} \\
&+3b\int_{S\setminus S'}\frac{\|\mathbf{y}(\mathbf{r})\|}{V_S}d\mathbf{r} \\
<&\frac{3b}{V_S}\int_{S_{\varepsilon_0}}\|\mathbf{r}-\mathbf{r}_0\|\frac{\|\mathbf{y}(\mathbf{r})\|}{\|\mathbf{r}-\mathbf{r}_0\|}d\mathbf{r} \\
&+\frac{3b}{V_S}\int_{S'\setminus S_{\varepsilon_0}}\|\mathbf{y}(\mathbf{r})\|d\mathbf{r}+M_0 \\
<&\frac{3b}{V_S}\int_{S_{\varepsilon_0}}\frac{\varepsilon_0}{\rho_2}\varepsilon_0 d\mathbf{r} \\
&+\frac{3b}{V_S}\int_{S'\setminus S_{\varepsilon_0}}\varepsilon_1\|\mathbf{r}-\mathbf{r}_0\|d\mathbf{r}+M_0 \\
=&\frac{3b\varepsilon_0^2 V_{S_{\varepsilon_0}}}{\rho_2 V_S}+\frac{3b\varepsilon_1}{V_S}\|\mathbf{r}'-\mathbf{r}_0\|\int_{S'\setminus S_{\varepsilon_0}}d\mathbf{r}+M_0 \\
=&\frac{3b\varepsilon_0^2 V_{S_{\varepsilon_0}}}{\rho_2 V_S}+\frac{3b\varepsilon_1 V_{S'\setminus S_{\varepsilon_0}}}{V_S}\|\mathbf{r}'-\mathbf{r}_0\|+M_0 \\
:=&U^{S'}_{\rho_1,\rho_2,M_0}
\end{aligned}
$$

where $\mathbf{r}'$ is some point in $S'\setminus S_{\varepsilon_0}$. In the penultimate step, we apply Lemma A.2 from the Appendix. Since $V_{S'}=V_{S'\setminus S_{\varepsilon_0}}+V_{S_{\varepsilon_0}}$, by appropriately selecting ε_0, the above upper bound $U^{S'}_{\rho_1,\rho_2,M_0}$ can be minimized.

□

In our proposed framework, we aim to maximize D. When $D>U^{S'}_{\rho_1,\rho_2,M_0}$, it follows that $\mathbf{y}(\mathbf{r})\notin\Omega^{S'}_{\rho_1,\rho_2,M_0}$. Therefore, as D increases, the likelihood of $\mathbf{y}(\mathbf{r})$ satisfying Equation (5) decreases, while the likelihood of decoupling $\mathbf{h}_t$ and $\mathbf{h}_s$ increases.

4. Experiments

In this section, we conduct extensive experiments to evaluate our proposed framework DisGen. We aim to answer the following questions: (**RQ1**) Does DisGen effectively enhance the size generalizability of GNNs, and how does it compare to other baselines? (**RQ2**) how do various components influence the size generalizability of DisGen?

4.1. Experimental Setup

Datasplits. Each dataset is divided into four subsets: training, validation, small test, and large test. The small test sets include graphs similar in size to the training set, while the large test sets contain significantly larger graphs. The splits are generated by first sorting the dataset samples by size. The training, validation, and small test subsets are randomly selected from the smallest 50% of graphs, while the large test subsets are chosen from the largest 10% of graphs. Further details can be found in Appendix C.

Datasets. We perform experiments on four datasets: BBBP (Wu et al., 2018), PROTEINS (Hobohm et al., 1992), GraphSST2 (Yuan et al., 2021), and NCI1 (Xinyi & Chen, 2018). Each method is evaluated using the F_1 score. Details about the datasets can be found in Appendix B. We also test our method on two larger datasets and report the prediction performance in Appendix E.

Baselines. We assess the performance of DisGen against four baseline approaches for graph size generalization: SizeShiftReg (SSR) (Buffelli et al., 2022), CIGAV2 (Chen et al., 2022), RPGNN (Murphy et al., 2019), and IRM (Arjovsky et al., 2019). We utilize different GNN backbones—GCN (Kipf & Welling, 2016), GIN (Xu et al., 2018), and GraphTransformer (Shi et al., 2020)—in both the mentioned approaches and our framework. The training details and hyperparameter choices can be found in Appendix D.

4.2. Effectiveness of DisGen

We assess the graph prediction performance of DisGen on both small and large test graphs, comparing it against four baseline methods. The results are summarized in Table 1. Our results reveal two key insights:

First, across various datasets, DisGen consistently improves the graph prediction performance of various GNN backbones on both small and large test graphs. Notably, DisGen achieves an average improvement of up to 2.50% on small test graphs and up to 6.64% on large test graphs. These results highlight the effectiveness of DisGen in enhancing both in-distribution and size generalization.

Second, DisGen surpasses other baselines by achieving the highest improvements in F_1 scores for both small and large test graphs, with a more pronounced impact on the latter. Compared to other competitive baselines, such as SSR (Buffelli et al., 2022) and CIGAV2 (Chen et al., 2022), DisGen's superior efficacy stems from its disentangled learning approach. Through explicit removal of size-related information from task-related representations, DisGen consistently exhibits improvements across various GNN backbones. In contrast, SSR and CIGAV2 fail to improve the size generalizability for the graph transformer model (Shi et al., 2020).

4.3. Ablation Study

Design Choices. In this subsection, we study how various design choices impact the size generalizability of DisGen. For simplicity, we employ GCN (Kipf & Welling, 2016) as the backbone model, though our findings are applicable to other GNN backbones. The following design choices are considered: **w/o Aug.**, we remove the augmen-

Table 1. Graph classification performance evaluated on small and large test graphs. Results are reported as average F_1 scores along with their standard deviations. The rightmost column shows the average improvements relative to the original performance using the same GNN backbones. For each backbone model and size category (small/large), the best performance is highlighted in red, and the second-best in violet.

Models \ Dataset	BBBP		PROTEINS		GraphSST2		NCI1		Avg. Imprv.	
	Small	Large	Small	Large	Small	Large	Small	Large	Small	Large
GCN	90.79 ±0.04	76.01 ±0.03	71.74 ±0.04	72.71 ±0.03	89.85 ±0.02	83.55 ±0.02	53.11 ±0.04	38.86 ±0.07	0.0	0.0
GCN+SSR	92.17 ±0.21	82.06 ±0.04	72.21 ±0.02	70.87 ±0.01	89.37 ±0.04	84.05 ±0.03	54.37 ±0.03	39.87 ±0.02	+0.54	+2.16
GCN+CIGAV2	92.00 ±0.09	84.49 ±0.07	72.50 ±0.07	76.67 ±0.02	87.51 ±0.02	83.97 ±0.04	56.63 ±0.03	41.22 ±0.06	+1.60	+5.79
GCN+RPGNN	84.44 ±0.04	75.43 ±0.07	71.91 ±0.12	67.53 ±0.18	90.00 ±0.06	83.30 ±0.06	52.59 ±0.02	40.60 ±0.05	-1.89	-0.93
GCN+IRM	91.63 ±0.13	77.33 ±0.11	70.40 ±0.31	74.50 ±0.02	90.91 ±0.12	83.28 ±0.20	51.40 ±0.08	39.91 ±0.11	-0.75	+1.68
GCN+DisGen	92.92 ±0.09	85.63 ±0.07	75.00 ±0.21	83.41 ±0.31	90.08 ±0.11	85.18 ±0.29	56.14 ±0.02	43.46 ±0.02	+2.16	+6.64
GIN	86.53 ±0.05	65.17 ±0.02	74.00 ±0.07	74.42 ±0.08	89.76 ±0.11	84.70 ±0.22	47.78 ±0.16	30.94 ±0.10	0.00	0.00
GIN+SSR	81.52 ±0.08	70.44 ±0.01	70.20 ±0.05	75.06 ±0.04	90.41 ±0.07	83.43 ±0.44	50.11 ±0.32	32.61 ±0.12	-1.33	+3.21
GIN+CIGAV2	88.46 ±0.07	72.12 ±0.02	75.44 ±0.07	76.67 ±0.10	89.51 ±0.06	84.25 ±0.06	46.56 ±0.04	32.70 ±0.33	+0.33	+2.63
GIN+RPGNN	82.19 ±0.22	70.64 ±0.41	74.08 ±0.05	71.03 ±0.08	89.00 ±0.04	83.83 ±0.04	51.40 ±0.03	31.87 ±0.32	+0.46	+1.45
GIN+IRM	86.09 ±0.31	65.27 ±0.50	75.27 ±0.02	75.07 ±0.09	90.43 ±0.04	84.84 ±0.08	47.52 ±0.06	30.02 ±0.08	+0.35	-0.44
GIN+DisGen	87.15 ±0.08	72.45 ±0.07	75.30 ±0.02	78.25 ±0.06	90.30 ±0.12	84.87 ±0.32	50.28 ±0.42	33.02 ±0.33	+1.24	+3.34
GT	89.78 ±0.02	83.37 ±0.08	70.22 ±0.01	71.62 ±0.02	90.80 ±0.04	83.33 ±0.04	59.00 ±0.12	41.50 ±0.19	0.00	0.00
GT+SSR	91.85 ±0.01	77.65 ±0.02	73.91 ±0.02	73.73 ±0.06	90.46 ±0.12	83.44 ±0.02	59.17 ±0.01	42.80 ±0.01	+1.87	-0.16
GT+CIGAV2	91.43 ±0.31	84.75 ±0.20	70.21 ±0.03	73.01 ±0.04	89.47 ±0.05	84.56 ±0.03	59.86 ±0.03	39.23 ±0.03	+0.45	-0.10
GT+RPGNN	90.98 ±0.19	83.16 ±0.31	71.73 ±0.07	70.00 ±0.10	90.72 ±0.43	84.32 ±0.02	58.82 ±0.03	42.50 ±0.04	+0.77	+0.27
GT+IRM	91.50 ±0.02	85.51 ±0.32	72.53 ±0.01	71.68 ±0.11	90.40 ±0.56	83.20 ±0.04	58.08 ±0.04	35.69 ±0.05	+0.80	-2.87
GT+DisGen	93.77 ±0.01	88.01 ±0.02	73.06 ±0.03	79.22 ±0.03	90.77 ±0.02	85.03 ±0.15	62.19 ±0.03	45.54 ±0.02	+2.50	+4.50

Table 2. Impact of different design choices on graph classification performance across small and large test graphs. Performance is assessed by average F_1 scores and their standard deviations. The rightmost column shows the average improvements relative to DisGen. For each size category (small/large), the model with the highest performance is highlighted in red.

Models \ Dataset	BBBP		PROTEINS		GraphSST2		NCI1		Avg. Imprv.	
	Small	Large	Small	Large	Small	Large	Small	Large	Small	Large
DisGen	92.92 ±0.09	85.63 ±0.07	75.00 ±0.21	83.41 ±0.31	90.08 ±0.11	85.18 ±0.29	56.14 ±0.02	43.46 ±0.02	0	0
w/o Aug.	88.51 ±0.01	80.29 ±0.04	73.34 ±0.05	72.33 ±0.01	84.14 ±0.02	82.62 ±0.01	54.22 ±0.04	38.99 ±0.13	-3.48	-5.86
w/o Decpl.	86.33 ±0.04	78.72 ±0.02	77.43 ±0.02	80.41 ±0.01	88.33 ±0.02	83.95 ±0.04	55.22 ±0.03	32.98 ±0.08	-1.71	-5.41
w/o Decpl.+Decorr.	90.34 ±0.02	73.28 ±0.01	77.11 ±0.03	81.82 ±0.03	88.81 ±0.02	84.21 ±0.04	54.25 ±0.02	36.14 ±0.11	-0.91	-5.56
w/o (Decpl.+ Aug.)+Decorr.	88.76 ±0.00	72.54 ±0.00	76.13 ±0.00	82.35 ±0.01	88.53 ±0.02	85.12 ±0.18	55.35 ±0.01	35.61 ±0.02	-1.34	-5.51

tation components from DisGen; **w/o Decpl.**, we exclude the decoupling loss in DisGen; **w/o Decpl.+Decorr.**, we replace the decoupling loss with a widely used decorrelation loss that enforces cosine similarity to approach 0; and **w/o (Decpl.+Aug.)+Decorr.**, we apply the same decorrelation loss to DisGen and remove both augmentations and decoupling loss. The results for different design choices are presented in Table 1. In general, the removal or replacement of the augmentations and decoupling loss results in significant performance degradation on both small and large test graphs, with a more pronounced effect on the latter. This emphasizes the effectiveness of these two designs. Specifically, excluding the decoupling loss leads to a reduction of 5.41% in F_1 on large test graphs, and this reduction cannot be compensated for by adding a common decorrelation loss, as demonstrated by the performance of **w/o Decpl.+Decorr.**. This underscores the crucial role of the decoupling loss in effectively disentangling size- and task-related information. Additionally, augmentations also contribute to enhancing size generalizability. This is evident in the performance of **w/o Aug.**, where the removal of augmentations results in a decline of 5.86% in F_1 on large test graphs. A similar trend is observed in the performance of **w/o (Decpl.+Aug.)+Decorr.**, where the performance on large test graphs experiences a further decline compared to **w/o Decpl.+Decorr.**.

Augmentations. We evaluate the impact of size- and task-invariant augmentations on the size generalizability of DisGen. Specifically, we create two models: one without the task-invariant branch (denoted by **w/o_task-inv_branch**) and another without the size-invariant branch (denoted by **w/o_size-inv_branch**). Due to the elimination of one branch, employing the contrastive loss becomes impractical since it needs both size- and task-invariant views for comparison. Consequently, we modify the contrastive loss. In **w/o_task-inv_branch** model, we replace the contrastive loss with a loss to maximize the cosine similarity of the size representations $s_i^{(1)}$ and s_i; and in **w/o_size-inv_branch** model, we use a loss to minimize the cosine similarity of $s_i^{(2)}$ and s_i. We present their results in Table 3. As can be seen from the results, removing any augmentation leads to a performance decline on both small and large test graphs.

Pre-trained GNNs. To minimize the effect of knowledge distillation (Yoon et al., 2022) from augmented views, such as task-invariant views, we employ a simple model, GCN (Kipf & Welling, 2016), in the explainable graph model. To

Table 3. Impact of different augmentations on graph classification performance across small and large test graphs. Performance is assessed by F_1 scores and their standard deviations. The rightmost column shows the average improvements relative to DISGEN. For each size category (small/large), the model with the highest performance is highlighted in red.

Models \ Dataset	BBBP		PROTEINS		GraphSST2		NCI1		Avg. Imprv.	
	Small	Large	Small	Large	Small	Large	Small	Large	Small	Large
DISGEN	92.92 $\pm$0.09	85.63 $\pm$0.07	75.00 $\pm$0.21	83.41 $\pm$0.31	90.08 $\pm$0.11	85.18 $\pm$0.29	56.14 $\pm$0.02	43.46 $\pm$0.02	0	0
W/o_size-inv_branch	91.00 $\pm$0.02	82.45 $\pm$0.06	73.35 $\pm$0.03	81.11 $\pm$0.03	89.46 $\pm$0.01	83.38 $\pm$0.02	56.28 $\pm$0.03	39.03 $\pm$0.02	-1.01	-2.93
W/o_task-inv_branch	90.12 $\pm$0.02	81.01 $\pm$0.01	72.39 $\pm$0.03	80.76 $\pm$0.06	89.48 $\pm$0.01	82.33 $\pm$0.03	54.10 $\pm$0.04	36.99 $\pm$0.10	-2.01	-4.15

Table 4. Impact of pre-trained models fed to PGExplainer, assessed by average F_1 scores and their standard deviations.

Dataset \ Models	GIN		GT	
	Small	Large	Small	Large
BBBP	92.22 $\pm$0.09	85.69 $\pm$0.03	91.88 $\pm$0.08	85.01 $\pm$0.03
PROTEINS	75.01 $\pm$0.12	83.49 $\pm$0.09	74.99 $\pm$0.51	84.99 $\pm$0.02
GraphSST2	89.99 $\pm$0.08	85.03 $\pm$0.31	89.33 $\pm$0.05	85.21 $\pm$0.22
NCI1	56.90 $\pm$0.05	43.51 $\pm$0.11	57.16 $\pm$0.21	45.71 $\pm$0.05

evaluate the influence of other pre-trained models, we conduct experiments using PGExplainer (Luo et al., 2020) with different pre-trained models (GIN, and GraphTransformer) to generate task-invariant views. Table 4 presents the performance of our methods with various pre-trained models. The results indicate that (1) our model consistently performs well with different pre-trained models for augmentation, and (2) using superior pre-trained models, such as GT, may further enhance our method's performance compared to the results in Table 1.

Therefore, the observed enhancements in our model are not due to knowledge distillation from the pre-trained models, as we used the least effective pre-trained model. Instead, these improvements result from our model's ability to separate size and task information.

5. Related Work

Size Generalization on Graphs. To enhance the size generalizability of GNNs, Yehudai et al. (2021) introduce the concept of d-patterns and propose a self-supervised framework to address discrepancies in d-patterns between small and large graphs. Bevilacqua et al. (2021) design a size-invariant causal method by modeling the generative process for graphs in the dataset. Chen et al. (2022) propose a model that captures the invariance of graphs under various distribution shifts. Buffelli et al. (2022) propose to simulate the graph size shifts using graph coarsening methods. Chu et al. (2023) tackle the size generalization problem with a Wasserstein barycenter matching layer, which represents an input graph using Wasserstein distances between its node embeddings and learned class-wise barycenters (Keriven et al., 2020; Agueh & Carlier, 2011).

Disentangled Representation Learning. Existing disentangled representation learning has made significant progress in identifying and separating hidden factors in various fields. (Higgins et al., 2016; Xu et al., 2021a; Sarhan et al., 2020; Creager et al., 2019). In the graph domain, GraphLoG (Xu et al., 2021b) employs a self-supervised learning framework to disentangle local similarities and global semantics. DGCL (Li et al., 2021b) disentangles graph-level representations by ensuring that the factorized representations independently capture expressive information from different latent factors. Additionally, Mo et al. (2023) address the complex relationships between nodes, designing a framework to separate task-relevant from task-irrelevant information in multiplex graphs.

Graph Augmentations. To address the problem of data insufficiency and improve the data quality, graph augmentation methods are proposed to generate new graphs by either slightly modifying the existing data samples or generating synthetic ones (Trivedi et al., 2022; Ding et al., 2022; Shorten & Khoshgoftaar, 2019). Perturbations on graph structures, e.g., adding or dropping edges, are widely adopted augmentation methods (Veličković et al., 2018; You et al., 2020; Zhu et al., 2021). Another line of research employs explainable models to identify the key structures of the input graphs, guiding the augmentation process for acquiring more effective unsupervised representations (Shi et al., 2023; Wang et al., 2021).

6. Conclusion

In this paper, we propose a novel framework DISGEN to enhance the size generalization of GNNs with disentangled representation learning. We first utilize size- and task-invariant augmentations to guide the model in learning relative size information. Additionally, we design a decoupling loss to minimize the shared information with theoretical guarantees, which effectively disentangle task- and size-related information. Comprehensive experimental results demonstrate that our DISGEN consistently outperforms state-of-the-art methods.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning. Our work improves the generalizability of GNNs across different sizes, potentially benefiting sectors such as healthcare, bioinformatics, and program synthesis. We anticipate no direct negative societal or ethical implications from our research.

References

Agueh, M. and Carlier, G. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2): 904–924, 2011.

Apostol, T. A. Mathematical analysis / tom m. apostol. 1974. URL https://api.semanticscholar.org/CorpusID:117122580.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Ben-Israel, A. and Greville, T. N. E. Generalized inverses: theory and applications. 1974. URL https://api.semanticscholar.org/CorpusID:121529370.

Bevilacqua, B., Zhou, Y., and Ribeiro, B. Size-invariant graph representations for graph classification extrapolations. In *International Conference on Machine Learning*, pp. 837–851. PMLR, 2021.

Buffelli, D., Liò, P., and Vandin, F. Sizeshiftreg: a regularization method for improving size-generalization in graph neural networks. *Advances in Neural Information Processing Systems*, 35:31871–31885, 2022.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., and Murphy, K. Machine learning on graphs: A model and comprehensive taxonomy. *The Journal of Machine Learning Research*, 23(1):3840–3903, 2022.

Chen, Y., Zhang, Y., Bian, Y., Yang, H., Kaili, M., Xie, B., Liu, T., Han, B., and Cheng, J. Learning causally invariant representations for out-of-distribution generalization on graphs. *Advances in Neural Information Processing Systems*, 35:22131–22148, 2022.

Chu, X., Jin, Y., Wang, X., Zhang, S., Wang, Y., Zhu, W., and Mei, H. Wasserstein barycenter matching for graph size generalization of message passing neural networks. In *International Conference on Machine Learning*, pp. 6158–6184. PMLR, 2023.

Creager, E., Madras, D., Jacobsen, J.-H., Weis, M., Swersky, K., Pitassi, T., and Zemel, R. Flexibly fair representation learning by disentanglement. In *International conference on machine learning*, pp. 1436–1445. PMLR, 2019.

Ding, K., Xu, Z., Tong, H., and Liu, H. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24(2):61–77, 2022.

Fout, A., Byrd, J., Shariat, B., and Ben-Hur, A. Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems*, 30, 2017.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N., Peters, M., Schmitz, M., and Zettlemoyer, L. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

Guo, D., Ren, S., Lu, S., Feng, Z., Tang, D., Shujie, L., Zhou, L., Duan, N., Svyatkovskiy, A., Fu, S., et al. Graphcodebert: Pre-training code representations with data flow. In *International Conference on Learning Representations*, 2020.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2016.

Hobohm, U., Scharf, M., Schneider, R., and Sander, C. Selection of representative protein data sets. *Protein Science*, 1(3):409–417, 1992.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., and Riley, P. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30:595–608, 2016.

Keriven, N., Bietti, A., and Vaiter, S. Convergence and stability of graph convolutional networks on large random graphs. *Advances in Neural Information Processing Systems*, 33:21512–21523, 2020.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Li, G., Müller, M., Ghanem, B., and Koltun, V. Training graph neural networks with 1000 layers. In *International conference on machine learning*, pp. 6437–6449. PMLR, 2021a.

Li, G., Duda, M., Zhang, X., Koutra, D., and Yan, Y. Interpretable sparsification of brain graphs: Better practices and effective designs for graph neural networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1223–1234, 2023.

Li, H., Wang, X., Zhang, Z., Yuan, Z., Li, H., and Zhu, W. Disentangled contrastive learning on graphs (supplementary material).

Li, H., Wang, X., Zhang, Z., Yuan, Z., Li, H., and Zhu, W. Disentangled contrastive learning on graphs. *Advances in Neural Information Processing Systems*, 34:21872–21884, 2021b.

Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., and Zhang, X. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. *arXiv preprint arXiv:1812.09902*, 2018.

Mo, Y., Lei, Y., Shen, J., Shi, X., Shen, H. T., and Zhu, X. Disentangled multiplex graph representation learning. In *International Conference on Machine Learning*, pp. 24983–25005. PMLR, 2023.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Murphy, R., Srinivasan, B., Rao, V., and Ribeiro, B. Relational pooling for graph representations. In *International Conference on Machine Learning*, pp. 4663–4673. PMLR, 2019.

Orsini, F., Frasconi, P., and De Raedt, L. Graph invariant kernels. In *Proceedings of the twenty-fourth international joint conference on artificial intelligence*, volume 2015, pp. 3756–3762. IJCAI-INT JOINT CONF ARTIF INTELL, 2015.

Sarhan, M. H., Navab, N., Eslami, A., and Albarqouni, S. Fairness by learning orthogonal disentangled representations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pp. 746–761. Springer, 2020.

Shi, Y., Huang, Z., Feng, S., Zhong, H., Wang, W., and Sun, Y. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020.

Shi, Y., Zhou, K., and Liu, N. Engage: Explanation guided data augmentation for graph representation learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 104–121. Springer, 2023.

Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Trivedi, P., Lubana, E. S., Yan, Y., Yang, Y., and Koutra, D. Augmentations in graph contrastive learning: Current methodological flaws & towards better practices. In *Proceedings of the ACM Web Conference 2022*, pp. 1538–1549, 2022.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

Wang, Y., Min, Y., Shao, E., and Wu, J. Molecular graph contrastive learning with parameterized explainable augmentations. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1558–1563. IEEE, 2021.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

Xinyi, Z. and Chen, L. Capsule graph neural network. In *International conference on learning representations*, 2018.

Xu, J., Ren, Y., Tang, H., Pu, X., Zhu, X., Zeng, M., and He, L. Multi-vae: Learning disentangled view-common and view-peculiar visual representations for multi-view clustering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9234–9243, 2021a.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Xu, M., Wang, H., Ni, B., Guo, H., and Tang, J. Self-supervised graph-level representation learning with local and global structure. In *International Conference on Machine Learning*, pp. 11548–11558. PMLR, 2021b.

Yan, Y., Zhu, J., Duda, M., Solarz, E., Sripada, C., and Koutra, D. Groupinn: Grouping-based interpretable neural network for classification of limited, noisy brain data. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 772–782, 2019.

Yan, Y., Swersky, K., Koutra, D., Ranganathan, P., and Hashemi, M. Neural execution engines: Learning to execute subroutines. *Advances in Neural Information Processing Systems*, 33:17298–17308, 2020.

Yan, Y., Li, G., et al. Size generalizability of graph neural networks on biological data: Insights and practices from the spectral perspective. *arXiv preprint arXiv:2305.15611*, 2023.

Yanardag, P. and Vishwanathan, S. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1365–1374, 2015.

Yehudai, G., Fetaya, E., Meirom, E., Chechik, G., and Maron, H. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*, pp. 11975–11986. PMLR, 2021.

Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Yoon, J., Kang, D., and Cho, M. Semi-supervised domain adaptation via sample-to-sample self-distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1978–1987, 2022.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

Yuan, H., Yu, H., Wang, J., Li, K., and Ji, S. On explainability of graph neural networks via subgraph explorations. In *International conference on machine learning*, pp. 12241–12252. PMLR, 2021.

Yuan, H., Yu, H., Gui, S., and Ji, S. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):5782–5799, 2022.

Zhang, J. Graph neural networks for small graph and giant network representation learning: An overview. *arXiv e-prints*, pp. arXiv–1908, 2019.

Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., and Wang, L. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, pp. 2069–2080, 2021.

A. Lemmas in Theoretical Analysis

Lemma A.1. *(Solutions of Linear Systems (Ben-Israel & Greville, 1974)) Let* $\mathbf{C} \in \mathbb{C}^{m\times n}, \mathbf{G} \in \mathbb{C}^{p\times q}, \mathbf{B} \in \mathbb{C}^{m\times q}$*. Then the matrix equation*

$$\mathbf{CPG} = \mathbf{B}$$

is consistent if and only if, for some $\mathbf{C}^+, \mathbf{G}^+$*,*

$$\mathbf{CC}^+\mathbf{BG}^+\mathbf{G} = \mathbf{B},$$

in which case the general solution is

$$\mathbf{P} = \mathbf{C}^+\mathbf{BG}^+ + \mathbf{Y} - \mathbf{C}^+\mathbf{CYGG}^+$$

for arbitrary $\mathbf{Y} \in \mathbb{C}^{n\times p}$*.*

In our case, $\mathbf{G}$ and $\mathbf{G}^+$ are identity matrices.

Lemma A.2. *(Mean-Value Theorem for multiple integrals, Theorem 14.16 in Apostol (1974)) Assume that* $g \in \mathbb{R}$ *and* $f \in \mathbb{R}$ *on a Jordan-measurable set* S *in* $\mathbb{R}^n$ *and suppose that* $g(\mathbf{x}) \geq 0$ *for each* $\mathbf{x}$ *in* S*. Let* $m = \inf f(S), M = \sup f(S)$*. Then there exists a real number* λ *in the interval* $m \leq \lambda \leq M$ *such that*

$$\int_S f(\mathbf{x})g(\mathbf{x})d\mathbf{x} = \lambda \int_S g(\mathbf{x})d\mathbf{x}. \tag{19}$$

In particular, we have

$$mc(S) \leq \int_S f(\mathbf{x})d\mathbf{x} \leq Mc(S), \tag{20}$$

where $c(S)$ *represents the area of set* S *in* $\mathbb{R}^n$*.*
If, in addition, S *is connected and* f *is continuous on* S*, then* $\lambda = f(\mathbf{x}_0)$ *for some* $\mathbf{x}_0$ *in* S *and Equation (19) becomes*

$$\int_S f(\mathbf{x})g(\mathbf{x})d\mathbf{x} = f(\mathbf{x}_0) \int_S g(\mathbf{x})d\mathbf{x}. \tag{21}$$

In particular, Equation (21) implies $\int_S f(\mathbf{x})d\mathbf{x} = f(\mathbf{x}_0)\, c(S)$*, where* $\mathbf{x}_0 \in S$*.*

B. Dataset

In this section, we introduce the datasets used in our study. We utilize four pre-processed datasets for the graph classification task: BBBP from the Open Graph Benchmark (Hu et al., 2020), PROTEINS and NCI1 from the TuDataset (Morris et al., 2020), and the GraphSST2 (Yuan et al., 2022) dataset. Below is a detailed description of each dataset:

- **BBBP**: The Blood-Brain Barrier Penetration (BBBP) dataset originates from a study on modeling and predicting barrier permeability. It represents molecules as graphs where nodes are atoms and edges are chemical bonds. Each node features a 9-dimensional vector including atomic number, chirality, formal charge, and ring presence, among other attributes (Hu et al., 2020; Wu et al., 2018). The dataset comprises over 2000 compounds with binary labels indicating their permeability properties.

- **PROTEINS**: This dataset contains macromolecular graphs of proteins, with nodes corresponding to amino acids. Edges connect nodes that are less than six Angstroms apart. Node features include three-dimensional vectors representing the type of secondary structure elements (helix, sheet, or turn). The dataset provides binary labels for protein functionality (enzyme or non-enzyme) and includes 1113 samples.

- **GraphSST2**: A real-world dataset for natural language sentiment analysis. Sentences are transformed into grammar tree graphs using the Biaffine parser (Gardner et al., 2018), with each node representing a word associated with corresponding 768-dimensional word embeddings. The dataset's binary classification task involves predicting the sentiment polarity of sentences and includes a total of 35909 samples.

- **NCI1**: This dataset includes chemical compounds screened for their activity against non-small cell lung cancer. Graphs represent chemical compounds, where nodes denote atoms with one-hot encoded features for atom types, and edges represent chemical bonds. The dataset contains 4110 samples.

Dataset	BBBP	PROTEINS	GraphSST2	NCI1
Train	1026	389	21857	1438
Validation	117	86	4686	310
Small test	142	82	4683	307
Large test	285	82	4683	307

Table 5. Number of graphs in the train, validation and test sets.

Datasets	Graph size		
BBBP	**Train**	**Small test**	**Large test**
Max	27	27	132
Mean	19.3	19	30.5
PROTEINS			
Max	26	26	620
Mean	14.9	16.8	132.8
GraphSST2			
Max	7	7	56
Mean	4.1	4.1	32
NCI1			
Max	27	27	111
Mean	20.4	20.6	58.4

Table 6. Statistics on graph sizes in the train, small and large test sets.

C. Data Pre-processing

This section outlines the data pre-processing techniques employed to prepare the datasets for analysis.

Data splits. For each dataset, we create four distinct sets: training, validation, small test, and large test. The large test set contains graphs that are significantly larger than those in the other sets. To generate the training, validation, and small test sets, we first select the smallest 50% of graphs from each dataset. These graphs are then randomly split in a 70:15:15 ratio for the training, validation, and small test sets, respectively. It is important to note that this split is performed within each class to maintain a consistent label distribution across the training, validation, and small test sets. The large test set is formed by selecting graphs from the remaining pool, ensuring an equal number of graphs per class as in the small test set. This selection process starts from the largest graph in each class, aiming to match the class distribution observed in the small test subset.

Upsampling. Despite careful data splitting, the BBBP dataset presents a significant class imbalance. To mitigate the risk of training a model biased towards the majority class, we employ an upsampling strategy during the training phase. Specifically, graphs belonging to class 0 in the BBBP dataset are upsampled at a 6:1 ratio. The detailed statistics of the resulting dataset are presented in Tables 5 and 6.

D. Training Details

This section outlines the details of the training process and the choices of hyperparameters. We first offer detailed descriptions of the baseline models employed in our study.

- **SizeShiftReg** (Buffelli et al., 2022): This method introduces a regularization approach using graph coarsening techniques to simulate size variations within the training set, thereby enhancing the size generalizability of GNNs in the graph classification tasks. The coarsening technique produces a simplified version of the original graph, preserving certain properties while altering its size, to better accommodate size shifts during training.

- **CIGAV2** (Chen et al., 2022): This framework ensures out-of-distribution generalization under various distribution

shifts by capturing the invariance of graphs. Specifically, it employs Structural Causal Models (SCMs) to characterize these distribution shifts in graphs and posits that GNNs remain invariant to these shifts if they focus on invariant and critical subgraphs.

- **RPGNN** (Murphy et al., 2019): This framework has good representational power and is invariant to graph isomorphism. Built on the principles of finite partial exchangeability, it is model-agnostic and theoretically grounded.

- **IRM** (Arjovsky et al., 2019): This learning paradigm aims to distinguish between properties of the training data that indicate spurious correlations and those that represent the actual phenomenon of interest, thereby enhancing generalization.

Dataset \ Models	GCN		GIN		GT	
	β_1	β_3	β_1	β_3	β_1	β_3
BBBP	0.5	5e4	0.05	5e8	0.1	5e9
PROTEINS	0.05	5e4	0.05	5e4	0.1	1e4
GraphSST2	0.5	5e4	0.15	5e8	0.1	1e8
NCI1	0.5	5e4	0.05	10	0.1	5e9

Table 7. Setting of β_1 and β_3.

HPs \ Dataset	BBBP			PROTEINS	
	Small	Large		Small	Large
β_1	$\beta_2 = 1, \beta_3 = 5e4$		β_1	$\beta_2 = 1, \beta_3 = 5e4$	
0.3	92.38 ± 0.31	82.91 ± 0.05	0.03	76.35 ± 0.02	81.92 ± 0.01
0.4	89.74 ± 0.01	81.40 ± 0.02	0.04	77.42 ± 0.01	81.11 ± 0.04
0.5	92.92 ± 0.09	85.63 ± 0.07	0.05	75.0 ± 0.21	83.41 ± 0.31
0.6	89.43 ± 0.01	82.71 ± 0.05	0.06	76.09 ± 0.01	82.57 ± 0.44
0.7	90.78 ± 0.01	82.18 ± 0.05	0.07	76.23 ± 0.01	81.40 ± 0.40
β_2	$\beta_1 = 0.5, \beta_3 = 5e4$		β_2	$\beta_1 = 0.05, \beta_3 = 5e4$	
0.8	90.44 ± 0.02	82.13 ± 0.08	0.8	75.37 ± 0.03	80.11 ± 0.20
0.9	91.43 ± 0.01	80.83 ± 0.19	0.9	76.4 ± 0.20	84.09 ± 0.07
1	92.92 ± 0.09	85.63 ± 0.07	1	75.0 ± 0.21	83.41 ± 0.31
1.1	92.97 ± 0.01	84.11 ± 0.12	1.1	77.15 ± 0.01	82.44 ± 0.25
1.2	92.66 ± 0.01	81.12 ± 0.01	1.2	76.09 ± 0.02	81.33 ± 0.04
$\beta_3(*1e4)$	$\beta_1 = 0.5, \beta_2 = 1$		$\beta_3(*1e4)$	$\beta_2 = 1, \beta_1 = 0.05$	
3	89.31 ± 0.11	81.76 ± 0.01	3	76.11 ± 0.01	80.90 ± 0.02
4	92.00 ± 0.35	84.02 ± 0.05	4	77.90 ± 0.02	81.56 ± 0.09
5	92.92 ± 0.09	85.63 ± 0.07	5	75.0 ± 0.21	83.41 ± 0.31
6	92.66 ± 0.09	84.87 ± 0.09	6	75.75 ± 0.01	80.91 ± 0.51
7	91.87 ± 0.03	83.58 ± 0.02	7	76.41 ± 0.02	80.86 ± 0.25

Table 8. Impact of hyperparameter choices on graph classification performance across small and large test graphs in BBBP and PROTEINS datasets. Performance is assessed using average F_1 scores and their standard deviations.

Next, we describe our training setup and provide the configurations of our hyperparameters. We use a batch size of 32 and the Adam optimizer with a learning rate of 0.001. For each of the GNN backbones, we employ three convolutional layers and a global mean pooling layer to generate the graph representations. To address overfitting, we implement an early stopping mechanism. This mechanism operates with a patience interval of 50 epochs, and the selection criterion is the lowest validation loss. We run each setup five times and report the average F_1 scores and standard deviations in Table 1.

The hyperparameter settings for the baseline models are consistent with those specified in their original publications to ensure fidelity. Our hyperparameters are chosen based on the performance on the validation set, which contains graphs of

similar sizes to the training data. For DISGEN, we set $k_1 = k_2 = 0.2N$ in the augmentation process, where N is the number of nodes for an input graph. We set $\beta_2 = 1$ for all the experiments. The values for β_1 and β_3 are specified in Table 7.

In Table 8, we present our model's performance with various hyperparameter settings. We vary one hyperparameter at a time while keeping the others constant. For these experiments, we use the GCN (Kipf & Welling, 2016) backbone. Our results indicate that our method is insensitive to hyperparameter changes and demonstrates consistent performance.

E. Two Large Real-world Datasets

We include two more real-world datasets, REDDIT-BINARY (Yanardag & Vishwanathan, 2015) and FRANKENSTEIN (Orsini et al., 2015), and present their results in Table 12. Specifically, REDDIT-BINARY is a social network data. In each graph, nodes represent users, and there is an edge between them if at least one of them responds to the other's comment. A graph is labeled based on whether it belongs to a question/answer-based community or a discussion-based community. FRANKENSTEIN is a dataset of molecules, with each molecule represented as a graph. In this representation, vertices denote chemical atoms labeled with their respective symbols, while edges represent bond types. Tables 9 and 10 provide the statistics for the two datasets. For hyperparameters, we set $\beta_2 = 1$ for all the experiments. The values for β_1 and β_3 are specified in Table 11. According to these tables, the average size of test graphs exceeds 1300 nodes. The performance of DISGEN and the baselines in graph classification is outlined in Table 12. These results confirm that our method successfully generalizes to graphs that are up to ten times larger than those used in training.

Datasets	**Graph size**		
REDDIT-BINARY	**Train**	**Small test**	**Large test**
Max	304	302	3782
Mean	119.2	114	1319.5
FRANKENSTEIN			
Max	16	16	214
Mean	10.5	10.7	40.4

Table 9. Statistics on graph sizes in the train, small and large test sets.

Dataset	**REDDIT-BINARY**	**FRANKENSTEIN**
Train	699	1518
Validation	152	327
Small test	149	324
Large test	149	324

Table 10. Number of graphs in the train, validation and test sets.

Models / **Dataset**	**GCN**		**GIN**		**GT**	
	β_1	β_3	β_1	β_3	β_1	β_3
REDDIT-BINARY	0.5	1e12	0.05	1e12	0.5	1e12
FRANKENSTEIN	0.5	5e4	0.05	5e4	0.5	1e4

Table 11. Setting of β_1 and β_3.

Models \ Datasets	REDDIT-BINARY		FRANKENSTEIN	
	Small	Large	Small	Large
GCN	93.86 $\pm$0.01	41.40 $\pm$0.02	41.57 $\pm$0.01	34.14 $\pm$0.02
GCN + IRM	91.58 $\pm$0.01	39.50 $\pm$0.04	42.54 $\pm$0.02	37.38 $\pm$0.02
GCN + SSR	92.21 $\pm$0.02	41.60 $\pm$0.02	45.47 $\pm$0.03	37.00 $\pm$0.01
GCN + CIGAv2	91.78 $\pm$0.01	42.57 $\pm$0.02	39.81 $\pm$0.01	36.81 $\pm$0.01
GCN + RPGNN	93.96 $\pm$0.01	41.66 $\pm$0.04	43.65 $\pm$0.01	33.98 $\pm$0.01
GCN + DisGen	94.11 $\pm$0.03	45.16 $\pm$0.01	44.80 $\pm$0.01	38.00 $\pm$0.02
GIN	88.47 $\pm$0.01	37.44 $\pm$0.04	54.69 $\pm$0.01	37.38 $\pm$0.01
GIN + IRM	88.70 $\pm$0.02	41.36 $\pm$0.03	57.33 $\pm$0.01	43.64 $\pm$0.01
GIN + SSR	89.10 $\pm$0.01	46.76 $\pm$0.03	58.38 $\pm$0.01	47.90 $\pm$0.01
GIN + CIGAv2	89.57 $\pm$0.03	45.80 $\pm$0.03	57.33 $\pm$0.02	48.01 $\pm$0.02
GIN + RPGNN	86.46 $\pm$0.01	39.67 $\pm$0.04	56.48 $\pm$0.01	47.59 $\pm$0.03
GIN + DisGen	88.91 $\pm$0.03	48.24 $\pm$0.05	57.87 $\pm$0.01	48.30 $\pm$0.02
GT	94.22 $\pm$0.01	59.43 $\pm$0.08	59.72 $\pm$0.01	52.26 $\pm$0.01
GT + IRM	92.15 $\pm$0.01	58.41 $\pm$0.02	59.70 $\pm$0.02	51.65 $\pm$0.02
GT + SSR	91.92 $\pm$0.01	60.11 $\pm$0.02	52.04 $\pm$0.04	53.23 $\pm$0.03
GT + CIGAv2	91.50 $\pm$0.01	61.99 $\pm$0.01	58.20 $\pm$0.01	54.29 $\pm$0.02
GT + RPGNN	92.40 $\pm$0.01	59.22 $\pm$0.01	59.82 $\pm$0.02	50.30 $\pm$0.01
GT + DisGen	93.27 $\pm$0.03	63.19 $\pm$0.03	59.76 $\pm$0.11	53.99 $\pm$0.01

Table 12. Graph classification performance evaluated on two large real-world datasets. Results are reported in average F_1 scores along with their standard deviations. The highest F_1 scores are highlighted in red, and the second-highest are in violet.