# Deep Graph Reprogramming

Yongcheng Jing[1], Chongbin Yuan[2], Li Ju[2], Yiding Yang[2], Xinchao Wang[2], Dacheng Tao[1]
[1]The University of Sydney, Australia, [2]National University of Singapore, Singapore
xinchao@nus.edu.sg, dacheng.tao@gmail.com

## Abstract

*In this paper, we explore a novel model reusing task tailored for graph neural networks (GNNs), termed as "deep graph reprogramming". We strive to reprogram a pre-trained GNN, without amending raw node features nor model parameters, to handle a bunch of cross-level downstream tasks in various domains. To this end, we propose an innovative Data Reprogramming paradigm alongside a Model Reprogramming paradigm. The former one aims to address the challenge of diversified graph feature dimensions for various tasks on the input side, while the latter alleviates the dilemma of fixed per-task-per-model behavior on the model side. For data reprogramming, we specifically devise an elaborated Meta-FeatPadding method to deal with heterogeneous input dimensions, and also develop a transductive Edge-Slimming as well as an inductive Meta-GraPadding approach for diverse homogenous samples. Meanwhile, for model reprogramming, we propose a novel task-adaptive Reprogrammable-Aggregator, to endow the frozen model with larger expressive capacities in handling cross-domain tasks. Experiments on fourteen datasets across node/graph classification/regression, 3D object recognition, and distributed action recognition, demonstrate that the proposed methods yield gratifying results, on par with those by re-training from scratch.*

## 1. Introduction

With the explosive growth of graph data, graph neural networks (GNNs) have been deployed across increasingly wider areas [18, 20, 55, 57, 58], such as recommendation system [48] and autonomous driving [32,45,47]. However, the favorable performance for such applications generally comes at the expense of tremendous training efforts and high memory loads, precluding the deployment of GNNs on the edge side. As such, reusing pre-trained GNNs to alleviate training costs has recently emerged as a trending research topic [7, 11, 19, 21, 39, 53, 54, 56, 69].

Pioneered by the work of [56] that generalize knowledge distillation [14,31,34,59–61] to the non-Euclidean domain, almost all existing approaches on reusing GNNs are achieved by following the distillation pipeline in [56]. Despite the encouraging results, the distilling-based scheme is limited to the *per-task-per-distillation* setting, where a distilled model can only tackle the same task as the teacher can, leading to considerable storage and computation burdens, especially for the deployment of multiple tasks.

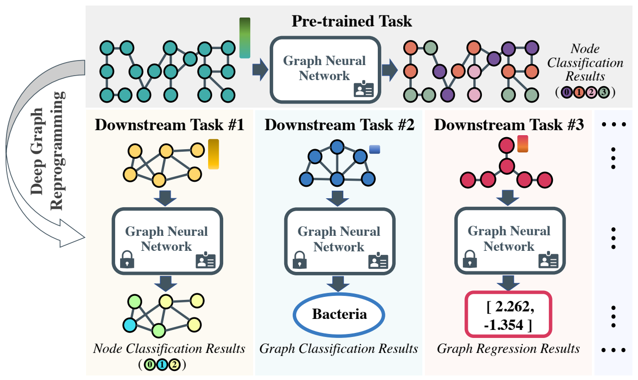

Figure 1. Illustrations of the proposed task of *deep graph reprogramming* (GARE) that aims to reuse pre-trained GNNs to handle plenty of cross-level tasks with heterogeneous graph feature dimensions, without changing model architectures nor parameters.

Meanwhile, the distillation mechanism rests upon the hypothesis that abundant pre-trained models are available in the target domains, which indeed holds for image-based areas that always take data in the regular RGB form, thereby readily allowing for *per-model-multiple-dataset* reusing. However, such an assumption is typically *not* satisfied in the non-Euclidean domain: on the input side, irregular graph samples have heterogeneous feature dimensions, as shown with the color bars in Fig. 1; on the task side, graph analysis takes various task levels and settings, such as graph-, node-, and edge-level learning, as well as transductive and inductive scenarios. Such nature of topological diversities leads to inadequate pre-trained GNNs that fit the target downstream tasks.

In this paper, we strive to take one step towards generalized and resource-efficient GNN reusing, by studying a novel *deep graph reprogramming* (GARE) task. Our goal is to reuse a *single* pre-trained GNN across *multiple* task levels and domains, for example the pre-trained one working on node classification and the downstream ones on graph classification and regression, as shown in Fig. 1. We further impose two constraints to both data and model, where raw features and parameters are frozen in handling downstream tasks. As such, unlike distillation that essentially leverages a pre-trained teacher to guide the *re-training* of a student, the proposed task of GARE, without re-training nor fine-tuning, can thereby be considered to *reprogram* a pre-trained GNN to perform formerly unseen tasks.

Nonetheless, such an ambitious goal is accomplished with challenges: diversified graph feature dimensions and limited model capacities with a single frozen GNN. Driven by this observation, we accordingly reformulate GARE into two dedicated paradigms on data and model sides, respectively, termed as *data reprogramming* (DARE) and *model reprogramming* (MERE). The goal of DARE is to handle downstream graph samples with both the heterogeneous and homogenous dimensions, without amending pre-trained architectures. Meanwhile, MERE aims to strengthen the expressive power of frozen GNNs by dynamically changing model behaviors depending on various tasks.

Towards this end, we propose a universal *Meta-FeatPadding (MetaFP)* approach for heterogeneous-DARE that allows the pre-trained GNN to manipulate heterogeneous-dimension graphs, by accommodating pre-trained feature dimensions via adaptive feature padding in a task-aware manner. The rationale behind the proposed *MetaFP*, paradoxically, is derived from *adversarial reprogramming examples* [8] that are conventionally treated as attacks to learning systems, where attackers secretly repurpose the use of a target model without informing model providers, by inserting perturbations to input images. Here we turn the role of the adversarial reprogramming example on its head, by padding around graph perturbations for generalized cross-task model reusing.

Complementary to the dedicated *MetaFP* that is tailored for heterogeneous-DARE, we also devise a *transductive Edge-Slimming (EdgSlim)* and an *inductive Meta-GraPadding (MetaGP)* methods for homogenous-DARE, that handle the downstream graphs with homogenous dimensions under transductive and inductive task settings, respectively, by adaptively eliminating node connections or inserting a tiny task-specific graph, with only, for example, ten vertices, to the raw input sample. Furthermore, we perform a pilot study on MERE, exploring the pre-trained model capacity for various downstream tasks, by only reprogramming the pre-trained aggregation behavior (*ReAgg*) upon the well-established *Gumbel-Max* trick.

In sum, our contribution is a novel GNN-based model reusing paradigm that allows for the adaption of a pre-trained GNN to multiple cross-level downstream tasks, and meanwhile requires no re-training nor fine-tuning. This is typically achieved through a series of complementary approaches entitled *MetaFP*, *EdgSlim*, and *MetaGP*, that tackle the heterogeneous- and homogenous-dimension graphs within the transductive and inductive scenarios, respectively, together with an elaborated *ReAgg* method to enhance the model capacity. Experimental results on fourteen benchmarks demonstrate that a pre-trained GNN with GARE is competent to handle all sorts of downstream tasks.

## 2. Related Work

**Model Reusing.** With the increasing number of pre-trained GNNs that have been generously released online for reproducibility, the vision community [3, 43, 52, 65–67] has witnessed a growing interest in reusing GNNs to enhance performance, alleviate training efforts, and improve inference speed [10, 11, 21, 24, 25, 42, 44, 64]. The seminal reusing work is performed by Yang *et al*. [56], where a dedicated knowledge distillation (KD) method, tailored for GNNs, is proposed to obtain a lightweight GNN from a teacher. The follow-up work [7] further polishes [56] with a more challenging setting of graph-free KD. Unlike prior works that merely study KD-based GNN reusing, we propose in this paper a novel parallel task of GARE-based model reusing.

**Universal Model.** Other than model reusing, graph reprogramming is also essentially a problem of deriving a *universal* model that is applicable to various-domain tasks. Such universal models have been previously studied in the image and language domains [1, 28, 30, 33, 63]. In this work, we perform a pilot study on developing universal models in the non-Euclidean domain, thereby making one step further towards artificial general intelligence (AGI) [29].

**Adversarial Reprogramming.** The name of the proposed deep graph reprogramming stems from *adversarial reprogramming*, which is a novel type of adversarial attack that seeks to utilize a class-agnostic perturbation to repurpose a target model to perform the task designated by the attacker. Adversarial reprogramming has recently been studied in various areas, including image classification [5, 8, 9, 23, 68] and language understanding [12]. However, the existence of adversarial reprogramming has not yet been validated in the non-Euclidean graph domain. Our paper is the first work that explores adversarial reprogramming in GNNs, and further innovatively *turns* such adversarial manner into guards to derive a novel task for resource-efficient model reusing.

## 3. Motivation and Pre-analysis

In this section, we start by giving a detailed analysis on the dilemma of the prevalent reusing scheme of knowledge distillation, and accordingly propose the novel task of *deep graph reprogramming (*GARE*)*, which leads to resource-efficient and generalized GNN reusing. Then, we uncover the two key challenges of GARE and introduce the proposed paradigms of DARE and MERE with the elaborated rationales on how to address the two challenges.

## 3.1. Task Motivation and Definition

**Prevalent Distillation-based Reusing.** In the literature, almost all existing methods for reusing GNNs are achieved by knowledge distillation (*KD*) elaborated in Task 1.

**Task 1** (**Reusing GNNs via Knowledge Distillation**). *The goal of knowledge distillation is to re-train a compact student model from scratch, that masters the expertise of the pre-trained teacher, via extracting and transferring the knowledge from the pre-trained cumbersome teacher model.*

Such a *KD* manner is inevitably limited by two issues:
**- 1.** ***KD*** is built upon an ideal condition that for any downstream task, sufficient pre-trained teacher models are always available for reusing. Such an assumption indeed holds for most cases of image analysis, where the input data is always RGB-pattern. As such, the publicly available model trained on large-scale datasets, such as *ImageNet*, is readily reusable for downstream classification tasks. However, graph data instead has highly diversified input dimensions, feature types (*e.g*., node and edge features), as well as various task levels (*e.g*., node- and graph-level analysis), making it challenging to reuse online-released GNNs, as image-domain does, for such diversified graph scenarios;
**- 2.** ***KD*** is resource-*inefficient*. The distilled model from *KD* only handles exactly the same task as the teacher does. In other words, every student is always unique to a single task, leading to model redundancy for multi-task scenarios.

**Proposed Novel Deep Graph Reprogramming (GARE).** Driven by the challenges of the *KD*-based model reusing scheme, we develop in this paper a novel paradigm of *deep graph reprogramming* (GARE) for more generalized and resource-efficient model reusing, that explicitly considers the topological uniqueness of graph data:

**Task 2** (**Reusing GNNs via Deep Graph Reprogramming**). *Deep graph reprogramming aims to reuse a pre-trained model, without changing any architecture nor parameter, for a bunch of various-domain and cross-level downstream tasks, via reprogramming graph data or model behaviors.*

As such, the proposed GARE is ideally superior to the ubiquitous *KD* with the following merits:
**+ 1. GARE** allows for the reuse of a *single* pre-trained GNN for *multiple* cross-level/domain downstream tasks and datasets, as shown in Fig. 1, thereby getting rid of the *KD* restriction on well-provided pertinent pre-trained models;
**+ 2. GARE** is free of re-training or fine-tuning, unlike *KD* that substantially re-trains a student model from scratch, thereby making it possible for deployment in resource-constrained environments such as edge computing;
**+ 3. GARE** is memory-efficient, where a pre-trained model with GARE is anticipated to be versatile and multi-talented that integrates the expertise of multiple tasks.

## 3.2. Challenges Towards GARE

The ambitious goal of GARE in Task 2 is primarily accomplished with the two key challenges:
✳ **Data Side:** The first issue to be tackled regards handling various-dimension downstream features, considering that the pre-trained GNN in GARE is frozen without auxiliary transforming layers nor fine-tuning. For example, every node in the *Cora* citation network has 1433 input features, whereas that in *Amazon Co-purchase* graphs has 767 ones;
✳ **Model Side:** The second challenge towards GARE lies in the insufficient model capacity under the per-GNN-multiple-task scenario of GARE, especially for cross-domain downstream tasks as shown in Fig. 1.

To tackle the data and model dilemmas of GARE, we devise a couple of data and model reprogramming paradigms, respectively, as will be elaborated in the following sections.

## 3.3. Reprogramming Paradigms for GARE

### 3.3.1 Data Reprogramming (DARE)

**Rationale Behind DARE.** To tackle the problem of diversified features on the input side, a naïve idea is rearranging graph representations to adapt varying-dimension downstream features to accommodate to the pre-trained GNN. As such, the challenge instead comes to be how to adapt the target downstream features.

To address this challenge of feature adaption, we paradoxically resort to a special type of adversarial attack, termed as *adversarial reprogramming attack*, as introduced in Sect. 2. In essence, the adversarial reprogramming attack demonstrates a security vulnerability of CNNs, where an attacker can easily redirect a model with perturbations to perform the selected task without letting the model providers know, thereby leading to ethical concerns, such as repurposing housekeeping robots to criminal activities.

Our idea here is to flip the role of adversarial reprogramming attacks, by turning the attackers that mean to perturb the model usage, into guards that aim to repurpose a pre-trained GNN to perform the intended downstream tasks.

However, adversarial reprogramming attack is formerly merely studied in the CNN domain. As such, it remains unknown in the machine learning community whether GNNs are also vulnerable to adversarial reprogramming attacks, which is a *prerequisite* for the success in applying the idea of adversarial reprogramming to DARE.

To this end, we as attackers perform in Tab. 1 an evasion attack on graph data, that tries to repurpose a pre-trained GNN designated for *product category prediction* to the new tasks of *molecule classification* and *molecule property regression*, through simply *adding* the generated adversarial perturbations to the raw node features [35]. We employ here

Table 1. Results of adversarial reprogramming attacks on graphs.

| Roles | Model Provider | Adversarial Attacker | Model Provider | Adversarial Attacker |
|---|---|---|---|---|
| Datasets | `Computers` | `ogbg-molbbbp` | `Photo` | `ogbg-molesol` |
| Task Types | Computer-Category Prediction | Molecule Classification | Photo-Category Prediction | Molecule Regression |
| **Before Attack** | *Accuary*: 0.9485 | – | *Accuary*: 0.9561 | – |
| **After Attack** | – | *ROC-AUC*: **0.6132** | – | *RMSE*: **2.7479** |
| **Re-training** | – | *ROC-AUC*: 0.6709 | – | *RMSE*: 1.3000 |

the datasets of *AmazonCoBuy* [27, 41], *ogbg-molbbbp* [49], and *ogbg-molesol* [49] as examples. Surprisingly, with such a vanilla manner, the attacked pre-trained GNNs are extraordinarily competent to handle the unseen tasks, which have a significant domain gap with the former ones, leading to the observation as follows:

**Remark 1** (**Adversarial Reprogramming Attacks on Graph Data**). *Graph neural networks are susceptible to adversarial reprogramming attacks, where an adversarial perturbation on graph data can readily repurpose a graph neural network to perform a task chosen by the adversary, without notifying the model provider.*

**Motivations of Heter-DARE-*MetaFP* and Homo-DARE-*EdgSlim+MetaGP* Methods.** Now, we turn our role back from attackers to reputable citizens that would like to reuse a pre-trained GNN to alleviate the training efforts for downstream tasks. Remark 1 thereby illustrates that:

► **1.** It is technically feasible to convert adversarial reprogramming attack to effective DARE on graph data, which **motivates** us to devise a universal *Meta-FeatPadding (MetaFP)* approach, upon adversarial node feature perturbations, for heterogeneous-DARE (Sect. 4.2);

► **2.** Except for node-level perturbations, other adversarial example types tailored for graph data should also be effective for DARE, such as edge-level perturbations and structure-level perturbations [35], **motivating** us to develop a *transductive Edge-Slimming (EdgSlim)* (Sect. 4.3) and an *inductive Meta-GraPadding (MetaGP)* (Sect. 4.4) approaches, respectively, for homogenous-DARE.

### 3.3.2 Model Reprogramming (MERE)

**Rationale Behind MERE.** Backed by the theory of adversarial reprogramming attacks (Remark 1), in most cases, a pre-trained model equipped with DARE in Sect. 3.3.1 can already achieve encouraging results in tackling various downstream tasks. Despite its gratifying performance, we empirically observe that the downstream performance by only using DARE is prone to a bottleneck especially for some tasks that have considerable domain gaps with the pre-trained one, for example the pre-trained task on paper analysis and the other one on e-commerce prediction.

We conjecture that such a bottleneck is due to the frozen GNN parameters and architectures, leading to insufficient expressive capabilities in modeling cross-domain topological properties. Motivated by the above observation, we further develop a MERE paradigm to strengthen the model capacities, acting as a complement to DARE under the scenarios of tremendous-domain-gap GNN reusing.

To this end, a vanilla possible solution for model enhancement will be resorting to dynamic networks [13] that are well-studied in the CNN domain. Plenty of dynamic inference schemes that are designated for CNNs, in fact, are equally feasible to the non-Euclidean domain of GNNs, such as early exiting, layer skipping, and dynamic routing. Moreover, almost all these dynamic strategies require no changes to original model parameters, thus readily acting as a specific implementation of MERE.

Nevertheless, instead of simply using CNN-based dynamic network schemes, we perform in this paper a pilot study of MERE by explicitly considering the most critical characteristic that is unique to GNNs, namely *message aggregation*, and leaving the explorations of other dynamic paradigms in MERE for future works.

In the literature, message aggregation schemes have already been identified as one of the most crucial components in graph analysis, both empirically and theoretically [6]. However, the significance of aggregation behaviors in model reusing has not yet been explored, which is a precondition for the success of aggregation-based MERE.

To this end, we explore the MERE paradigm by firstly performing a prior study with *Cora* dataset in Fig. 2, that attempts to observe

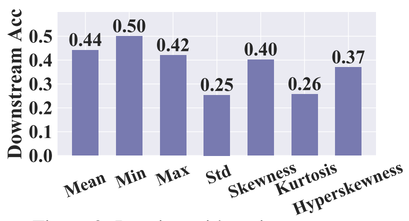

Figure 2. Reusing with various aggregators.

the diverse performance of reusing a fixed GNN pre-trained for classifying the node categories of {Case-Based, Genetic-Algorithm, Neural-Network, Probabilistic-Method}, to directly handle the separate downstream classes of {Reinforcement-Learning, Rule-Learning, Theory}, by only replacing the pre-trained aggregator with other aggregation methods. Remarkably, different aggregators in Fig. 2 lead to distinct downstream performance, which can be summarized as:

**Remark 2** (**Aggregation Matters for Reusing**). *Various aggregators lead to diversified downstream task performance with the same model. There exists an optimal aggregation method tailored for each pair of downstream tasks and pre-trained models.*

► **Motivated** by Remark 2, we accordingly derive a *reprogrammable aggregating (ReAgg)* method as a specific implementation of the MERE paradigm, that aims to dynamically change the aggregation behaviors under various downstream scenarios, which will be elaborated in Sect. 4.5.

## 4. Proposed Methods: Implementing DARE and MERE Paradigms

In this section, we instantiate the proposed paradigms of DARE (Sect. 3.3.1) and MERE (Sect. 3.3.2), by elaborating three DARE methods and one MERE approach, tailored for various scenarios of the GARE-based model reusing.

## 4.1. Overview and Case Discussions

As analyzed in Sect. 3.3, a vanilla method to achieve DARE is generating an adversarial feature perturbation as an addition to raw features. However, such a naïve addition manner is prone to a heavy computational burden, especially for high-dimensional-feature scenarios, where we have to optimize an equally high-dimensional perturbation for downstream tasks. Also, such perturbation addition manner completely changes all raw inputs, thereby intrinsically can be interpreted as transforming downstream data to pre-trained one for model reusing, leading to performance bottleneck when the data gap is too significant for transformation.

Motivated by this observation, we resort to generating lower-dimensional perturbations as *paddings* around raw features, *never* amending any raw input feature. As such, the issues of both computational costs and troublesome transformation are simultaneously alleviated.

Despite its merits, such a perturbation padding manner can only be applicable to the scenario where pre-trained and downstream features have heterogeneous dimensions. Driven by this consideration, we propose to divide the GARE scenarios into *three cases*, and explore them separately to devise the corresponding best-suited methods for more resource-efficient model reusing:

- **Case #1. Universal-Heter-DARE:** Heterogeneous dimensions between pre-trained and downstream features under both transductive and inductive settings, addressed by *Meta-FeatPadding* in Sect. 4.2;
- **Case #2. Transductive-Homo-DARE:** Homogenous pre-trained and downstream dimensions for transductive tasks, solved by *Edge-Slimming* in Sect. 4.3;
- **Case #3. Inductive-Homo-DARE:** Homogenous input dimensions for inductive tasks, tackled by *Meta-GraPadding* in Sect. 4.4.

Furthermore, we provide in Sect. 4.5 an examplar implementation of the MERE paradigm, by proposing *reprogrammable aggregation (ReAgg)* that aims to complement DARE on the model side for challenging downstream tasks.

## 4.2. Universal Meta-FeatPadding for Heter-DARE

The proposed *Meta-FeatPadding (MetaFP)* aims to accommodate the diverse downstream feature dimensions, by padding around the raw features, supported by the theory of *node-level* adversarial perturbations [35].

Given a pre-trained model termed $\text{GNN}_{\text{pre-trained}}$, the process of generating the padded features for downstream tasks with *MetaFP* can be formulated as follows:

$$\min_{\boldsymbol{\delta}_{\text{padding}}} E_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}_{\text{downstream}} \left(\text{GNN}_{\text{pre-trained}}[x||\boldsymbol{\delta}_{\text{padding}}], y\right)\right], \tag{1}$$

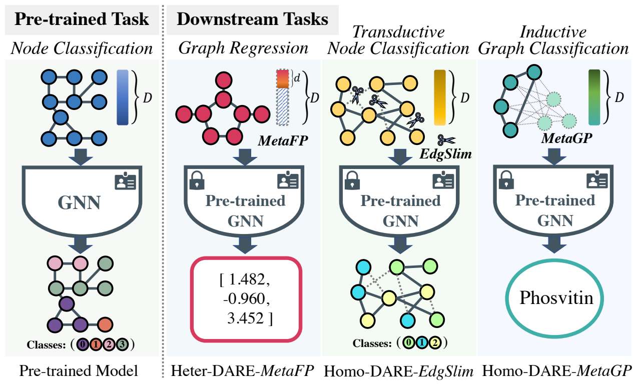

Figure 3. Illustration of the proposed approaches of *MetaFP*, *EdgSlim*, and *MetaGP* for transductive and inductive DARE with heterogeneous and homogenous input dimensions.

where $\mathcal{D}$ is the downstream data distribution, with $(x, y)$ denoting the downstream graph features and the associated labels, respectively. Also, we use $||$ to represent the concatenation operation, which, in fact, performs feature padding that combines the optimized padding features $\boldsymbol{\delta}_{\text{padding}}$ with the raw input features $x$.

As such, the task-specific $\boldsymbol{\delta}_{\text{padding}}$ not only accommodates the feature dimensions of the downstream tasks to those of the pre-trained one with the frozen model of $\text{GNN}_{\text{pre-trained}}$, but also benefits the downstream performance by reducing the loss derived from the downstream loss function of $\mathcal{L}_{\text{downstream}}$. The generation process of $\boldsymbol{\delta}_{\text{padding}}$ is empirically very fast for most cases, where only one or several epochs are typically sufficient for converged results.

During inference, the optimized $\boldsymbol{\delta}_{\text{padding}}$ on training downstream data is padded around all the testing downstream samples to obtain the prediction results. Furthermore, for the case where the output downstream dimensions are not aligned with the pre-trained ones, we simply use the corresponding part of the pre-trained neurons at the final linear layer, which is a common avenue in dynamic networks.

## 4.3. Transductive Edge-Slimming for Homo-DARE

The devised *MetaFP* in Sect. 4.2 is competent to tackle the heterogeneous-dimension case of GARE-based model reusing. Despite its encouraging performance, *MetaFP* is not effective in handling the downstream graph samples that have homogenous feature dimensions to the pre-trained ones, since it is no longer necessary to conduct meta padding for dimension accommodation. As such, it remains challenging in such homogenous-dimension cases, on performing DARE to adapt the pre-trained model to new tasks.

Driven by this challenge, we turn from *node-level* perturbations to another type of adversarial graph examples, namely *adversarial edge-level perturbations* [35], that aim to attack the model by manipulating edges. Here, we flip again the attacker role of *adversarial edge-level perturba-*

*tions* to achieve resource-efficient model reusing, by modifying the node connections in the downstream graph data, meanwhile without changing raw node features, leading to the proposed *Edge-Slimming (EdgSlim)* DARE approach.

To this end, we formulate the algorithmic process of *EdgSlim* as a combinatorial optimization problem:

$$\begin{aligned} \min_{\{u_i,v_i\}_{i=1}^{m}} & \sum_{i=1}^{m} \left| \frac{\partial \mathcal{L}_{\text{downstream}}}{\partial \alpha_{u_i,v_i}} \right| \\ \textit{s.t. } \tilde{\mathcal{G}} &= \text{Modify}\left(\mathcal{G}, \{\alpha_{u_i,v_i}\}_{i=1}^{m}\right) \\ &= \left(\mathcal{G} \setminus \{u_i, v_i\}\right), \text{ if } \frac{\partial \mathcal{L}_{\text{downstream}}}{\partial \alpha_{u_i,v_i}} > 0, \end{aligned} \tag{2}$$

where $\{u, v\}$ denotes the connection between the node $u$ and $v$, with $m$ representing the total number of edges in the input graph $\mathcal{G}$. $\mathcal{L}_{\text{downstream}}$ is the loss function for the downstream task. $\alpha_{u_i,v_i}$ is our constructed unary edge feature, such that we can compute the derivative of $\mathcal{L}_{\text{downstream}}$ with respect to the adjacency matrix of $\mathcal{G}$, with $\alpha_{u,v} = \mathbb{I}(u \in \mathcal{N}(v))$ where $\mathcal{N}(v)$ denotes the set of neighbors for the node $v$. $\setminus$ represents the edge deletion operation.

As such, Eq. 2 indicates that the proposed *EdgSlim* sequentially slims the connections in the downstream graph of which the corresponding edge gradients are greater than 0, starting from the edge with the largest gradients. The downstream loss can thereby be reduced by simply optimizing the connections. Notably, similar to *MetaFP*, the optimization with *EdgSlim* converges very fast, typically with only several epochs, and meanwhile occupies limited resources.

## 4.4. Inductive Meta-GraPadding for Homo-DARE

In spite of the gratifying results of *EdgSlim*, Eq. 2 is not applicable to the inductive task setting, where plenty of graphs are received as inputs. In this case, the edge slimming operation can only be performed on training graphs, not capable of transferring to the testing ones. To alleviate this dilemma, we propose a *Meta-GraPadding (MetaGP)* method to tackle the inductive GARE scenarios, where the downstream features have the same dimensions as the pre-trained ones, as illustrated in Fig. 3.

Our design of *MetaGP* is driven by the *structure-level perturbation* in adversarial examples [35]. In particular, instead of padding the generated perturbations around raw node features, the proposed *MetaGP* yields a tiny subgraph, with only, for example, ten nodes, which is then padded around every downstream graph, of which each meta node connects the downstream graph nodes in a fully-connected manner. The features in the introduced meta graph are generated in the same way as that of yielding padded features in Eq. 1. At the inference stage, the learned meta-graph is padded around all the testing graphs, leading the pre-trained GNN to perform the target downstream inductive task.

The process of generating the meta-graph in *MetaGP* is computation-efficient, where a meta-graph with only ten nodes is typically sufficient for most tasks. Moreover, the feature generation procedure is also lightweight, given the property of inductive graph learning tasks where the input features are generally low-dimension, *e.g.*, the *QM7b* dataset having only 1-dimension features, as well as the *ogbg-molbace*, *ogbg-molbbbp*, and *ogbg-molesol* datasets with an input feature dimension of nine.

## 4.5. Reprogrammable Aggregating for MERE

With the three elaborated DARE methods demonstrated in the preceding sections, a pre-trained GNN can already achieve empirically encouraging results in various downstream tasks and settings. To further improve the reusing performance especially under the large-domain-gap scenarios, we propose here a *reprogrammable aggregating (ReAgg)* method as a pilot study of the MERE paradigm.

Driven by Remark 2, the goal of the proposed *ReAgg* is to adaptively determine the optimal aggregation behaviors conditioned on different downstream tasks, without changing model parameters, thereby strengthening the model capacities. However, such an ambitious goal comes with the challenge of the undifferentiable discrete decisions of aggregators. To address this challenge, one possible solution is resorting to *reinforcement learning (RL)*. However, it is a known issue that *RL* is prone to a high computation burden, due to its Monte Carlo search process. Another solution is to use the *improved SemHash* technique [22] for discrete optimization. However, we empirically observe that *improved SemHash* for MERE is likely to cause the collapse issue, where a specific aggregator is always or never picked up.

Motivated by the above analysis, we propose to leverage Gumbel-Max trick [36] for *ReAgg*, which is a more prevalent strategy for optimizing discrete variables than *improved SemHash* in dynamic neural networks [13]. In particular, to alleviate the dilemma of model collapse, we propose incorporating stochasticity into the aggregator decision process with the well-studied Gumbel sampling [26, 36]. We then propagate the gradients via the continuous form of the Gumbel-Max trick [17]. Specifically, despite the capability in parameterizing discrete distributions, the Gumbel-Max trick is, in fact, dependent on the *argmax* operation, which is non-differentiable. To address this issue, we thereby employ its continuous relaxation form of the Gumbel-softmax estimator that replaces *argmax* with a *softmax* function.

The detailed process of determining the optimal aggregation manner for each downstream task can be formulated as: $\text{Aggregator}_{\mathbf{k}} = \text{softmax}\big((\mathcal{F}(\mathcal{G}) + G)/\tau\big)$, where $\mathbf{k}$ denotes the $k$-th downstream task. Also, $G$ denotes the sampled Gumbel random noise, which introduces stochasticity to avoid the collapse problem. $\mathcal{F}$ represents the intermediate features with $\mathcal{G}$ as inputs. $\tau$ is a constant denoting the softmax temperature. We clarify that for superior performance, $\mathcal{F}$ can be generated by feeding $\mathcal{G}$ into a transforma-

Table 2. Results of reusing a pre-trained model on *Citeseer* to simultaneously handle four unseen tasks with heterogeneous dimensions and objectives, averaged with 20 independent runs. "Re-training" indicates whether the pre-trained parameters are changed. Notably, the 8th line shows that *ReAgg* is more competent for the large-domain-gap scenarios (2.3% improvement averagely), but slightly falls behind for *similar*-domain tasks, such as {*Cora*, *Citeseer*}, both of which classify computer science papers. Also, our *MetaFP* yields stable results that only slightly vary with padding initializations, with standard deviations of {0.0030, 0.0023, 0.0006, 0.0008} for the four downstream tasks.

| **Methods** | **Model Re-training?** | **Model Parameter Sizes** | **Pre-trained Task** | **Downstream Heterogeneous Tasks** | | | |
|---|---|---|---|---|---|---|---|
| | | | `Citeseer` | `Cora` | `Pubmed` | `Computers` | `Photo` |
| Pre-trained Model [37] | × | 474.89K | 0.7950 | N/A | N/A | N/A | N/A |
| Training from Scratch [37] | √ | {474.89K, 184.33K, 64.52K, 99.09K, 96.27K} | 0.7950 | 0.9144 | 0.8530 | 0.9475 | 0.9555 |
| Reusing via Fine-tuning [16] | √ | 474.89K | 0.7950 | 0.8710 | 0.8860 | 0.9542 | 0.9555 |
| Multi-task Learning [2] + SlimGNN [19] | √ | 477.62K | 0.7880 | 0.8780 | 0.8450 | 0.9108 | 0.9317 |
| Vanilla Reusing [37] + SlimGNN [19] | × | 474.89K | 0.7950 | 0.1571 | 0.3250 | 0.5037 | 0.2183 |
| **Ours (*MetaFP*)** | × | **474.89K** | **0.7950** | **0.8335** | **0.7790** | **0.9085** | **0.8909** |
| **Ours (*MetaFP* + *ReAgg*)** | × | **474.89K** | **0.7950** | **0.8312** | **0.8030** | **0.9229** | **0.9213** |

Table 3. Ablation studies of diverse padding sizes/positions and various pre-trained/downstream tasks. Notably, our *MetaFP* is effective even with tiny sizes and random positions.

| **Set of Heterogeneous Tasks {Pre-trained, Downstream}** | **Padding Size** | **Padding Positions** | | | |
|---|---|---|---|---|---|
| | | Front | Center | End | Random |
| {`Computers`, `Photo`} | 22 | 0.9183 | 0.9161 | 0.9212 | 0.9165 |
| {`Photo`, `Pubmed`} | 245 | 0.8420 | 0.8430 | 0.8420 | 0.8440 |
| {`Computers`, `Pubmed`} | 267 | 0.8300 | 0.8320 | 0.8370 | 0.8330 |
| {`Cora`, `Computers`} | 666 | 0.8585 | 0.8792 | 0.8561 | 0.8910 |
| {`Cora`, `Photo`} | 688 | 0.8337 | 0.8785 | 0.8402 | 0.8915 |
| {`Cora`, `Pubmed`} | 933 | 0.8180 | 0.8210 | 0.8200 | 0.8240 |
| {`Citeseer`, `Cora`} | 2270 | 0.8370 | 0.8335 | 0.8417 | 0.8535 |
| {`Citeseer`, `Pubmed`} | 3203 | 0.7790 | 0.7750 | 0.7740 | 0.8010 |

tion layer, which lies out of the pre-trained model and does not directly participate in the inference process as a part of the Gumbel-softmax estimator. In this way, the proposed *ReAgg* adaptively determines the optimal aggregator conditioned on each task, also without changing any pre-trained parameters, thereby enhancing the model capability.

## 5. Experiments

We evaluate the performance of a series of DARE and MERE approaches on fourteen publicly available benchmarks. Here we clarify that our goal in the experiments is *not* to achieve the state-of-the-art performance, but rather reusing a pre-trained GNN to yield favorable results for as many downstream tasks as possible under limited computational resources.

### 5.1. Experimental Settings

**Implementation Details.** Detailed dataset statistics can be found in the supplement. In particular, we follow [15, 19] to split *Amazon Computers*, *Amazon Photo*, and the OGB datasets, whereas for *Cora*, *Citeseer*, and *Pubmed* datasets, we use the splitting protocol in the supervised scenario for more stable results, as also done in [4]. Task-by-task architectures and hyperparameter settings can be found in the supplementary material. All the experiments are performed using a single NVIDIA GeForce RTX 2080 Ti GPU.

**Comparison Methods.** Given our novel GARE setting, there are few existing methods in the literature for a fair comparison, either with distinct task settings or inconsistent

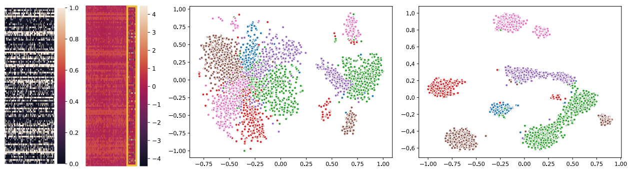

Figure 4. Feature/t-SNE visualizations of (a, c) before padding and (b, d) after padding, with the yellow frame indicating the paddings.

objectives. As such, we derive two possible solutions that partly match our task. Specifically, we derive a *reusing via fine-tuning* approach that reuses a GNN via fine-tuning the parameters for downstream tasks. Moreover, a *multi-task-learning+SlimGNN* pipeline is also developed, that trains a slimmable graph convolution [19] from scratch to accommodate the pre-trained and downstream feature dimensions.

### 5.2. Reprogramming in Heterogeneous Domains

**Heterogeneous Node Property Prediction.** We show in Tab. 2 the results of reprogramming a pre-trained GNN for a bunch of cross-domain node classification tasks, and further give in Tab. 3 the ablation studies of diverse padding sizes and positions as well as different pre-trained and downstream tasks. The 6th line of Tab. 2 shows that *MetaFP* makes it possible for cross-domain GNN reusing. Also, the proposed *ReAgg* for MERE further improves the downstream performance by about 2.3% on average (the 7th line

Table 4. Results of reusing a single model of a node-level task to directly tackle graph regression and graph classification tasks.

| | | | | |
|---|---|---|---|---|
| **Pre-trained** Task | `Photo` | | | |
| **Heterogeneous** Task Type | ***Product Category Prediction*** | | | |
| Pre-trained Results | *Acc*: 0.9561 | | | |
| **Downstream** Tasks | `QM7b` | | `PROTEINS` | |
| **Heterogeneous** Task Types | ***Molecule Regression*** | | ***Protein Prediction*** | |
| Reusing Methods | Vanilla | Ours | Vanilla | Ours |
| Downstream Results | *MAE*: 24.18 | *MAE*: **2.3093** | *Acc*: 0.3304 | *Acc*: **0.6071** |
| Re-training from Scratch | *MAE*: 0.7264 | | *Acc*: 0.6964 | |
| **Pre-trained** Task | `Cora` | | | |
| **Heterogeneous** Task Type | ***Publication Classification*** | | | |
| Pre-trained Results | *Acc*: 0.9121 | | | |
| **Downstream** Tasks | `QM7b` | | `PROTEINS` | |
| **Heterogeneous** Task Types | ***Molecule Regression*** | | ***Protein Prediction*** | |
| Reusing Methods | Vanilla | Ours | Vanilla | Ours |
| Downstream Results | *MAE*: 13.04 | *MAE*: **0.8889** | *Acc*: 0.4018 | *Acc*: **0.5893** |
| Re-training from Scratch | *MAE*: 0.7264 | | *Acc*: 0.6964 | |

of Tab. 2). Moreover, the visualization results before and after applying *MetaFP* are demonstrated in Fig. 4. Furthermore, we'd also like to highlight in Fig. 5 that our method reaches convergence with only a few epochs, making it possible for deployment in resource-constrained environments.

**Heterogeneous Graph Classification and Regression.** Tab. 4 shows the results of reusing a GNN for the more challenging cross-level tasks, indicating our proficiency in such cross-level scenarios.

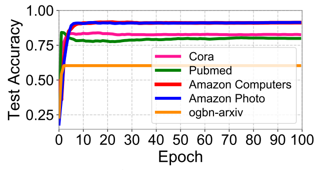

Figure 5. Convergence speed.

## 5.3. Reprogramming in Homogenous Domains

**Homogenous Node Property Prediction.** We perform in Tab. 5 extensive experiments of reusing a GNN for homogenous downstream tasks in a class-incremental setting. The proposed *EdgSlim*, as shown in Tab. 5, achieves competitive performance at a low computational cost (Fig. 5).

Table 5. Results of Homo-DARE that adapts a pre-trained node property prediction model (*ogbn-arxiv-s1*) to handle *20* unseen *homogenous* categories (*ogbn-arxiv-s2*) in ogbn-arxiv dataset [15].

| **Homogenous Multi-class** | **Re-train?** | **Params** | **Pre-trained Task** `ogbn-arxiv-s1` | **Downstream Task** `ogbn-arxiv-s2` |
|---|---|---|---|---|
| **Number of Classes** | - | - | 20 | 20 |
| Pre-trained Model [37] | × | 35.75K | 0.7884 | N/A |
| Training from Scratch [37] | √ | 35.75K | N/A | 0.8115 |
| Reusing via Fine-tuning [16] | √ | 35.75K | N/A | 0.8112 |
| Multi-task Learning [2] | √ | 38.35K | 0.6387 | 0.6776 |
| Vanilla Reusing [37] | × | 35.75K | N/A | 0.2334 |
| **Ours (*EdgSlim*)** | × | **35.75K** | **0.7884** | **0.6034** |

**Homogenous Graph Classification and Regression.** We show in Tab. 6 the cross-domain results for homogenous graph-level tasks, where our *MetaGP* is proficient in reusing a graph classification model for the task of graph regression.

Table 6. Results of homogenous cross-domain graph-level tasks.

| **Pre-trained** Task | `ogbg-molbace` | | | |
|---|---|---|---|---|
| **Homogenous** Task Type | ***Molecular Classification*** | | | |
| Pre-trained Results | *ROC-AUC*: 0.7734 | | | |
| **Downstream** Tasks | `ogbg-molbbbp` | | `ogbg-molesol` | |
| **Homogenous** Task Types | ***Molecular Classification*** | | ***Molecular Regression*** | |
| Reusing Methods | Vanilla | Ours | Vanilla | Ours |
| Downstream Results | *ROC-AUC*: 0.5136 | *ROC-AUC*: **0.6691** | *RMSE*: 6.950 | *RMSE*: **2.050** |
| Re-training from Scratch | *ROC-AUC*: 0.6709 | | *RMSE*: 1.300 | |

**3D Object Recognition.** Tab. 7 shows the results of reusing a pre-trained DGCNN tailored for `ModelNet40` [50], to tackle distinct downstream classes in `ShapeNet` [62]. Remarkably, the proposed *MetaGP* makes it possible for such cross-domain model reusing with large-scale 3D datasets. We also illustrate in Fig. 6 the structure of the feature space at the intermediate layer, showing that ours leads to semantically similar structures to those of re-training from scratch.

Table 7. Results of 3D object recognition tasks with DGCNN [46].

| **Types** | **Tasks** | **# Classes** | ***Pre-trained Performance*** | ***Reusing Performance*** **Vanilla** | **Ours** |
|---|---|---|---|---|---|
| Pre-trained Acc | `ModelNet40` | 40 | 0.9327 | N/A | N/A |
| Downstream Acc | `ShapeNet` | 16 | N/A | 0.1545 | **0.6090** |

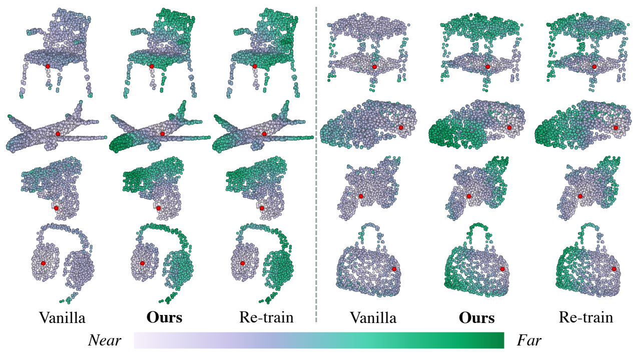

Figure 6. Visualization results of feature space structures, depicted as the distance between the red point and the rest of the others.

**Distributed Action Recognition.** We construct temporally growing graphs from `WARD` [40, 51] and convert the problem of distributed action recognition into that of subgraph classification, as is also done in [38]. The results are shown in Tab. 8, demonstrating the effectiveness of our method.

Table 8. Results of distributed action recognition with incremental time-series data streams and categories as downstream tasks.

| **Tasks** | **Pre-trained Action Categories** `Up` | `ReLi` | `WaLe` | `TuLe` | `Down` | `Jog` | `Push` | `ReSt` | **Acc** |
|---|---|---|---|---|---|---|---|---|---|
| **Pre-trained Acc** | 0.9721 | 0.8563 | 0.9704 | 0.9731 | 0.9265 | 0.9875 | 0.9522 | 0.9229 | 0.9366 |

| **Tasks** | **Downstream Action Categories** `ReSi` | `WaFo` | `TuRi` | `WaRi` | `Jump` | **Acc** |
|---|---|---|---|---|---|---|
| **Downstream Acc** | 0.7337 | 0.9574 | 0.6419 | 0.6893 | 0.8081 | **0.7871** |

## 6. Conclusions

In this paper, we introduce a novel GARE task for resource-efficient and generalized model reusing, tailored for GNNs. Our objective is to reuse a pre-trained GNN for diverse cross-level/domain downstream tasks, being rid of re-training or fine-tuning. To this end, we identified two key challenges on the data and model sides, respectively, and propose a suit of three *data reprogramming* (DARE) and one *model reprogramming* (MERE) approaches to resolve the dilemma. Experiments on fourteen benchmarks across various domains, including node and graph classification, graph property regression, 3D object recognition, and distributed action recognition, demonstrate that the proposed methods lead to encouraging downstream performance, and meanwhile enjoy a low computational cost. In our future work, we will strive to generalize GARE to other domains.

## Acknowledgements

Mr Yongcheng Jing was supported by Australian Research Council Projects in part by FL170100117 and IH180100002. Dr Xinchao Wang is supported by the National Research Foundation Singapore under its AI Singapore Programme (Award Number: AISG2-RP-2021-023).

## References

[1] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2

[2] Fabio Capela, Vincent Nouchi, Ruud Van Deursen, Igor V Tetko, and Guillaume Godin. Multitask learning on graph neural networks applied to molecular property predictions. *arXiv preprint arXiv:1910.13124*, 2019. 7, 8

[3] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, 2021. 2

[4] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018. 7

[5] Pin-Yu Chen. Model reprogramming: Resource-efficient cross-domain machine learning. *arXiv preprint arXiv:2202.10629*, 2022. 2

[6] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *NeurIPS*, 2020. 4

[7] Xiang Deng and Zhongfei Zhang. Graph-free knowledge distillation for graph neural networks. In *IJCAI*, 2021. 1, 2

[8] Gamaleldin F Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial reprogramming of neural networks. In *ICLR*, 2019. 2

[9] Matthias Englert and Ranko Lazic. Adversarial reprogramming revisited. *arXiv preprint arXiv:2206.03466*, 2022. 2

[10] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. *arXiv preprint arXiv:2301.12900*, 2023. 2

[11] Kaituo Feng, Changsheng Li, Ye Yuan, and Guoren Wang. Freekd: Free-direction knowledge distillation for graph neural networks. In *KDD*, 2022. 1, 2

[12] Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. Warp: Word-level adversarial reprogramming. In *ACL*, 2021. 2

[13] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *TPAMI*, 2021. 4, 6

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. 1

[15] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020. 7, 8

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *ICLR*, 2020. 7, 8

[17] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 6

[18] Yongcheng Jing, Yining Mao, Yiding Yang, Yibing Zhan, Mingli Song, Xinchao Wang, and Dacheng Tao. Learning graph neural networks for image style transfer. In *ECCV*, 2022. 1

[19] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Amalgamating knowledge from heterogeneous graph neural networks. In *CVPR*, 2021. 1, 7

[20] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Meta-aggregator: Learning to aggregate for 1-bit graph neural networks. In *ICCV*, 2021. 1

[21] Chaitanya K Joshi, Fayao Liu, Xu Xun, Jie Lin, and Chuan-Sheng Foo. On representation knowledge distillation for graph neural networks. *arXiv preprint arXiv:2111.04964*, 2021. 1, 2

[22] Łukasz Kaiser and Samy Bengio. Discrete autoencoders for sequence models. *arXiv preprint arXiv:1801.09797*, 2018. 6

[23] Eliska Kloberdanz, Jin Tian, and Wei Le. An improved (adversarial) reprogramming technique for neural networks. In *ICANN*, 2021. 2

[24] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In *NeurIPS*, 2022. 2

[25] Songhua Liu, Jingwen Ye, Sucheng Ren, and Xinchao Wang. Dynast: Dynamic sparse transformer for exemplar-guided image generation. In *ECCV*, 2022. 2

[26] Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *NeurIPS*, 2014. 6

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, 2015. 4

[28] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018. 2

[29] Scott McLean, Gemma JM Read, Jason Thompson, Chris Baber, Neville A Stanton, and Paul M Salmon. The risks associated with artificial general intelligence: A systematic review. *JETAI*, 2021. 2

[30] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 2

[31] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014. 1

[32] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1

[33] David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artificial Intelligence*, 2021. 2

[34] Jie Song, Ying Chen, Jingwen Ye, and Mingli Song. Spot-adaptive knowledge distillation. *TIP*, 2022. 1

[35] Lichao Sun, Yingtong Dou, Carl Yang, Ji Wang, Philip S Yu, Lifang He, and Bo Li. Adversarial attack and defense on graph data: A survey. *TKDE*, 2022. 3, 4, 5, 6

[36] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. *IJCV*, 2020. 6

[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018. 7, 8

[38] Chen Wang, Yuheng Qiu, Dasong Gao, and Sebastian Scherer. Lifelong graph learning. In *CVPR*, 2022. 8

[39] Can Wang, Zhe Wang, Defang Chen, Sheng Zhou, Yan Feng, and Chun Chen. Online adversarial distillation for graph neural networks. *arXiv preprint arXiv:2112.13966*, 2021. 1

[40] Chen Wang, Le Zhang, Lihua Xie, and Junsong Yuan. Kernel cross-correlator. In *AAAI*, 2018. 8

[41] Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, et al. Deep graph library: Towards efficient and scalable deep learning on graphs. In *ICLR Workshop*, 2019. 4

[42] Peihao Wang, Rameswar Panda, Lucas Torroba Hennigen, Philip Greengard, Leonid Karlinsky, Rogerio Feris, David Daniel Cox, Zhangyang Wang, and Yoon Kim. Learning to grow pretrained models for efficient transformer training. *arXiv preprint arXiv:2303.00980*, 2023. 2

[43] Wen Wang, Yang Cao, Jing Zhang, and Dacheng Tao. Fp-detr: Detection transformer advanced by fully pre-training. In *ICLR*, 2021. 2

[44] Wen Wang, Jing Zhang, Yang Cao, Yongliang Shen, and Dacheng Tao. Towards data-efficient detection transformers. In *ECCV*, 2022. 2

[45] Yue Wang and Justin M Solomon. Object dgcnn: 3d object detection using dynamic graphs. In *NeurIPS*, 2021. 1

[46] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 2019. 8

[47] Cheng Wen, Baosheng Yu, and Dacheng Tao. Learning progressive point embeddings for 3d point cloud generation. In *CVPR*, 2021. 1

[48] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *CSUR*, 2020. 1

[49] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 2018. 4

[50] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 8

[51] Allen Y Yang, Roozbeh Jafari, S Shankar Sastry, and Ruzena Bajcsy. Distributed recognition of human actions using wearable motion sensor networks. *JAISE*, 2009. 8

[52] Erkun Yang, Cheng Deng, Wei Liu, Xianglong Liu, Dacheng Tao, and Xinbo Gao. Pairwise relationship guided deep hashing for cross-modal retrieval. In *AAAI*, 2017. 2

[53] Xingyi Yang, Zhou Daquan, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. In *NeurIPS*, 2022. 1

[54] Xingyi Yang, Jingwen Ye, and Xinchao Wang. Factorizing knowledge in neural networks. In *ECCV*, 2022. 1

[55] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. Factorizable graph convolutional networks. *NeurIPS*, 2020. 1

[56] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *CVPR*, 2020. 1, 2

[57] Yiding Yang, Zhou Ren, Haoxiang Li, Chunluan Zhou, Xinchao Wang, and Gang Hua. Learning dynamics via graph neural networks for human pose estimation and tracking. In *CVPR*, 2021. 1

[58] Yiding Yang, Xinchao Wang, Mingli Song, Junsong Yuan, and Dacheng Tao. Spagan: Shortest path graph attention network. In *IJCAI*, 2019. 1

[59] Jingwen Ye, Yifang Fu, Jie Song, Xingyi Yang, Songhua Liu, Xin Jin, Mingli Song, and Xinchao Wang. Learning with recoverable forgetting. In *ECCV*, 2022. 1

[60] Jingwen Ye, Yixin Ji, Xinchao Wang, Xin Gao, and Mingli Song. Data-free knowledge amalgamation via group-stack dual-gan. *CVPR*, 2020. 1

[61] Jingwen Ye, Yining Mao, Jie Song, Xinchao Wang, Cheng Jin, and Mingli Song. Safe distillation box. In *AAAI*, 2021. 1

[62] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *TOG*, 2016. 8

[63] Jiahui Yu and Thomas S Huang. Universally slimmable networks and improved training techniques. In *ICCV*, 2019. 2

[64] Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review. *arXiv preprint arXiv:2301.07014*, 2023. 2

[65] Wei Zhai, Yang Cao, Zheng-Jun Zha, HaiYong Xie, and Feng Wu. Deep structure-revealed network for texture recognition. In *CVPR*, 2020. 2

[66] Wei Zhai, Yang Cao, Jing Zhang, and Zheng-Jun Zha. Exploring figure-ground assignment mechanism in perceptual organization. In *NeurIPS*, 2022. 2

[67] Wei Zhai, Hongchen Luo, Jing Zhang, Yang Cao, and Dacheng Tao. One-shot object affordance detection in the wild. *IJCV*, 2022. 2

[68] Yang Zheng, Xiaoyi Feng, Zhaoqiang Xia, Xiaoyue Jiang, Ambra Demontis, Maura Pintor, Battista Biggio, and Fabio Roli. Why adversarial reprogramming works, when it fails, and how to tell the difference. *arXiv preprint arXiv:2108.11673*, 2021. 2

[69] Sheng Zhou, Yucheng Wang, Defang Chen, Jiawei Chen, Xin Wang, Can Wang, and Jiajun Bu. Distilling holistic knowledge with graph neural networks. In *ICCV*, 2021. 1