Property-Aware Relation Networks for Few-Shot Molecular Property Prediction

Quanming Yao, *Senior Member, IEEE*, Zhenqian Shen, Yaqing Wang, *Member, IEEE*, and Dejing Dou, *Senior Member, IEEE*

***Abstract*—Molecular property prediction plays a fundamental role in AI-aided drug discovery to identify candidate molecules, which is also essentially a few-shot problem due to lack of labeled data. In this paper, we propose Property-Aware Relation networks (PAR) to handle this problem. We first introduce a property-aware molecular encoder to transform the generic molecular embeddings to property-aware ones. Then, we design a query-dependent relation graph learning module to estimate molecular relation graph and refine molecular embeddings w.r.t. the target property. Thus, the facts that both property-related information and relationships among molecules change across different properties are utilized to better learn and propagate molecular embeddings. Generally, PAR can be regarded as a combination of metric-based and optimization-based few-shot learning method. We further extend PAR to Transferable PAR (T-PAR) to handle the distribution shift, which is common in drug discovery. The keys are joint sampling and relation graph learning schemes, which simultaneously learn molecular embeddings from both source and target domains. Extensive results on benchmark datasets show that PAR and T-PAR consistently outperform existing methods on few-shot and transferable few-shot molecular property prediction tasks, respectively. Besides, ablation and case studies are conducted to validate the rationality of our designs in PAR and T-PAR.**

***Index Terms*—Few-Shot learning, meta-learning, molecular property prediction, transfer learning.**

I. Introduction

DRUG discovery is an important biomedical task, which targets at finding new potential medical compounds with desired properties such as better absorption, distribution, metabolism, and excretion (ADME), low toxicity and active pharmacological activity [1], [2], [3]. It is recorded that drug discovery takes more than 2 billion and at least 10 years in average while the clinical success rate is around 10% [4], [5], [6]. To speed up this process, quantitative structure property/activity relationship (QSPR/QSAR) modeling uses machine learning methods to establish the connection between molecular structure and particular properties [7]. Predictive models can be leveraged in virtual screening to discover potential molecules more efficiently [8]. However, molecular property prediction (MPP) is essentially a few-shot problem, which makes it hard to solve. Only a small amount of candidate molecules can pass virtual screening to be evaluated in the lead optimization stage of drug discovery [9]. After a series of wet-lab experiments, most candidates eventually fail to be a potential drug due to the lack of any desired properties [7]. These together result in a limited number of labeled data [10].

Few-shot learning (FSL) [11], [12] methods target at generalizing from a limited number of labeled data. The mainstream methods can be roughly categorized into two types: metric-based methods and optimization-based methods. Metric-based methods target at learning an embedding space that can be shared across different tasks, such as Prototypical Networks (ProtoNet) [13], Relation Network [14] and EPNet [15]. Optimization-based methods train a meta-learner that can be quickly adapted to new tasks with limited labeled samples, such as MAML [16], LEO [17] and MetaOptNet [18]. These FSL methods have been widely applied in computer vision and natural language processing, achieving good results. The development of FSL methods makes it possible for machine learning models to be utilized in scenarios where labels are rare or hard to obtain.

As MPP is naturally a few-shot problem, FSL methods are also developed to solve this problem [3], [8]. These methods all first use graph-based molecular encoders, i.e., variants of Graph Neural Networks (GNNs) [19], [20], [21], [22], to obtain graph-level representation as the molecular embedding, then develop FSL approaches that enable them to learn from a few labeled molecules. Specifically, the pioneering IterRefLSTM [3] adopts GCN [19] as the molecular encoder and adapts Matching Networks [23] proposed for few-shot image classification to handle few-shot MPP tasks. The recent Meta-MGNN [8] leverages a GNN pretrained from large-scale self-supervised tasks as molecular encoder [24] and uses MAML [16] as its FSL method. Moreover, Meta-MGNN introduces additional self-supervised tasks such as bond reconstruction and atom type prediction to be jointly optimized with the MPP tasks.

However, the aforementioned methods neglect two key facts in MPP. The first fact is that the same molecule shows different properties in different property prediction tasks. This can be

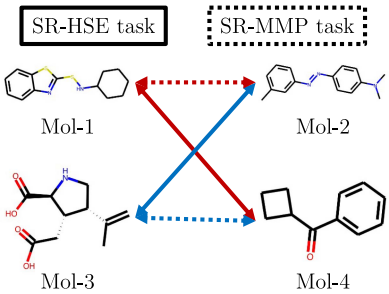
Manuscript received 26 September 2022; revised 25 January 2024; accepted 11 February 2024. Date of publication 21 February 2024; date of current version 2 July 2024. This work was supported by the National Natural Science Foundation of China under Grant 92270106, in part by the Independent Research Plan of the Department of Electronic Engineering Department at Tsinghua University, and in part by the Tsinghua University - Tencent Joint Laboratory for Internet Innovation Technology. Recommended for acceptance by S. K. Zhou. *(Corresponding author: Yaqing Wang.)*

Quanming Yao and Zhenqian Shen are with the Department of Electronic Engineering, Tsinghua University, Beijing 100190, China.

Yaqing Wang and Dejing Dou are with Baidu Research, Baidu Inc., Beijing 100085, China (e-mail: wangyaqing01@baidu.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2024.3368090, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2024.3368090



Molecule		Label	
ID	SMILES	SR-HSE	SR-MMP
Mol-1	c1ccc2sc(SNC3CCCCC3)nc2c1	1	1
Mol-2	Cc1cccc(/N=N/c2ccc(N(C)C)cc2)c1	0	1
Mol-3	C=C(C)[C@H]1CN[C@H](C(=O)O)[C@H]1CC(=O)O	0	0
Mol-4	O=C(c1ccccc1) C1CCC1	1	0

Fig. 1. Examples of relation graphs for the same molecules coexisting in two tasks of Tox21. Red (blue) edges mean the connected molecules are both active (inactive) on the target property.

commonly observed in benchmark MPP datasets. As shown in Fig. 1, Mol-2 from the Tox21 dataset [25] is active in SR-HSE task while inactive in SR-MMP task. However, IterRefLSTM and Meta-MGNN use graph-based molecular encoders to encode molecules regardless of target properties. The second fact is that the relationships among molecules are important and worth-learning. We can see that learning the relationships among molecules will help us predict the property of molecules from Fig. 1. However, existing works do not try to learn the relationships among molecules.

To handle these problems, we propose Property-Aware Relation networks (PAR) which is compatible with existing graph-based molecular encoders. In PAR, we design a property-aware molecular encoder to integrate property-related information into molecular embeddings. Moreover, we construct a query-dependent relation graph for molecules which learns relationships among different molecules to help the model predict the property of query molecules. But considering that different parts of PAR model represent information of different levels, we propose a MAML-based meta-learning strategy with selective update approach to separately capture the generic knowledge shared across different tasks and those specific to each task. PAR combines metric-based (query-dependent relation graph learning) and optimization-based (MAML framework) few-shot learning methods, absorbing the strengths of two kinds of methods.

While PAR performs well in few-shot MPP, it is unable to handle the distribution shift between training and test data, which is also very common in practical AI-aided drug discovery [26], [27]. For example, when a severe epidemic like COVID-19 occurs, we need to handle a new target with unseen data distribution and the performance of common few-shot MPP methods will largely degrade. To tackle the distribution shift, we first formulate the problem into transferable FSL setting where we need to transfer knowledge from source domain to target domain. Then, we propose Transferable Property-Aware Relation Networks, i.e., T-PAR, for transferable few-shot MPP. Based on PAR, T-PAR uses joint sampling strategy to model molecules from source and target domain in the training procedure together. Moreover, T-PAR extends joint relation graph learning by putting molecules from source and target domain into the same relation graph to model the cross domain relationship.

We conduct extensive empirical studies on real MPP datasets. Results show that PAR and T-PAR consistently outperform the compared methods respectively. Further model analysis shows PAR can obtain property-aware molecular embeddings and model molecular relation graph properly. We also conduct ablation studies that show the components in PAR and T-PAR are all vital to their success.

Difference with Conference Version: This paper is an extension of a previous conference paper published in NeurIPS 2021 [28]. The differences are as follows:

- *Related works:* We add more introductions of GNNs (Section II-A), FSL (Section II-B) and MPP (Section II-C).
- *New problem setting:* We introduce transferable few-shot MPP problems (Section IV-A), which consider the distribution shift between model training and testing in practical drug discovery applications.
- *New method:* We extend PAR as T-PAR to handle transferable few-shot MPP problems. The key is joint sampling strategy (Section IV-B) and joint relation graph learning (Section IV-C).
- *Experiments:* For FSL setting, we add more baselines (Section V-A2). For transferable FSL setting, we provide extensive experimental results to show that T-PAR (Section V-B2) outperforms the others (including PAR) on benchmark datasets. Finally, we conduct ablation study to validate the design considerations of adjustable components (Section V-B3 and V-B4) and provide case study (Section V-B5) for T-PAR.

Notation: In the sequel, we denote vectors by lowercase boldface, matrices by uppercase boldface, and sets by uppercase calligraphic font. For a vector $\mathbf{x}$, $[\boldsymbol{x}]_i$ denotes the ith element of $\mathbf{x}$. For a matrix $\mathbf{X}$, $[\mathbf{X}]_{i:}$ denotes the vector on its ith row, $[\mathbf{X}]_{ij}$ denotes the (i, j)th element of $\mathbf{X}$. The superscript $(\cdot)^\top$ denotes the matrix transpose, the parenthesized superscript $(\cdot)^{(\tau)}$ denotes the τth iteration, $\nabla_{\boldsymbol{\theta}} f$ denotes taking gradient of function f w.r.t $\boldsymbol{\theta}$.

II. Related Works

A. Graph Neural Networks (GNNs)

A graph neural network (GNN) can learn expressive node or graph representation from the topological structure and associated features of a graph via neighborhood aggregation [19], [29],

[30]. Consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with node feature $\mathbf{h}_v^{(0)}$ for each node $v \in \mathcal{V}$ and edge feature $\mathbf{b}_{vu}$ for each edge $e_{vu} \in \mathcal{E}$ between nodes v, u. At the lth layer, GNN updates the node embedding $\mathbf{h}_v^{(\tau)}$ of node v as

$$\mathbf{h}_v^{(\tau)} = \texttt{UPDATE}^{(\tau)}\left(\mathbf{h}_v^{(\tau-1)}, \mathbf{h}_{\text{agg}}^{(\tau-1)}\right), \tag{1}$$

where $\mathbf{h}_{\text{agg}}^{(\tau-1)} = \texttt{AGGREGATE}^{(\tau)}(\{(\mathbf{h}_v^{(\tau-1)}, \mathbf{h}_u^{(\tau-1)}, \mathbf{b}_{vu}) | u \in \mathcal{H}(v)\})$, and $\mathcal{H}(v)$ is a set of neighbors of v. After T iterations of aggregation, the graph-level representation $\mathbf{g}$ for $\mathcal{G}$ is obtained as $\mathbf{g} = \texttt{READOUT}(\{\mathbf{h}_v^{(T)} | v \in \mathcal{V}\})$, where $\texttt{READOUT}(\cdot)$ function aggregates all node embeddings into the graph embedding [21].

As the original graph can contain missing or noisy edges, graph structure learning is proposed to refine graph structure dynamically during learning node embeddings [31], [32], [33]. Generally, they iterate over two steps: (i) estimate adjacency matrix (i.e., refining neighborhood $u \in \mathcal{H}(v)$) which encodes graph structure from current node embeddings; and (ii) apply GNN on this updated graph to obtain new node embeddings.

B. Few-Shot Learning (FSL)

Few-shot learning (FSL) [12] is a type of machine learning that aim at generalizing from a limited number of examples. Recent years witness rapid development of FSL methods. They can be roughly divided into categories, metric-based methods and optimization-based methods. Metric-based methods [13], [14], [15], [23], [34] target at learning a feature space and conduct classification through calculating similarity among different samples in that space. For example, ProtoNet [13] conducts classification for a sample by calculating its distance to each class prototype in a learned embedding space. Relation Network [14] additionally trains a deep non-linear distance-based model to compare support and query samples. EPNet [15] proposes an embedding propagation strategy to make feature representation of samples smoother. Optimization-based methods [16], [17], [18], [35] aim at finding a set of model parameters that can be quickly adapted to new FSL tasks. For example, MAML [16] trains a good initialized model from the base training set that can adjust parameters to new tasks with a small number of gradient steps. LEO [17] designs latent embeddings to encode model parameters and adapts the embeddings through gradient steps for different tasks. MetaOptNet [18] learns a feature representation model that generalizes well across different novel classes on linear classifiers. We elaborate MAML in the sequel, as it is used to train the proposed model.

1) MAML and Its Variants: Model-Agnostic Meta-Learning (MAML) [16] targets at learning a good initialization of a classification model $f_{\mathbf{\Phi}}$ with parameters $\mathbf{\Phi}$, so that the model can be quickly adapted to target task $\mathcal{T}_0$ using gradient descent. The framework of MAML contains two key points. Firstly, the parameter update of MAML is divided into inner loop update and outer loop update. Inner loop update performs for a specific training task $\mathcal{T}_\omega = \{\mathcal{S}_\omega, \mathcal{Q}_\omega\}$ and uses the loss $L_{\mathcal{S}_\omega}$ calculated on the support set $\mathcal{S}_\omega$ in $\mathcal{T}_\omega$. Outer loop update performs for all the tasks in $\mathcal{Z}$ and uses the loss $L_{\mathcal{Q}_\omega}$ calculated on the query sets $\mathcal{Q}_\omega$ in these tasks. Secondly, the learning objective for outer loop update should contain the adaptation form of model parameters $\mathbf{\Phi}$ in inner loop update.

Specifically, $\mathbf{\Phi}$ can be adapted for task $\mathcal{T}_\omega$ in inner loop update as follows $\mathbf{\Phi}_\omega = \mathbf{\Phi} - \alpha \nabla_{\mathbf{\Phi}} L_{\mathcal{S}_\omega}(f_{\mathbf{\Phi}})$, where α is the inner learning rate. Then, MAML leverages the few-shot classification tasks in $\mathcal{Z}$ and proposes the following learning objective w.r.t. $\mathbf{\Phi}$, i.e.,

$$\min_{\mathbf{\Phi}} \sum\nolimits_{\omega} L_{\mathcal{Q}_\omega}(f_{\mathbf{\Phi}_\omega}) = \sum\nolimits_{\omega} L_{\mathcal{Q}_\omega}(f_{\mathbf{\Phi} - \alpha \nabla_{\mathbf{\Phi}} L_{\mathcal{S}_\omega}(f_{\mathbf{\Phi}})}),$$

which can still be optimized with gradient descent in outer loop update. Due to the usefulness of MAML, many of its variants have been proposed. For example, ANIL [36] only retains the update of task-specific head layer in inner loop update to simplify MAML. UNICORN-MAML [37] increases the number of gradient steps in inner loop update and proposes an approach to tackle with class permutation in few-shot classification problems.

2) Transferable FSL: Most existing FSL methods assume that training tasks and target tasks follow the same distribution. However, target task usually has a distribution shift compared with training tasks in real-world applications, which calls for transferable FSL methods. Current transferable FSL methods mainly adopt two problem settings. The first type is called few-shot cross domain generalization which does not have any target domain data in training step. MLDG [38] proposes a meta-learning technique that synthesizes virtual testing domains within each mini-batch to achieve domain generalization. FT [39] introduces feature-wise transformation to simulate various feature distributions in unseen target domain. While the second type provides a few labeled samples in target domain during training. DAPN [40] proposes a metric-based transfer learning method for few-shot image classification. It obtains superior performance compared with other transfer learning methods (e.g. ADDA [41] and CDAN [42]) in this setting due to its specialized design for limited target domain training data. Building upon the principles of MAML, BOIL introduces a novel approach that updates the model's body while freezing the head in the inner loop update. This strategy leverages the representation changes across various tasks, making it applicable to the transferable FSL problem setting we explore in this work.

C. Molecular Property Prediction (MPP)

Molecular property prediction (MPP) attempts to establish connection between molecular structures and some particular properties [7], [43], [44]. There are many classical machine learning based methods that try to tackle with the MPP problems. ESOL [45] describes molecules by their basic chemical signatures and use linear regression to predict the molecular property. Molecular Graph Networks [46] turns a molecular graph into an adaptive molecular composite descriptor and use a simple neural network to get a scalar output. Extended-Connectivity Fingerprints [47] proposes topological fingerprints for molecular characterization and the prediction can be obtained through different kinds of predictor from the fingerprints.

The above methods provide a framework for the machine learning based molecular property prediction methods, which

contains two parts, a molecular encoder that encode a molecule into a vector, and a classifier that predict the property. However, they do not produce satisfactory results because of their drawbacks in two aspects. Firstly, these molecular encoders are simple and cannot exploit the graph structure of molecules. Secondly, they neglect the lack of labeled data in practical MPP problems [10], which will degrade the accuracy of machine learning models. Recently, machine learning based MPP methods gradually outperform the conventional *ab initio* computations [48], [49] as they make improvements upon the two drawbacks, which will be introduced in the following sections.

1) GNN-Based Molecular Encoder: Through the improvement of molecular encoders, better molecular representations can be obtained, which will enhance the accuracy of property prediction. Recently, graph-based molecular encoders are popularly used and obtain the state-of-the-art performance [49]. UG-RNN [50] is the first graph-based neural networks used in the MPP problems, which views a molecule as an undirected graph and propagates messages among different atoms by recursive neural network (RNN). Different kinds of GNNs with strong representation ability of graphs are also widely applied as molecular encoders. Graph convolutional networks (GCN) [19] is used in neural graph fingerprints [51] to better encode molecular graphs. Attentive FP [52] adopts the graph attention network (GAT) [22] to capture graph structures and achieves outstanding results. Message passing neural networks (MPNN) [29] integrates different kinds of molecular encoders and introduces a framework that leverage both node and edge features in molecular graphs.

2) Learning From a Few Labeled Data: Lack of labeled data is a ubiquitous problem in practical MPP applications. Classically, transfer learning [53] can handle the lack of labeled data through domain adaptation, which is a type of machine learning that targets at improving the learning of target domain knowledge using knowledge from the source domain. Transfer learning methods are applied in MPP problems in recent years. For example, ChemNet [54] trains a model on source domain and uses transfer learning methods to fine-tune the model on target domain tasks with few labeled data; DTCR [2] proposes a model using GCN and adversarial domain adaptation network [41] to transfer the knowledge across domains.

Recently, pretraining [55] techniques are proposed to tackle with the lack of labeled data, which can obtain better initialized model. Unlike the above transfer learning methods, these methods are able to generalize to new tasks. SMILES-BERT [56] pretrains the model with a large scale of unlabeled dataset through a masked SMILES recovery task that can be quickly generalized to new tasks. The GNN-based molecular encoder can also be pretrained for a better parameter initialization. Pre-GNN [24] implements self-supervised methods for pretrained GNN that can capture the local and global representations simultaneously. GROVER [9] integrates GNN into the Transformer-style architecture to pretrain a class of more expressive molecular encoders. Motif-based graph self-supervised learning method [57] designs motif-level self-supervised tasks to capture rich information in molecular subgraphs in the pretraining process.

As discussed in Section II-B, FSL methods are promising to learn from a few labeled samples and generalize from base dataset to new tasks. They have been recently introduced in MPP. IterRefLSTM [3] and Meta-MGNN [8] are two exemplar works in this direction. Specifically, IterRefLSTM modifies the Matching Networks to a molecular property predictor, while Meta-MGNN implements MAML as the few-shot predictor to obtain the property prediction. Besides, pretraining can be used to warm-start GNN-based molecular encoder in FSL methods as done in Meta-MGNN [8], which can help further boost learning performance.

III. Property-Aware Relation Networks

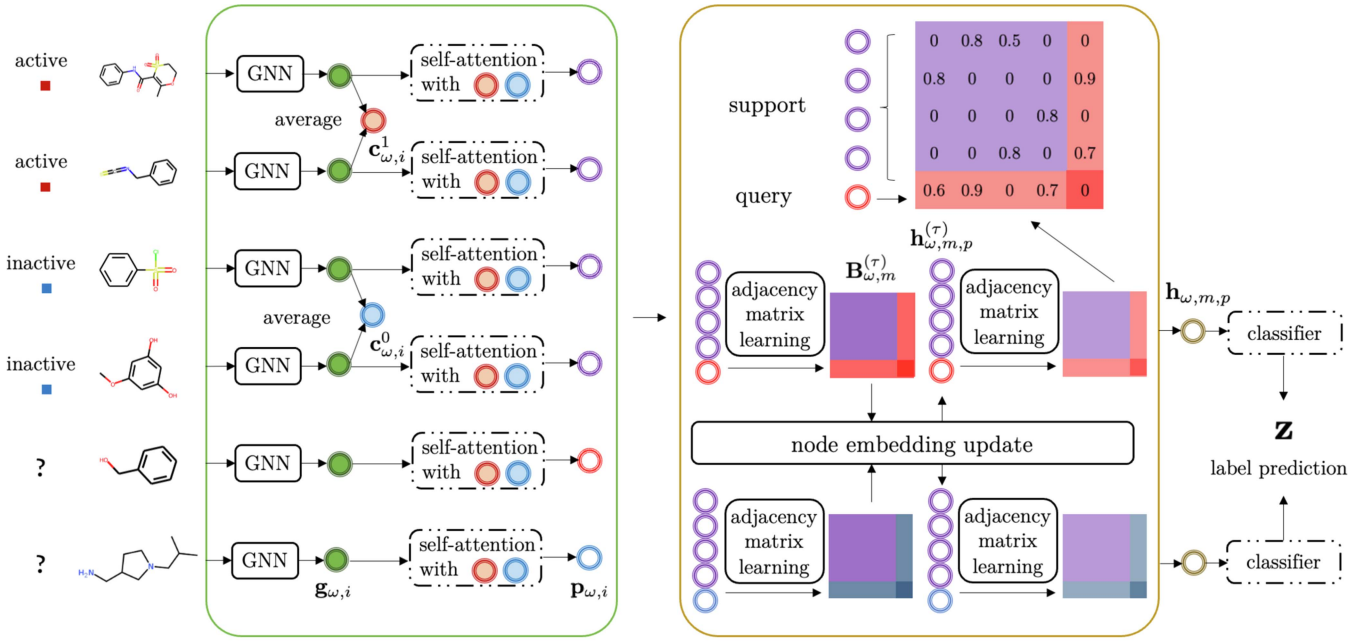
In this section, we introduce PAR (Fig. 2), Property-Aware Relation networks, for few-shot MPP tasks. We first provide the problem formulation (Section III-A). Then, we present the key components of PAR: (i) property-aware molecular encoder (Fig. 2(a)) which is specially designed to obtain property-aware molecular embedding for each molecule (Section III-B), and (ii) query-dependent relation graph learning (Fig. 2(b)) which learns a query-dependent relation graph among molecules to model molecular relationship w.r.t. the target property and propagate information among related molecules (Section III-C). We adopt a meta-learning strategy with selective update approach to optimize PAR (Section III-D). Finally, we conclude this section with the complete algorithm (Section III-E).

A. Problem Formulation

In this paper, we focus on classification tasks within MPP, the label of a molecule can only be "active" or "inactive" corresponding to a particular property. The discussion on applying the proposed method for regression tasks are provided in Appendix A. Following [3], [8], we take MPP as a 2-way K-shot binary classification problem. In this case, we are given a 2-way K-shot classification task $\mathcal{T}_0$, in which there is a support set $\mathcal{S}_0 = \{(\mathbf{x}_{0,i}, y_{0,i})\}_{i=1}^{2\times K}$ and a query set $\mathcal{Q}_0 = \{(\mathbf{x}_{0,j}, y_{0,j})\}_{j=1}^{M}$. The support set contains K samples from each of 2 classes. The query set contains M samples, whose labels are only used to test the classification model. The number of labeled data in the support set is usually too small to produce a good classification model, so we are given another auxiliary base training set $\mathcal{D}$ with sufficient numbers of labeled samples. The base training set has a disjoint label space from the support and query set. $\mathcal{D}$ is used to sample a set of 2-way K-shot tasks $\mathcal{Z} = \{\mathcal{T}_\omega\}$ to train the model in the training stage. Each task $\mathcal{T}_\omega$ contains a support set $\mathcal{S}_\omega = \{(\mathbf{x}_{\omega,i}, y_{\omega,i})\}_{i=1}^{2\times K}$ and a query set $\mathcal{Q}_\omega = \{(\mathbf{x}_{\omega,j}, y_{\omega,j})\}_{j=1}^{M}$. The target is to classify molecules in query set $\mathcal{Q}_0$ in target task accurately.

B. Property-Aware Molecular Encoder

As introduced in Section II-C1, graph-based molecular encoders can obtain good molecular embeddings. By learning from large-scale tasks, they can capture generic information shared by molecules [9], [24]. Thus, we first use a graph-based molecular encoder such as GIN [21] and Pre-GNN [24] to extract molecular embeddings $\mathbf{g}_{\omega,i} \in \mathbb{R}^{d^g}$ of length d^g for $\mathbf{x}_{\omega,i} \in \mathcal{S}_\omega \cup \mathcal{Q}_\omega$. The parameter of this graph-based molecular encoder is denoted as



(a) Property-aware molecular encoder (b) Query-dependent relation graph learning

Fig. 2. The architecture of the proposed PAR, where we plot a 2-way 2-shot task with 2 query molecules from Tox21. PAR is optimized over a set of tasks. Within each task $\mathcal{T}_\omega$, the modules with dotted lines are fine-tuned in inner loop update while those with solid lines are fixed. The PAR model can be mainly divided into two parts: (a) Property-aware molecular encoder: Each molecule $\mathbf{x}_{\omega,i}$ will first be represented as $\mathbf{g}_{\omega,i}$ using graph-based molecular encoder, then transformed to $\mathbf{p}_{\omega,i}$ by self-attention step with class prototypes; and (b) Query dependent relation graph learning: The property-aware molecular embeddings further iteratively update on the query-dependent relation graph as $\mathbf{h}_{\omega,m,p}$, which is taken as the final molecular embedding and used for class prediction.

$\mathbf{W}_g$. However, these methods cannot capture property-aware information. When learning across tasks, the same molecule can be evaluated for multiple properties. This leads to a one-to-many relationship between a molecule and different properties (as shown in Fig. 1), which makes few-shot molecular property prediction hard.

Thus, we are motivated to capture information of the target property of $\mathcal{T}_\omega$ during embedding learning. Specifically, we design a property-aware molecular encoder to transform the generic molecular embeddings to a property-aware space. Let $\mathcal{S}_\omega^c = \{(\mathbf{x}_{\omega,i}, y_{\omega,i}) | (\mathbf{x}_{\omega,i}, y_{\omega,i}) \in \mathcal{S}_\omega \text{ and } y_{\omega,i} = c\}$. We first compute the class prototype $\mathbf{c}_\omega^c$ for class $c \in \{0, 1\}$ as

$$\mathbf{c}_\omega^c = \frac{1}{|\mathcal{S}_\omega^c|} \sum\nolimits_{(\mathbf{x}_{\omega,i}, y_{\omega,i}) \in \mathcal{S}_\omega^c} \mathbf{g}_{\omega,i}. \tag{2}$$

Then, we take these class prototypes as the context information of task $\mathcal{T}_\omega$, and leverage them to obtain contextualized molecular embedding $\mathbf{b}_{\omega,i}$:

$$\mathbf{b}_{\omega,i} = [\mathtt{softmax}(\mathbf{C}_{\omega,i}\mathbf{C}_{\omega,i}^\top / \sqrt{d^g})\mathbf{C}_{\omega,i}]_{1:}, \tag{3}$$

where $\mathbf{C}_{\omega,i}^\top = [\mathbf{g}_{\omega,i}, \mathbf{c}_\omega^0, \mathbf{c}_\omega^1] \in \mathbb{R}^{d^g \times 3}$ and $[\cdot]_{l:}$ extracts the lth row vector which corresponds to $\mathbf{x}_{\omega,i}$. Here $\mathbf{b}_{\omega,i}$ is computed using scaled dot-product self-attention [58], such that each $\mathbf{g}_{\omega,i}$ can be compared with class prototypes in a dimension-wise manner. Finally, the property-aware molecular embedding $\mathbf{p}_{\omega,i}$ is obtained as

$$\mathbf{p}_{\omega,i} = \mathtt{MLP}_{\mathbf{W}_p}(\mathtt{concat}[\mathbf{g}_{\omega,i}, \mathbf{b}_{\omega,i}]). \tag{4}$$

Here, $\mathtt{MLP}_{\mathbf{W}_p}$ denotes multilayer perceptron (MLP) parameterized by $\mathbf{W}_p$, which is used to find a lower dimensional space that is more relevant to the target property of $\mathcal{T}_\omega$. The contextualized $\mathbf{p}_{\omega,i}$ is more predictive of the target property.

C. Query-Dependent Relation Graph Learning

Apart from property-aware information, the relationship among molecules also changes across properties. As shown in Fig. 1, two molecules with a shared property can be different from each other on another property [1], [59], [60]. Therefore, we further propose a query-dependent relation graph learning module to capture and leverage this property-aware relation graph among molecules, such that the molecular embeddings can be efficiently propagated and better learned.

The mth relation graph is established based on a set $\mathcal{V}_{\omega,m} = \mathcal{S}_\omega \cup (\mathbf{x}_{\omega,m}, y_{\omega,m})$, containing the support set and the mth query molecule. We will alternately estimate relation graphs among molecules, and refine the molecular embeddings on the learned relation graph for T times. At the τth iteration, we take each molecule in $\mathcal{V}_{\omega,m}$ as a node and construct a query-dependent graph $\mathcal{G}_{\omega,m}^{(\tau)}$, which is encoded by an adjacency matrix $\mathbf{A}_{\omega,m}^{(\tau)} \in \mathbb{R}^{(2K+1)\times(2K+1)}$. The element $[\mathbf{A}_{\omega,m}^{(\tau)}]_{pq}$ in $\mathbf{A}_{\omega,m}^{(\tau)}$ reflects the similarity between the pth and qth molecules in $\mathcal{V}_{\omega,m}$.

Let $\mathbf{h}_{\omega,m,p}^{(\tau)}$ be embedding of the pth molecule $\mathbf{x}_{\omega,p}$ in $\mathcal{V}_{\omega,m}$ during the τth iteration and we initialize $\mathbf{h}_{\omega,m,p}^{(0)}$ by its corresponding property-aware embedding $\mathbf{p}_{\omega,p}$. Then, we estimate

$\mathbf{A}_{\omega,m}^{(\tau)}$ using the current molecular embeddings $\mathbf{h}_{\omega,m,p}^{(\tau-1)}$'s, and the (p,q)th element of $\mathbf{A}_{\omega,m}^{(\tau)}$ is calculated as

$$[\mathbf{A}_{\omega,m}^{(\tau)}]_{pq}=\begin{cases}\texttt{MLP}_{\mathbf{W}_a}\left(\exp(-|\mathbf{h}_{\omega,m,p}^{(\tau-1)}-\mathbf{h}_{\omega,m,q}^{(\tau-1)}|)\right) & \text{if } p\neq q\\ 0 & \text{otherwise}\end{cases}, \tag{5}$$

where $\mathbf{W}_a$ is the parameter of this MLP. The resultant $\mathbf{A}_{\omega,m}^{(\tau)}$ is a dense matrix, which encodes a fully connected $\mathcal{G}_{\omega,m}^{(\tau)}$.

However, a query molecule only has K real neighbors in $\mathcal{G}_{\omega,m}^{(\tau)}$ in a 2-way K-shot task. For binary classification, choosing a wrong neighbor in the opposite class will heavily deteriorate the quality of molecular embeddings, especially when only one labeled molecule is provided per class. To avoid the interference of wrong neighbors, we further reduce $\mathcal{G}_{\omega,m}^{(\tau)}$ to a K-nearest neighbor (KNN) graph, where K is set to be exactly the same as the number of labeled molecules per class in $\mathcal{S}$. Denote $\mathcal{N}(\mathbf{v},K)$ the Kth largest element of vector $\mathbf{v}$. Then, we set

$$[\mathbf{B}_{\omega,m}^{(\tau)}]_{pq}=\begin{cases}[\mathbf{A}_{\omega,m}^{(\tau)}]_{pq} & \text{if } [\mathbf{A}_{\omega,m}^{(\tau)}]_{pq}\geq\mathcal{N}\left([\mathbf{A}_{\omega,m}^{(\tau)}]_{:q},K\right)\\ 0 & \text{otherwise}\end{cases}. \tag{6}$$

The values in $\mathbf{B}_{\omega,m}^{(\tau)}$ are normalized to range between 0 and 1, which is done by applying softmax function on each row $[\mathbf{B}_{\omega,m}^{(\tau)}]_{p:}$. This normalization can also be done by z-score, min-max and sigmoid normalization.

Then, we co-adapt each node embedding $\mathbf{h}_{\omega,m,p}^{(\tau)}$'s with respect to other node embeddings on this updated relation graph encoded by $\mathbf{B}_{\omega,m}^{(\tau)}$. Let $\mathbf{H}_{\omega,m}^{(\tau)}$ denote embeddings of all nodes in $\mathcal{G}_{\omega,m}^{(\tau)}$, where the ith column corresponds to $\mathbf{h}_{\omega,i}^{(\tau)}$. $\mathbf{H}_{\omega,m}^{(\tau)}$ is updated as

$$\mathbf{H}_{\omega,m}^{(\tau)}=\texttt{LeakyReLu}\left(\mathbf{W}_r\mathbf{H}_{\omega,m}^{(\tau-1)}\mathbf{B}_{\omega,m}^{(\tau)}\right), \tag{7}$$

where $\mathbf{W}_r$ is a learnable parameter. After T iterations, we return $\mathbf{h}_{\omega,m,p}=[\mathbf{H}_{\omega,m}^{(T)}]_{:p}$ as the final molecular embedding for the pth molecule in $\mathcal{V}_{\omega,m}$, and $\mathbf{B}_{\omega,m}=\mathbf{B}_{\omega,m}^{(T)}$ as the final optimized relation graph.

Finally, the class prediction, i.e., $\mathbf{z}_{\omega,m,p}$, of the pth molecule in $\mathcal{V}_{\omega,m}$ w.r.t. active/inactive is calculated as

$$\mathbf{z}_{\omega,m,p}=\texttt{softmax}\left(\mathbf{W}_c\cdot\mathbf{h}_{\omega,m,p}\right), \tag{8}$$

where $\mathbf{W}_c$ is a parameter.

D. Objective

We denote PAR model as $f_{\boldsymbol{\theta},\boldsymbol{\Phi}}$. In particular, $\boldsymbol{\theta}=\{\mathbf{W}_g,\mathbf{W}_a,\mathbf{W}_r\}$ includes the collection of parameters of graph-based molecular encoder and relation graph learning module that represent generic information. While $\boldsymbol{\Phi}=\{\mathbf{W}_p,\mathbf{W}_c\}$ includes the parameters of MLP to calculate $\mathbf{p}_{\omega,i}$ and the classifier that represent property-aware information.

Based on the framework of MAML introduced in Section II-B, we first propose new loss functions, then selectively update $\boldsymbol{\Phi}$ in inner loop update while simultaneously update $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ in outer loop update. Through the selective update strategy, the model can capture generic and property-aware information separately in the training procedure.

Denote $\mathbf{y}_{\omega,i}\in\mathbb{R}^2$ as a one-hot vector representing the ground-truth label of $\mathbf{x}_{\omega,i}\in\mathcal{S}_\omega$. $\mathbf{G}_{\omega,m}$ is the ground-truth relation graph of $\mathcal{G}_{\omega,m}^{(\tau)}$ with $[\mathbf{G}_{\omega,m}]_{pq}=1$ if ground-truth labels of the pth and qth samples in $\mathcal{V}_{\omega,m}$ are the same and 0 otherwise. First, we design the loss $\ell_{\mathcal{S}_\omega}$ evaluated on $\mathcal{S}_\omega$ with the mth query-dependent relation graph as follows

$$\begin{aligned}\ell_{\mathcal{S}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}},\mathbf{G}_{\omega,m},\mathbf{B}_{\omega,m})=&-\sum_{i=1}^{2K}\mathbf{y}_{\omega,i}^\top\cdot\log(\mathbf{z}_{\omega,m,i})\\&+\lambda\sum_{p=1}^{2K}\sum_{q=1}^{2K}\left([\mathbf{G}_{\omega,m}]_{pq}-[\mathbf{B}_{\omega,m}]_{pq}\right)^2,\end{aligned} \tag{9}$$

where λ is a hyper-parameter. The first term is the cross entropy loss for classification, and the second term is the specially designed neighbor alignment regularizer to penalize the selection of wrong neighbors among support molecules. Since there are M molecules in query set for each task and we need to construct relation graphs for each of them, the training loss for the in inner loop update is $L_{\mathcal{S}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}})=\sum_{m=1}^{M}\ell_{\mathcal{S}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}},\mathbf{G}_{\omega,m},\mathbf{B}_{\omega,m})$.

Denote $\mathbf{y}_{\omega,m}\in\mathbb{R}^2$ as a one-hot vector representing the ground-truth label of $\mathbf{x}_{\omega,m}\in\mathcal{Q}_\omega$. Following MAML, we introduce $\boldsymbol{\Phi}_\omega$ as

$$\boldsymbol{\Phi}_\omega=\boldsymbol{\Phi}-\alpha\nabla_{\boldsymbol{\Phi}}L_{\mathcal{S}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}}). \tag{10}$$

We use different strategies to update $\boldsymbol{\Phi}$ and $\boldsymbol{\theta}$. As $\boldsymbol{\theta}$ is considered as generic information, we keep it fixed in inner loop update. Variants of MAML such as ANIL and BOIL also consider selective update, but they only freeze head layer or body of the model in inner loop update. Here we further analyze the parameters of PAR and classify them into generic information $\boldsymbol{\theta}$ and property-aware information $\boldsymbol{\Phi}$. The selective update strategy with more elaborate parameter classification helps our PAR model learn better.

For outer loop update, we associate loss $\ell_{\mathcal{Q}_\omega}$ for the mth query sample as

$$\begin{aligned}\ell_{\mathcal{Q}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}_\omega},\mathbf{G}_{\omega,m},\mathbf{B}_{\omega,m})=&-\mathbf{y}_{\omega,m}^\top\cdot\log(\mathbf{z}_{\omega,m,2K+1})\\&+g\left(\mathbf{G}_{\omega,m},\mathbf{B}_{\omega,m}\right)+g\left(\mathbf{G}_{\omega,m}^\top,\mathbf{B}_{\omega,m}^\top\right),\end{aligned} \tag{11}$$

where $g(\mathbf{X},\mathbf{Y})=\mu\sum_{p=1}^{2K}([\mathbf{X}]_{p(2K+1)}-[\mathbf{Y}]_{p(2K+1)})^2$, and μ is a hyper-parameter. Again, the first term is the cross entropy loss for classification. The second and the third term are the regularizer that penalizes wrong relation prediction between support molecules and query molecules. Thus, the training loss evaluated on $\mathcal{Q}_\omega$ in outer loop update is $L_{\mathcal{Q}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}_\omega})=\sum_{m=1}^{M}\ell_{\mathcal{Q}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}_\omega},\mathbf{G}_{\omega,m},\mathbf{B}_{\omega,m})$. Finally, the meta-trained parameters $\boldsymbol{\theta}^*$ and $\boldsymbol{\Phi}^*$ can be obtained by $\min_{\boldsymbol{\theta},\boldsymbol{\Phi}}\sum_\omega L_{\mathcal{Q}_\omega}(f_{\boldsymbol{\theta},\boldsymbol{\Phi}_\omega})$, which can be trained by gradient descent as Section II-B1.

Algorithm 1: Meta-Training Procedure for PAR.

1: initialize $\boldsymbol{\theta} = \{\mathbf{W}_g, \mathbf{W}_a, \mathbf{W}_r\}$ and $\boldsymbol{\Phi} = \{\mathbf{W}_p, \mathbf{W}_c\}$ randomly; if a pretrained molecular encoder is available, take its parameter as $\mathbf{W}_g$;
2: **while** not done **do**
3: sample a set of tasks $\mathcal{Z}$;
4: **for** each task $\mathcal{T}_\omega \in \mathcal{Z}$ **do**
5: obtain molecular embedding $\mathbf{g}_{\omega,i}$ for each $\mathbf{x}_{\omega,i}$ by graph-based molecular encoder;
6: adapt $\mathbf{g}_{\omega,i}$ to be property-aware $\mathbf{p}_{\omega,i}$ by (4);
7: **for** $m = 1, \ldots, M$ **do**
8: initialize node embeddings $\mathbf{h}^{(0)}_{\omega,m,p}$ by $\mathbf{p}_{\omega,p}$;
9: **for** $\tau = 1, \ldots, T$ **do**
10: estimate dense graph $\mathbf{A}^{(\tau)}_{\omega,m}$ using $\mathbf{h}^{(\tau-1)}_{\omega,m,p}$ by (5);
11: obtain sparsified graph $\mathbf{B}^{(\tau)}_{\omega,m}$ by (6);
12: refine $\mathbf{h}^{(\tau)}_{\omega,m,p}$ by (7);
13: **end for**
14: obtain class prediction $\mathbf{z}_{\omega,m,p}$ by (8);
15: **end for**
16: fine-tune $\boldsymbol{\Phi}$ as $\boldsymbol{\Phi}_\omega$ by (10);
17: **end for**
18: update $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ by gradient descent;
19: **end while**

E. The Complete Algorithm

The meta-training procedure of PAR is shown in Algorithm 1. Line 5–6 correspond to property-aware molecular encoder which captures property-related information (Section II-I-B). Line 8–13 correspond to query-dependent relation graph learning which facilitates learning molecular embeddings and propagating labels among similar molecules (Section III-C). Line 15–17 correspond to the meta-learning strategy built on MAML with selective update approach (Section III-D). In the meta-testing step, meta-trained $\boldsymbol{\theta}^*$ and $\boldsymbol{\Phi}^*$ are used as the initialized parameters. We first fine-tune $\boldsymbol{\Phi}^*$ on $\mathcal{S}_0$ and fix $\boldsymbol{\theta}^*$, then use the updated model to predict the property of molecules in $\mathcal{Q}_0$. Detailed meta-testing procedure is in Appendix B.

IV. Transferable PAR

Apart from the lack of labeled data as discussed in Section II-C2, distribution shift, where the training distribution is not identical with the test distribution, is also very common in AI-aided drug discovery [26], [27]. Such a shift in the task distribution can significantly deteriorate the performance of a learning model [61]. For example, when designing drugs for a sudden epidemic, we need to handle new molecules targeted for the disease. Thus, when trained with molecules at hand, we need to deal with not only a few labeled data but also the possible distribution shift.

In the sequel, we formulate such a prediction task as a transferable FSL MPP problem (Section IV-A) To solve this problem, we extend PAR as T-PAR (Fig. 3), which is short for transferable PAR. T-PAR continues using the property-aware molecular encoder in PAR (Fig. 3(b)), which can obtain property-aware molecular embeddings for the tasks, i.e., $\mathbf{p}_{s,i}$ for $\mathcal{T}_s$ and $\mathbf{p}_{t,i}$ for $\mathcal{T}_t$, through (2)-(4) in Section III-B. To bridge the gap between source and target domain, T-PAR introduces two key designs to solve transferable few-shot MPP problems: (i) joint sampling strategy (Fig. 3(a)) which involves molecules from source and target domain in the training procedure together such that the models can better encode target domain (Section IV-B), and (ii) joint relation graph learning (Fig. 3(c)) which models the relationship between source and target domain so that the molecular embeddings can be effectively propagated between source and target domain (Section IV-C). Finally, we describe the learning objective of T-PAR (Section IV-D) and provide the complete algorithm (Section IV-E).

A. Problem Formulation

In the transferable FSL MPP setting, we need to tackle with 2-way K-shot MPP tasks from the target domain given labeled data mainly from the source domain (source domain tasks and target domain tasks do not have class overlap). Following Section II-B, we formulate the new problem as follows. The formation of the target domain task is $\mathcal{T}_0 = \{\mathcal{S}_0, \mathcal{Q}_0\}$. However, the given auxiliary data is quite different from those in traditional FSL setting. Here we are given an auxiliary base training set $\mathcal{D}$ from source domain with sufficient numbers of labeled samples and a set of MPP tasks $\mathcal{Z}_l = \{\mathcal{T}_t\}$ from target domain with only a few labeled samples. For example, molecules in $\mathcal{T}_t$ are a few candidate drugs for the new disease.

Note that none of existing MPP methods discussed in Section II-C2 can handle the transferable FSL setting here. First, few-shot MPP methods, e.g., IterRefLSTM [3], Meta-MGNN [8] and PAR, make the implicit assumption that the molecules from meta-training tasks and those from meta-testing tasks follow the same distribution. The performance of the models will inevitably degrade in these practical scenarios. Besides, transferable MPP methods including ChemNet [54] and DTCR [2] can handle the distribution shift, but they cannot generalize to new tasks with a few labeled samples. Finally, FSL methods that can be used in the transfer setting here, e.g., BOIL [62] and DAPN [40], do not consider the particularity of MPP problems, which leads to their unsatisfactory performance.

In the sequel, to easily identify notations extended from PAR to T-PAR with similar functionality, we mark them by $\hat{\cdot}$, e.g., $\mathbf{A}$ in PAR v.s. $\hat{\mathbf{A}}$ in T-PAR.

B. Joint Sampling

In PAR and other ordinary MPP methods [3], [8], they only use tasks sampled from source domain in the training process. Without obtaining any target domain data in the training stage, these methods are unable to generalize to the target domain in the testing stage, so they cannot tackle transferable few-shot MPP problems. The given labeled target domain tasks are very important for the MPP model to obtain knowledge from the target domain in the training process.

In order to fully make use of the labeled target domain data in the training stage, we propose a joint sampling strategy, which simultaneously obtains a meta-training task from source and

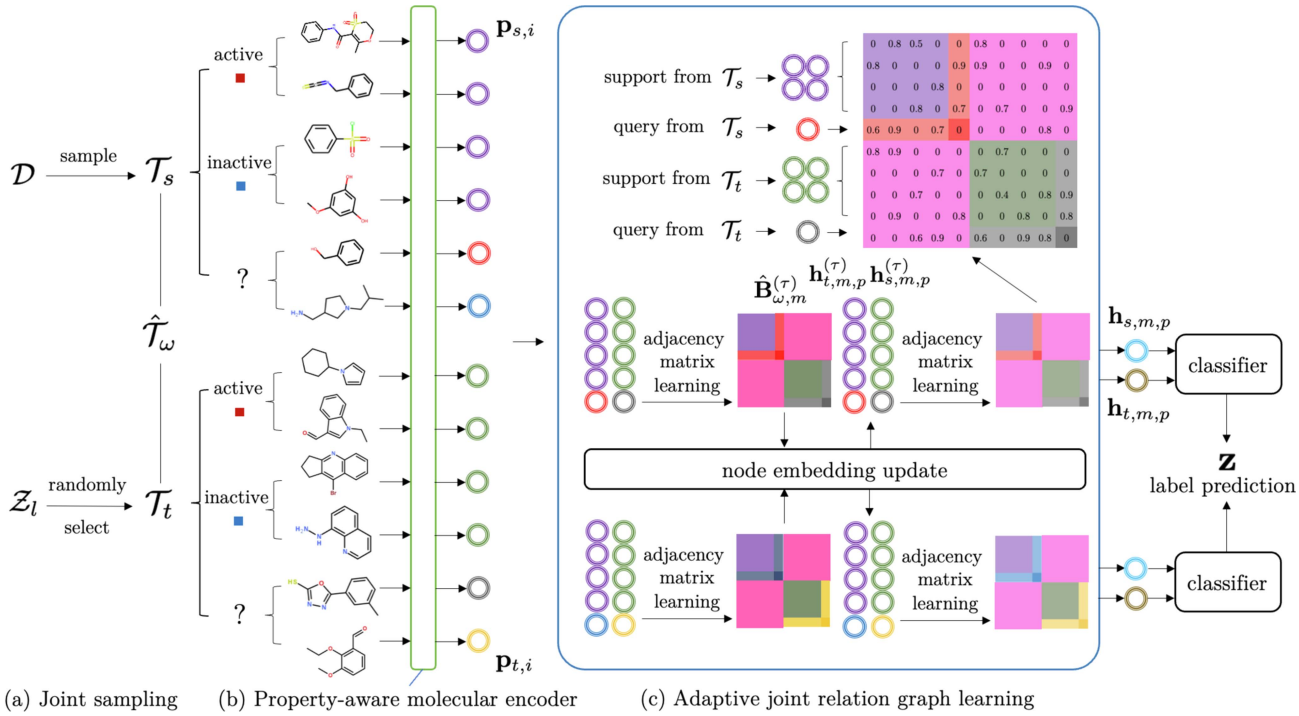


Fig. 3. The proposed T-PAR can be mainly divided into three parts: (a) Joint sampling where we sample $\mathcal{T}_s$ from $\mathcal{D}$ and randomly select $\mathcal{T}_t$ from $\mathcal{Z}_l$, and two 2-way 2-shot tasks form a task pair $\hat{\mathcal{T}}_\omega$; (b) Property-aware molecular encoder where we continue to use the property-aware molecular encoder in PAR to obtain property-aware molecular embeddings $\mathbf{p}_{s,i}$ and $\mathbf{p}_{t,i}$ for molecules in the two tasks; and (c) Joint relation graph learning, where we put molecules from $\mathcal{T}_s$ and $\mathcal{T}_t$ into the same graph to model the relationship between source and target domain, iteratively update their embeddings through the joint relation graph as $\mathbf{h}_{s,m,p}$ and $\mathbf{h}_{t,m,p}$, which are taken as the final molecular embeddings and used for class prediction.

target domain, i.e., sample $\mathcal{T}_s$ from $\mathcal{D}$ as source domain task and randomly select $\mathcal{T}_t$ from $\mathcal{Z}_l$ as target domain task. The two sample tasks form a task pair $\hat{\mathcal{T}}_\omega = \{\mathcal{T}_s, \mathcal{T}_t\}$. Repeat the sampling process we can obtain a set of task pairs $\hat{\mathcal{Z}} = \{\hat{\mathcal{T}}_\omega\}$ for the training process. Through the joint sampling strategy, the two tasks from different domains are able to simultaneously participate in the meta-training step and the adapt step of the model, which helps the model to learn knowledge from target domain better.

C. Joint Relation Graph Learning

Apart from introducing target domain data to the training process through joint training strategy, it is more important to model the relationship between source domain and target domain in the training process. To achieve this goal, we consider putting molecules in $\mathcal{T}_s$ and $\mathcal{T}_t$ into the same relation graph so that molecular embeddings can be propagated between source and target domain.

Each time we collect all support molecules and a query molecule in each of the two tasks to construct a joint relation graph. Specifically, assume the mth joint relation graph samples the j_1th query molecule $(\mathbf{x}_{s,j_1}, y_{s,j_1})$ in $\mathcal{T}_s$ and the j_2th query molecule $(\mathbf{x}_{t,j_2}, y_{t,j_2})$ in $\mathcal{T}_t$, the relation graph is established based on a set $\hat{\mathcal{V}}_{\omega,m} = \mathcal{V}_{s,m} \cup \mathcal{V}_{t,m}$, where $\mathcal{V}_{s,m} = \mathcal{S}_s \cup (\mathbf{x}_{s,j_1}, y_{s,j_1})$ and $\mathcal{V}_{t,m} = \mathcal{S}_t \cup (\mathbf{x}_{t,j_2}, y_{t,j_2})$. For each set $\hat{\mathcal{V}}_{\omega,m}$, we will iteratively update the molecular embeddings on the learned jointly relation graph for T times.

At the τth iteration, we take each molecule in $\hat{\mathcal{V}}_{\omega,m}$ as a node and calculate an adjacency matrix $\hat{\mathbf{A}}^{(\tau)}_{\omega,m} \in \mathbb{R}^{(4K+2)\times(4K+2)}$ to encode the joint relation graph $\mathcal{G}^{(\tau)}_{\omega,m}$ among these nodes. We denote $\mathbf{h}^{(\tau)}_{s,m,p}$ the embedding of the pth molecule $\mathbf{x}_{s,p}$ in $\mathcal{V}_{s,m}$ during the τth iteration and initialize $\mathbf{h}^{(0)}_{s,m,p}$ by its corresponding property-aware embedding $\mathbf{p}_{s,p}$ ($\mathbf{h}^{(\tau)}_{t,m,p}$ and $\mathbf{h}^{(0)}_{t,m,p}$ are defined similarly). Then, we can estimate $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ using the current molecular embeddings and here we split $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ into four parts

$$\hat{\mathbf{A}}^{(\tau)}_{\omega,m} = \begin{bmatrix} \hat{\mathbf{A}}^{(\tau)}_{ss,m} & \hat{\mathbf{A}}^{(\tau)}_{st,m} \\ \hat{\mathbf{A}}^{(\tau)}_{ts,m} & \hat{\mathbf{A}}^{(\tau)}_{tt,m} \end{bmatrix}, \tag{12}$$

where $\hat{\mathbf{A}}^{(\tau)}_{ss,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{tt,m}$ control the molecular embedding propagation among molecules in the same task while $\hat{\mathbf{A}}^{(\tau)}_{st,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{ts,m}$ control the molecular embedding propagation between molecules in different tasks. The similarity between the pth and qth molecules in $\hat{\mathcal{V}}_{\omega,m}$ is represented by the (p,q)th element $[\hat{\mathbf{A}}^{(\tau)}_{\omega,m}]_{pq}$ in $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$, its value can also be computed using $\mathbf{h}^{(\tau-1)}_{t,m,p}$ and $\mathbf{h}^{(\tau-1)}_{t,m,q}$ following (5). In addition, here in (5) we use $\mathbf{W}_a$ to calculate similarity between molecules from the same domain (i.e., $\hat{\mathbf{A}}^{(\tau)}_{ss,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{tt,m}$) and use another $\mathbf{W}_b$ to calculate similarity between molecules from different domains (i.e., $\hat{\mathbf{A}}^{(\tau)}_{st,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{ts,m}$).

However, since $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ jointly models molecules from both source and target domain, we take different thresholds to sparsify different blocks in $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$. Specifically, we again follow (6), but different thresholds are used.

- Within domain, i.e., $\hat{\mathbf{A}}^{(\tau)}_{ss,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{tt,m}$. Since the two blocks only model interactions between molecules in the same domain, we use $\mathcal{N}([\hat{\mathbf{A}}^{(\tau)}_{ss,m}]_{:q}, K)$ and $\mathcal{N}([\hat{\mathbf{A}}^{(\tau)}_{tt,m}]_{:q}, K)$ as thresholds for $\hat{\mathbf{A}}^{(\tau)}_{ss,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{tt,m}$ respectively.
- Cross domain, i.e., $\hat{\mathbf{A}}^{(\tau)}_{st,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{ts,m}$. We take the same $\mathcal{N}([\hat{\mathbf{A}}^{(\tau)}_{ss,m}]_{:q}, K)$ as the threshold as these two blocks both control the embedding propagation between molecules in $\mathcal{T}_t$ and those in $\mathcal{T}_s$.

Thus, we will obtain the sparsified relation graph as

$$\hat{\mathbf{B}}^{(\tau)}_{\omega,m} = \begin{bmatrix} \hat{\mathbf{B}}^{(\tau)}_{ss,m} & \hat{\mathbf{B}}^{(\tau)}_{st,m} \\ \hat{\mathbf{B}}^{(\tau)}_{ts,m} & \hat{\mathbf{B}}^{(\tau)}_{tt,m} \end{bmatrix}, \tag{13}$$

where each block is sparsified from the corresponding part in $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$. Then, we co-adapt each node embedding on this updated relation graph encoded by $\hat{\mathbf{B}}^{(\tau)}_{\omega,m}$.

Let $\mathbf{H}^{(\tau)}_{s,m}$ denotes all node embeddings in $\mathcal{V}_{s,m}$ collectively where the pth column corresponds to $\mathbf{h}^{(\tau)}_{s,m,p}$ and $\mathbf{H}^{(\tau)}_{t,m}$ is defined similarly. We obtain $\hat{\mathbf{H}}^{(\tau)}_{\omega,m} = [\mathbf{H}^{(\tau)}_{s,m}, \mathbf{H}^{(\tau)}_{t,m}]$ and update the molecular embeddings

$$\hat{\mathbf{H}}^{(\tau)}_{\omega,m} = \texttt{LeakyReLu}(\mathbf{W}_r \hat{\mathbf{H}}^{(\tau-1)}_{\omega,m} \hat{\mathbf{B}}^{(\tau)}_{\omega,m}), \tag{14}$$

where $\mathbf{W}_r$ is a learnable parameter. After T iterations, we return $\mathbf{h}_{s,m,p} = [\mathbf{H}^{(T)}_{s,m}]_{:p}$ and $\mathbf{h}_{t,m,p} = [\mathbf{H}^{(T)}_{t,m}]_{:p}$ as the final molecular embeddings for molecules in $\hat{\mathcal{V}}_{\omega,m}$, $\hat{\mathbf{B}}_{s,m} = \hat{\mathbf{B}}^{(T)}_{ss,m}$ and $\hat{\mathbf{B}}_{t,m} = \hat{\mathbf{B}}^{(T)}_{tt,m}$, as the final relation graph. We do not return $\hat{\mathbf{B}}^{(T)}_{st,m}$ and $\hat{\mathbf{B}}^{(T)}_{ts,m}$ in $\hat{\mathbf{B}}^{(T)}_{\omega,m}$ because there is no ground-truth relation between molecules from different tasks. Finally, we can obtain the MPP $\mathbf{z}_{s,m,p}$ for $\mathbf{h}_{s,m,p}$ and $\mathbf{z}_{t,m,p}$ for $\mathbf{h}_{t,m,p}$ through the classifier as (8) for each task.

D. Objective

Here, we extend selective update strategy in Section III-D to transfer learning scenario. Still, we denote T-PAR model as $\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}$, where $\hat{\boldsymbol{\theta}} = \{\mathbf{W}_g, \mathbf{W}_a, \mathbf{W}_b, \mathbf{W}_r\}$ represent generic information and $\hat{\boldsymbol{\Phi}} = \{\mathbf{W}_p, \mathbf{W}_c\}$ represent property-aware information.

Denote $\hat{\mathbf{G}}_{s,m}$ as the ground-truth relation graph of nodes in $\mathcal{V}_{s,m}$ with $[\hat{\mathbf{G}}_{s,m}]_{pq} = 1$ if ground-truth labels of the pth and qth samples in $\mathcal{V}_{s,m}$ are the same and 0 otherwise. $\hat{\mathbf{G}}_{t,m}$ can be defined similarly. As we separately calculate the MPP ($\mathbf{z}_{s,m,p}$ and $\mathbf{z}_{t,m,p}$) and the relation graph ($\hat{\mathbf{B}}_{s,m}$ and $\hat{\mathbf{B}}_{t,m}$) for molecules in $\mathcal{T}_s$ and $\mathcal{T}_t$, the loss within each task can be obtained as Section III-D. Specifically, the loss function $\ell_{\mathcal{S}_s}$ and $\ell_{\mathcal{S}_t}$ for the mth joint relation graph can be designed as (9). As there are M query molecules in each of the two tasks, we construct M^2 joint relation graphs. Considering all the joint relation graphs, we can calculate the training loss the inner loop update as $\hat{L}_{\mathcal{S}_\omega}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}) = \sum_{m=1}^{M^2}(\ell_{\mathcal{S}_s}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}, \hat{\mathbf{G}}_{s,m}, \hat{\mathbf{B}}_{s,m}) + \ell_{\mathcal{S}_t}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}, \hat{\mathbf{G}}_{t,m}, \hat{\mathbf{B}}_{t,m}))$, where we simultaneously consider the loss from source and target domain. Next, we selectively update the property-aware information $\hat{\boldsymbol{\Phi}}$ to $\hat{\boldsymbol{\Phi}}_\omega$ as

$$\hat{\boldsymbol{\Phi}}_\omega = \hat{\boldsymbol{\Phi}} - \alpha \nabla_{\hat{\boldsymbol{\Phi}}} \hat{L}_{\mathcal{S}_\omega}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}), \tag{15}$$

and fix the generic information $\hat{\boldsymbol{\theta}}$ in inner loop update.

Then, we design the loss functions $\ell_{\mathcal{Q}_s}$ and $\ell_{\mathcal{Q}_t}$ for query samples in the mth joint relation graph as (11), through which we can propose the training loss evaluated on for $\hat{\mathcal{T}}_\omega$ in outer loop update as $\hat{L}_{\mathcal{Q}_\omega}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}_\omega}) = \sum_{m=1}^{M^2}(\ell_{\mathcal{Q}_s}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}, \hat{\mathbf{G}}_{s,m}, \hat{\mathbf{B}}_{s,m}) + \ell_{\mathcal{Q}_t}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}}, \hat{\mathbf{G}}_{t,m}, \hat{\mathbf{B}}_{t,m}))$. Considering all the task pairs in $\hat{\mathcal{Z}}$, we obtain the meta-trained parameters $\hat{\boldsymbol{\theta}}^*$ and $\hat{\boldsymbol{\Phi}}^*$ through $\min_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}} \sum_\omega \hat{L}_{\mathcal{Q}_\omega}(\hat{f}_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\Phi}}_\omega})$, where $\hat{\boldsymbol{\Phi}}_\omega$ can be obtained by (15).

E. The Complete Algorithm

Meta-training procedure of T-PAR is shown in Algorithm 2. Compared to PAR, T-PAR considers a practical scenario, where the training data follows a different data distribution with testing data. In the algorithm, line 3 corresponds to the joint sampling strategy which involve molecules from source and target domain in the training procedure together (Section IV-B). Line 5-6 correspond to property-aware molecular encoder (Section II-I-B). Line 10-12 correspond to joint relation graph learning which model the relationship between source and target domain (Section IV-C). Line 16-18 correspond to training and inference step (Section IV-D). With the proposed joint sampling and relation graph learning components, T-PAR can perform better in transferable few-shot MPP problems.

In the meta-testing procedure (details are in Appendix B), we only construct relation graph among molecules in $\mathcal{T}_0$ without task pairs like $\hat{\mathcal{T}}_\omega$. The meta-testing procedure of T-PAR uses $\hat{\boldsymbol{\theta}}^*$ and $\hat{\boldsymbol{\Phi}}^*$ meta-trained by T-PAR to initialize the model, adopts selective update strategy to fine-tune the model on the support set $\mathcal{S}_0$ and gets the MPP accuracy on the query set $\mathcal{Q}_0$.

V. Experiments

A. Few-Shot Setting

1) Setup: Datasets: We perform experiments[1] on widely used few-shot MPP datasets from MoleculeNet[2] [49], whose statistics are provided in Table I. Briefly, (i) Tox21 [25] contains assays each measuring the human toxicity of a biological target; (ii) SIDER [59] records the side effects for compounds used in marketed medicines, where 5868 side effects are grouped into 27 categories as in [3], [8]; (iii) MUV [1] is designed to validate virtual screening where active molecules are chosen to be structurally distinct from each another; and (iv) ToxCast [60] is a collection of compounds with toxicity labels which are obtained via high-throughput screening. Tox21, SIDER and MUV have public task splits provided by [3], which are adopted in this paper. For ToxCast, we randomly select 450 tasks for meta-training and use the rest for meta-testing.

[1]The codes of PAR are available at https://github.com/tata1661/PAR-NeurIPS21. The codes of T-PAR will be made public upon publication.

[2]All datasets are downloaded from http://moleculenet.ai/.

Algorithm 2: Meta-Training Procedure for T-PAR.

initialize $\hat{\boldsymbol{\theta}} = \{\mathbf{W}_g, \mathbf{W}_a, \mathbf{W}_b, \mathbf{W}_r\}$ and $\hat{\boldsymbol{\Phi}} = \{\mathbf{W}_p, \mathbf{W}_c\}$ randomly; if a pretrained molecular encoder is available, take its parameter as $\mathbf{W}_g$;
while not done **do**
  sample a set of tasks pairs $\hat{\mathcal{Z}}$;
  for each task pair $\hat{\mathcal{T}}_\omega \in \hat{\mathcal{Z}}$ **do**
    obtain molecular embedding $\mathbf{g}_{s,i}$ and $\mathbf{g}_{t,i}$ by a graph-based molecular encoder;
    adapt $\mathbf{g}_{s,i}$ and $\mathbf{g}_{t,i}$ to be property-aware $\mathbf{p}_{s,i}$ and $\mathbf{p}_{t,i}$ by (2) to (4);
    for $m = 1, \ldots, M^2$ **do**
      initialize node embeddings $\mathbf{h}^{(0)}_{s,m,p}$ and $\mathbf{h}^{(0)}_{t,m,p}$ by $\mathbf{p}_{s,p}$ and $\mathbf{p}_{t,p}$;
      for $\tau = 1, \ldots, T$ **do**
        estimate dense graph $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ by (12);
        obtain sparsified graph $\hat{\mathbf{B}}^{(\tau)}_{\omega,m}$ by (13);
        refine $\mathbf{h}^{(\tau)}_{s,m,p}$ and $\mathbf{h}^{(\tau)}_{t,m,p}$ by (14);
      end for
      obtain class prediction $\mathbf{z}_{s,m,p}$ and $\mathbf{z}_{t,m,p}$ by (8);
    end for
    fine-tune $\hat{\boldsymbol{\Phi}}$ as $\hat{\boldsymbol{\Phi}}_\omega$ by (15);
  end for
  update $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\Phi}}$ by gradient descent;
end while

TABLE I
BENCHMARK DATASETS USED IN THIS PAPER

Dataset	Tox21	SIDER	MUV	ToxCast
# Compounds	8014	1427	93127	8615
# Tasks	12	27	17	617
# Meta-Training Tasks	9	21	12	450
# Meta-Testing Tasks	3	6	5	167

Evaluation Metrics: Following [8], [24], we evaluate the binary classification performance by ROC-AUC scores calculated on the query set of each meta-testing task. We run experiments for ten times with different random seeds, and report the mean and standard deviations of ROC-AUC computed over all meta-testing tasks.

Baselines: Following [8], [24], we use RDKit [63] to build molecular graphs from raw SMILES, and to extract atom features (atom number and chirality tag) and bond features (bond type and bond direction). For FSL methods with graph-based molecular encoder learned from scratch, we use GIN [21] as the graph-based molecular encoder to extract molecular embeddings, which include

- Siamese [64] which learns dual convolutional neural networks to identify whether the input molecule pairs are from the same class;
- ProtoNet [13] which assigns each query molecule with the label of its nearest class prototype;
- MAML [16] which adapts the meta-learned parameters to new tasks via gradient descent;
- ANIL [36] and BOIL [62] which are variants of MAML;
- TPN [65] which conducts label propagation on relation graphs with rescaled edge weights under transductive setting;
- EGNN [66] which learns to predict edge labels of relation graphs;
- IterRefLSTM [3] which adapts Matching Networks [23] to handle MPP tasks;
- and the proposed PAR (Algorithm 1).

Besides, for FSL methods leveraging pretrained graph-based molecular encoder, we consistently use the pretrained GIN provided by the authors of [24], which include

- Pre − GNN [24] which uses graph-level and node-level self-supervised tasks to fine-tune the model on support sets;
- Meta − MGNN [8] which optimizes the MPP task with self-supervised bond reconstruction and atom type prediction tasks;
- Pre − PAR which is our PAR (Algorithm 1) equipped with pretrained molecular encoder.

GROVER [9] is not compared as it uses a different set of atom and bond features. We use results of Siamese and IterRefLSTM reported in [3] as their codes are not available. For the other methods compared with PAR and Pre-PAR, we use public codes from the respective authors. Hyper-parameter settings are in Appendix C.

2) Performance Comparison: Table II shows the results. Results of Siamese and IterRefLSTM on ToxCast are not provided because the two methods lack codes and are not evaluated on ToxCast before. As can be seen, methods using the pretrained graph-based molecular encoder generally perform better than methods using graph-based molecular encoders learned from scratch. This reveals that pretrained graph-based molecular encoder, which encodes rich generic molecular information via pretraining, indeed provides better molecular embeddings. Among all methods, Pre-PAR consistently obtains the best performance, while PAR obtains the best performance among methods using graph-based molecular encoders learned from scratch. These outperforming results can be attributed to the combination of metric-based and optimization-based method in the design of PAR method. In terms of average improvement, PAR obtains significantly better performance than the best baseline learned from scratch (e.g. EGNN) by 1.59%, and Pre-PAR is better than the best baseline with pretrained molecular encoders (e.g. Pre-GNN) by 1.49%. In addition, we observe that FSL methods that learn relation graphs (i.e., TPN, EGNN and PAR) obtain better performance than the classical Siamese and MAML.

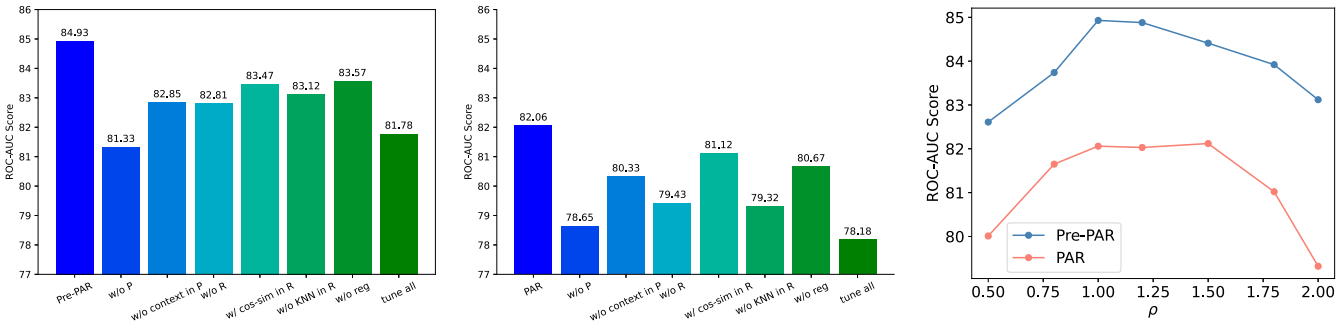
3) Varying Components in PAR: We further compare PAR (and Pre-PAR) with the following variants following control variates method:

- w/oP which removes property-aware embedding function;
- w/ocontextinP which removes contextualized molecular embedding $\mathbf{b}_{\omega,i}$ in (4);
- w/oR which removes query-dependent relation graph learning;
- w/cos − siminR which obtain the adjacency matrix $[\mathbf{A}^{(\tau)}_{\omega,m}]_{pq} = (\mathbf{p}^{(\tau)}_{\omega,m,p})^\top \mathbf{p}^{(\tau)}_{\omega,m,q} / (\|\mathbf{p}^{(\tau)}_{\omega,m,p}\|_2 \|\mathbf{p}^{(\tau)}_{\omega,m,q}\|_2)$

TABLE II
ROC-AUC SCORES ON BENCHMARK MOLECULAR PROPERTY PREDICTION DATASETS

Method	Tox21		SIDER		MUV		ToxCast	
	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot
Siamese	$80.40_{(0.35)}$	$65.00_{(1.58)}$	$71.10_{(4.32)}$	$51.43_{(3.31)}$	$59.96_{(5.13)}$	$50.00_{(0.17)}$	-	-
ProtoNet	$74.98_{(0.32)}$	$65.58_{(1.72)}$	$64.54_{(0.89)}$	$57.50_{(2.34)}$	$65.88_{(4.11)}$	$58.31_{(3.18)}$	$63.70_{(1.26)}$	$56.36_{(1.54)}$
MAML	$80.21_{(0.24)}$	$75.74_{(0.48)}$	$70.43_{(0.76)}$	$67.81_{(1.12)}$	$63.90_{(2.28)}$	$60.51_{(3.12)}$	$66.79_{(0.85)}$	$65.97_{(5.04)}$
ANIL	$80.32_{(0.17)}$	$76.63_{(0.21)}$	$70.09_{(0.41)}$	$68.61_{(0.85)}$	$64.54_{(2.89)}$	$61.16_{(4.08)}$	$66.76_{(0.47)}$	$65.74_{(1.36)}$
BOIL	$80.53_{(0.20)}$	$76.75_{(0.11)}$	$70.52_{(0.42)}$	$67.97_{(0.93)}$	$63.42_{(2.09)}$	$60.13_{(2.94)}$	$66.79_{(0.85)}$	$66.97_{(1.43)}$
TPN	$76.05_{(0.24)}$	$60.16_{(1.18)}$	$67.84_{(0.95)}$	$62.90_{(1.38)}$	$65.22_{(5.82)}$	$50.00_{(0.51)}$	$62.74_{(1.45)}$	$50.01_{(0.05)}$
EGNN	$81.21_{(0.16)}$	$79.44_{(0.22)}$	$72.87_{(0.73)}$	$70.79_{(0.95)}$	$65.20_{(2.08)}$	$62.18_{(1.76)}$	$63.65_{(1.57)}$	$61.02_{(1.94)}$
IterRefLSTM	$81.10_{(0.17)}$	$80.97_{(0.10)}$	$69.63_{(0.31)}$	$71.73_{(0.14)}$	$49.56_{(5.12)}$	$48.54_{(3.12)}$	-	-
PAR	$82.06_{(0.12)}$	$80.46_{(0.13)}$	$74.68_{(0.31)}$	$71.87_{(0.48)}$	$66.48_{(2.12)}$	$64.12_{(1.18)}$	$69.72_{(1.63)}$	$67.28_{(2.90)}$
Pre-GNN	$82.14_{(0.08)}$	$81.68_{(0.09)}$	$73.96_{(0.08)}$	$73.24_{(0.12)}$	$67.14_{(1.58)}$	$64.51_{(1.45)}$	$73.68_{(0.74)}$	$72.90_{(0.84)}$
Meta-MGNN	$82.97_{(0.10)}$	$82.13_{(0.13)}$	$75.43_{(0.21)}$	$73.36_{(0.32)}$	$68.99_{(1.84)}$	$65.54_{(2.13)}$	$74.04_{(0.34)}$	$70.89_{(0.05)}$
Pre-PAR	$84.93_{(0.11)}$	$83.01_{(0.09)}$	$78.08_{(0.16)}$	$74.46_{(0.29)}$	$69.96_{(1.37)}$	$66.94_{(1.12)}$	$75.12_{(0.84)}$	$73.63_{(1.00)}$

The best results (the pairwise t-test with 95% confidence) are highlighted in gray. Methods using the pretrained graph-based molecular encoder are in green.



(a) Varying components in Pre-PAR.

(b) Varying components in PAR.

(c) Varying the number of elements retained in each column in matrix $\mathbf{A}$.

Fig. 4. Experiments for varying components in PAR on 10-shot tasks from Tox21.

using cosine similarity, then calculates (6) and (7) as in PAR;

- w/oKNNinR which reduces $\mathcal{G}_\omega$ to KNN graph, i.e. without sparsification of $\mathbf{B}$;
- w/oreg which removes the neighbor alignment regularizer in (9) and (11);
- tuneall which fine-tunes all parameters on line 15 of Algorithm 1.

These variants cover all components of training PAR without overlapping functionalities.

Fig. 4 (a)-(b) show the results of PAR and Pre-PAR obtained on 10-shot tasks respectively. Again, Pre-PAR obtains better performance than PAR due to a better start. PAR and Pre-PAR outperform their variants. The removal of any component leads to significant performance drop. In particular, the performance gain of PAR and Pre-PAR with respect to w/ cos-sim in R validates the necessity of learning a similarity function from the data rather than using the fixed cosine similarity. We also try to iterate the estimation of relation graph constructed by cosine similarity, but observe a performance drop given more iterations.

In (9) in PAR, the top-K elements in each column in matrix $\mathbf{A}$ are retained to obtain the sparsified matrix $\mathbf{B}$. We further conduct experiment to retain top-$\lceil \rho K \rceil$ elements in each column in matrix $\mathbf{A}$ (we set ρ as an adaptable parameter ranging from 0.5 to 2). The results on 10-shot setting are shown in Fig. 4 (c), which shows that the best performance can be obtained when ρ is set between 1 and 1.5.

4) Using Other Graph-Based Molecular Encoders: In the experiments, we use GIN and its pretrained version as the molecular encoder. However, as introduced in Section III-B, our PAR is compatible with any existing graph-based molecular encoder introduced in Section II-A. Here, we consider the following popular choices as the encoder to output $\mathbf{g}_{\omega,i}$: GIN [21], GCN [51], GraphSAGE [20] and GAT [22], which are either learned from scratch or pretrained. We compare the proposed PAR and Pre-PAR with simply fine-tuning the encoder on support sets (denote as GNN or Pre-GNN if encoder is pretrained).

Fig. 5 shows the results on 10-shot tasks. GIN is the best graph-based molecular encoder among the four chosen GNNs. PAR outperforms the fine-tuned GNN consistently. This validates the effectiveness of the property-aware molecular encoder and the relation graph learning module. We further notice that using pretrained encoders can improve the performance except for GAT, which is also observed in [24]. Although using pretrained graph-based molecular encoders can improve the performance

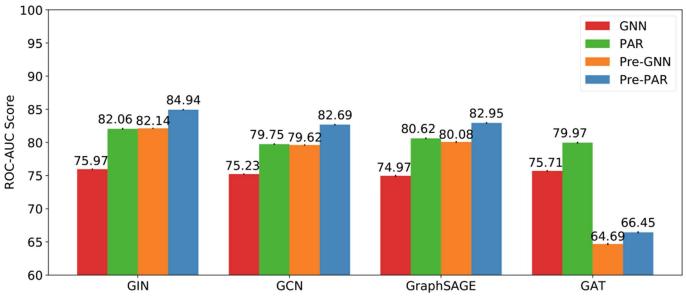
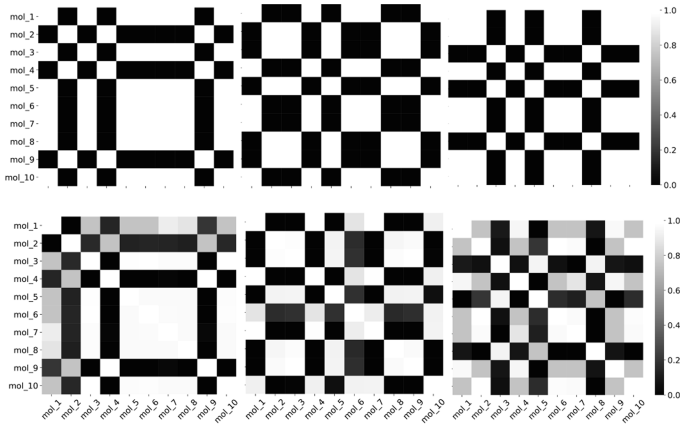


Fig. 5. ROC-AUC scores on 10-shot tasks from Tox21 using different graph-based molecular encoders.



(a) The 10th task. (b) The 11th task. (c) The 12th task.

Fig. 6. Comparison between $\mathbf{G}_{\omega,m}$ computed using ground-truth labels (the first row) and adjacency matrix $\mathbf{A}_{\omega,m}$ returned by PAR (the second row) for the ten molecules. We set $[\mathbf{G}_{\omega,m}]_{pq} = 1$ if molecules $\mathbf{x}_{\omega,m,p}$ and $\mathbf{x}_{\omega,m,q}$ have the same label and 0 otherwise.

in general, note that both molecular encoders learned from scratch or pretrained are useful. Pretrained encoders contain rich generic molecular information by learning enormous unlabeled data, while encoders learned from scratch can carry some new insights. For example, the recent DimeNet [67] can model directional information such as bond angles and rotations between atoms, which has no pretrained version. As our proposed method can use any molecular encoder to obtain generic molecular embedding, it can easily accommodate newly proposed molecular encoder w/o or w/ pretraining.

5) Case Study: Finally, we validate whether PAR can obtain different property-aware molecular embeddings and correct relation graphs. To examine this under a controlled setting, we sample a fixed group of 10 molecules on Tox21 (see Appendix D.5) which coexist in different meta-testing tasks (i.e., the 10th, 11th and 12th tasks). Provided with the meta-learned parameters $\boldsymbol{\theta}^*$ and $\boldsymbol{\Phi}^*$, we take these 10 molecules as the support set to fine-tune $\boldsymbol{\Phi}^*$ as $\boldsymbol{\Phi}^*_\omega$ and keep $\boldsymbol{\theta}^*$ fixed in each task $\mathcal{T}_\omega$. As the support set is fixed now, the ratio of active molecules to inactive molecules among the 10 molecules may not be 1:1 in the three tasks. Thus, the resultant task may not evenly contain K labeled samples per class.

Learned Relation Graphs: We first visualize the learned relation graph in PAR. As described in Section III-C, PAR returns $\mathbf{B}_{\omega,m}$ as the adjacency matrix encoding the optimized relation graph among molecules. Each element $[\mathbf{B}_{\omega,m}]_{pq}$ records the pairwise similarity of the 10 molecules and a random query (which is dropped then). As the number of active and inactive molecules may not be equal in the support set, we no longer reduce adjacency matrices $\mathbf{A}_{\omega,m}$ to $\mathbf{B}_{\omega,m}$ which encodes KNN graph. Fig. 6 plots the optimized adjacency matrices obtained on all three tasks. As can be observed, PAR obtains different adjacency matrices for different property-prediction tasks. Besides, the learned adjacency matrices are visually similar to the ones computed using ground-truth labels.

Molecular Embeddings: We also present the t-SNE [68] visualization of $\mathbf{g}_{\omega,i}$ (molecular embedding obtained by graph-based molecular encoders), $\mathbf{p}_{\omega,i}$ (molecular embedding obtained by property-aware molecular encoder), and $\mathbf{h}_{\omega,i}$ (molecular embedding returned by PAR) for these 10 molecules. For the same $\mathbf{x}_{\omega,i}$, $\mathbf{g}_{\omega,i}$ is the same across the 10th, 11th, 12th tasks, while $\mathbf{p}_{\omega,i}$ and $\mathbf{h}_{\omega,i}$ are property-aware, $\mathbf{h}_{\omega,i}$ which collectively contains molecular embeddings adjusted on relation graphs. Fig. 7 shows the results. As shown, PAR indeed captures property-aware information during encoding the same molecules for different MPP tasks. From the first row to the third row in Fig. 7, molecular embeddings gradually get closer to the class prototypes on all three tasks. The visualization of $\mathbf{h}_{\omega,i}$ shows that the two classes are properly clustered, which shows the importance of relation graph learning (metric-based method).

B. Transferable Few-Shot Setting

1) Setup: Datasets: As suggested by IterRefLSTM [3] and DTCR [2], we consider transfer learning (i) between Tox21 and SIDER which contain distinct tasks; (ii) from ToxCast to Tox21 which both evaluate toxicity and (iii) from ToxCast to SIDER which differ largely, based on datasets in Table I.

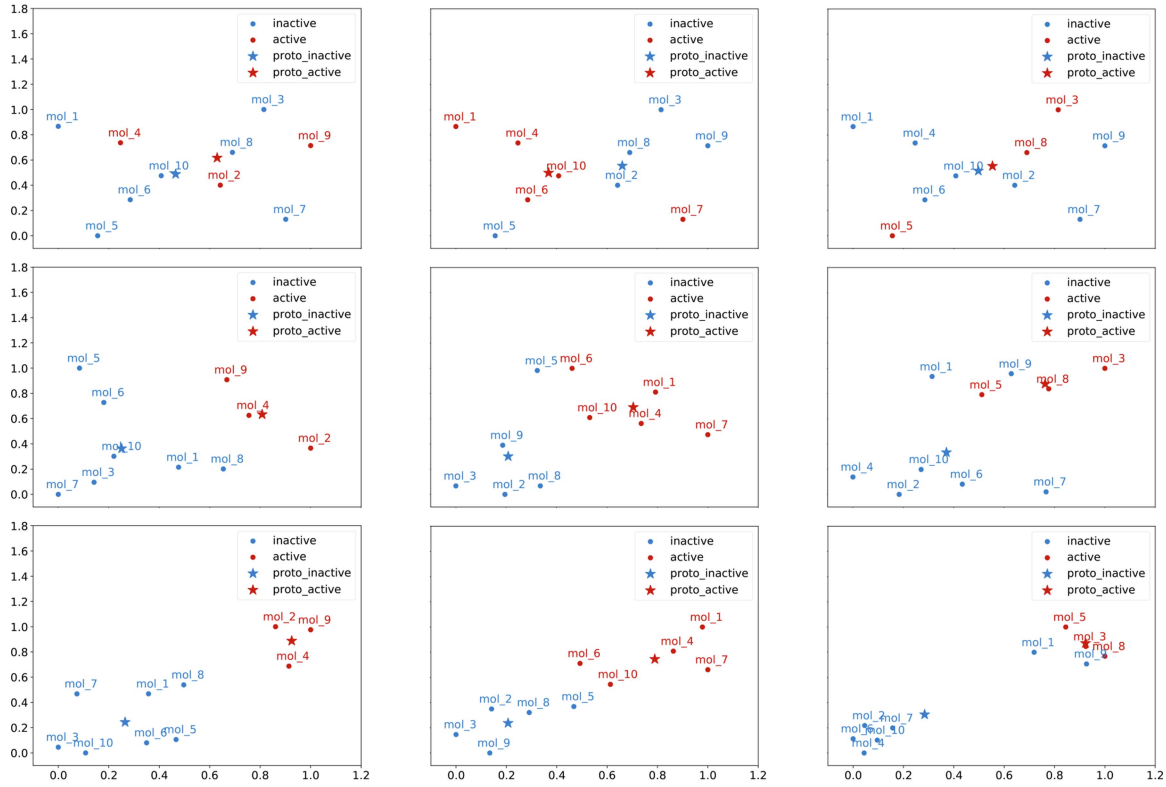
Evaluation Metrics: We compute ROC-AUC scores on query set of all tasks in the target datasets as in Section V-A1.

Baselines: On transferable FSL setting, the methods we used can be categorized into two types. Firstly, we consider the methods that perform well in the FSL setting in Section V-A2. Here, we change the test set to target domain data to test their performance (target domain data is not available in the training step). They include

- Meta − MGNN [8] which is the few-shot MPP method that performs second best in ordinary few-shot setting;
- PAR which is our few-shot MPP method introduced in Section III that does not consider distribution shift;

Secondly, we consider the transferable FSL methods that fit for the transferable few-shot MPP problems (only a few labeled samples in target domain are available in the training step). They include

- BOIL [62] which is a variant of MAML that can be applied to transferable FSL problems;
- DAPN [40] which is selected as representative transferable FSL baseline due to its superior performance, as introduced in Section II-B2;
- and the proposed T − PAR (Algorithm 2).



(a) The 10th task. (b) The 11th task. (c) The 12th task.

Fig. 7. t-SNE visualization of $\mathbf{g}_{\omega,i}$ (the first row), $\mathbf{p}_{\omega,i}$ (the second row), and $\mathbf{h}_{\omega,i}$ (the third row) of the ten molecules, where $\mathbf{h}_{\omega,i}$ collectively contains molecular embeddings obtained after relation graph learning (metric-based method). Proto_active (proto_inactive) denotes the class prototype of active (inactive) class.

TABLE III
ROC-AUC SCORES IN TRANSFERABLE FEW-SHOT MOLECULAR PROPERTY PREDICTION PROBLEMS

Method	Tox21 → SIDER		SIDER → Tox21		ToxCast → SIDER		ToxCast → Tox21	
	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot
Meta-MGNN	$54.62_{(0.20)}$	$54.48_{(0.70)}$	$63.52_{(2.64)}$	$62.61_{(1.33)}$	$54.75_{(0.36)}$	$54.61_{(1.33)}$	$72.86_{(0.40)}$	$71.40_{(0.63)}$
PAR	$55.18_{(0.47)}$	$55.31_{(0.56)}$	$66.56_{(0.46)}$	$65.13_{(1.89)}$	$56.28_{(0.84)}$	$56.00_{(0.62)}$	$73.98_{(0.11)}$	$73.40_{(0.03)}$
BOIL	$57.28_{(0.56)}$	$57.07_{(0.85)}$	$57.89_{(0.78)}$	$57.68_{(1.07)}$	$57.28_{(0.77)}$	$56.89_{(1.04)}$	$69.74_{(1.21)}$	$69.30_{(0.87)}$
DAPN	$55.02_{(0.06)}$	$53.53_{(0.34)}$	$73.19_{(0.86)}$	$64.02_{(0.95)}$	$55.36_{(0.55)}$	$53.78_{(0.20)}$	$73.21_{(1.09)}$	$64.18_{(1.43)}$
T-PAR	$64.57_{(0.66)}$	$60.23_{(0.70)}$	$77.61_{(0.37)}$	$74.36_{(1.06)}$	$63.30_{(1.04)}$	$61.53_{(0.47)}$	$76.65_{(0.99)}$	$75.09_{(0.84)}$

The best results (the pairwise t-test with 95% confidence) are highlighted in gray. Methods in upper two rows are trained without any target domain data, while methods in lower three rows are trained with a few target domain data available. All the methods use the pretrained GIN model as molecular encoder uniformly.

For approaches mentioned-above apart from PAR and T-PAR, we use public codes from the respective authors to get the results. We uniformly use the pretrained graph based molecular encoder for fair comparison. DTCR [2] is not compared because it is trained on the whole base training set of source domain and does not follow the setting of FSL which samples meta-training tasks for the training step.

2) Performance Comparison: Table III presents the results where T-PAR constantly outperforms the others. In terms of average performance improvement, T-PAR outperforms the best baseline by 9.05%. PAR and Pre-GNN cannot predict the property of molecules in target domain well in transferable FSL setting, as they neither get access to target domain data during training nor consider the distribution shift between source and target domain. BOIL and DAPN also fail to get good results, as they neglect the specialty of MPP problems. We also observe that all methods obtain higher ROC-AUC score on ToxCast → Tox21 than ToxCast → SIDER, which shows transfer learning from similar source dataset is more helpful.

We also conduct experiment on the number of target domain molecules given for training in each task. The results on Tox21 → SIDER are in Fig. 8 and we put results on SIDER → Tox21 in Appendix D.7. The results show that T-PAR significantly improves performance given target domain molecules in

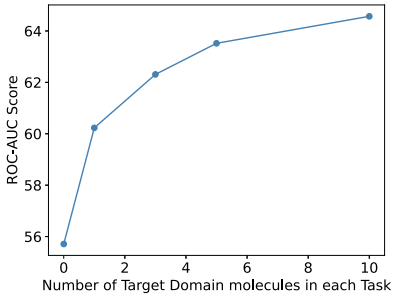


Fig. 8. Performance of T-PAR given different numbers of target domain samples for tasks from Tox21 → SIDER.

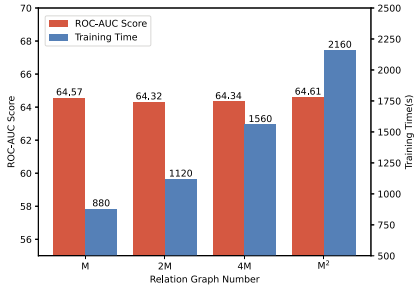


Fig. 10. The experiment on the number of joint relation graphs in each task pair for 10-shot tasks from Tox21 → SIDER.

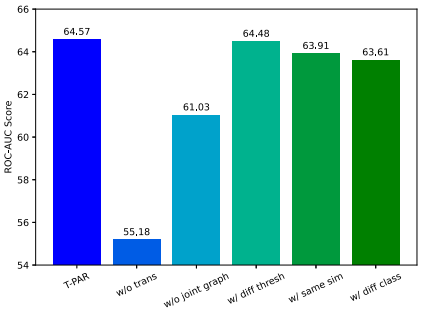


Fig. 9. Varying components in T-PAR for 10-shot tasks from Tox21 → SIDER.

the training procedure and performs better with more training target domain molecules, which demonstrate the importance of target domain data for T-PAR.

3) Varying Components in T-PAR: Here, we use the control variates method to explore the influence of these varying components in T-PAR. We compare T-PAR with the following variants:

- w/otrans which removes using joint sampling strategy and joint relation graph learning;
- w/ojointgraph which removes constructing joint relation graph among molecules from different domains;
- w/diffthresh which uses $\mathcal{N}([\hat{\mathbf{A}}^{(\tau)}_{tt,m}]_{:q}, K)$ as the threshold for the sparsification of $\hat{\mathbf{A}}^{(\tau)}_{st,m}$ and $\hat{\mathbf{A}}^{(\tau)}_{ts,m}$ instead of $\mathcal{N}([\hat{\mathbf{A}}^{(\tau)}_{ss,m}]_{:q}, K)$ for constructing (13);
- w/samesim which uses the same $\mathbf{W}_a$ to estimate (12);
- w/diffclass which uses different $\mathbf{W}_c$ to classify molecules from different domains.

Results for 10-shot tasks are shown in Fig. 9. We can see that the results of w/o trans and w/o joint graph are lower than our T-PAR, which show the positive effect of joint sampling and relation graph learning. For the performances of other variants, w/ same sim gets worse results as it cannot separately capture the knowledge of different kinds of relations, while w/ diff class is harmful to the prediction results because the classifier in the target domain cannot obtain useful information from source domain molecules when using separate classifiers.

4) Number of Joint Relation Graphs in Each Task Pair: In Section IV-D, we construct M^2 joint relation graphs in each task pair during training. In this part, we conduct experiments on the number of joint relation graphs in each task pair. Note that we have to construct at least M joint relation graphs by sampling each query molecule in the two tasks without replacement to obtain the property prediction for all the query molecules once.

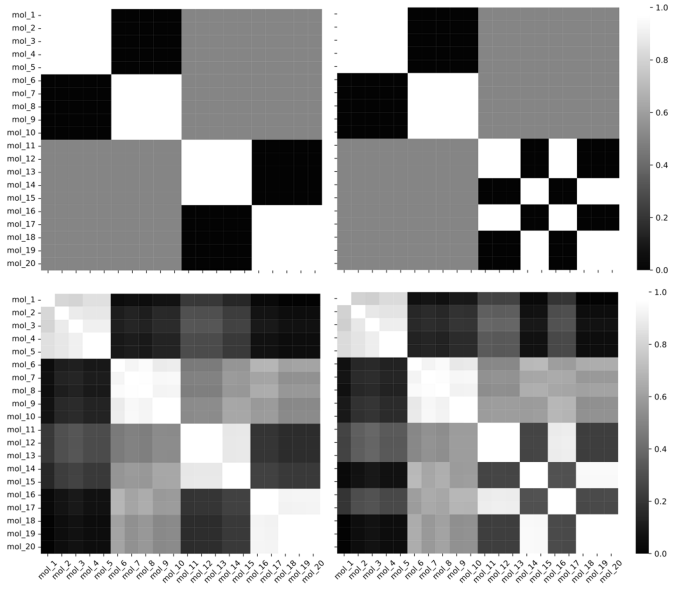
Fig. 10 shows the results on 10-shot tasks. We can see that constructing $2M$, $4M$ or M^2 joint relation graphs do not get better performance than constructing M joint relation graphs though taking more training time. Therefore, we construct only M joint relation graphs in each task pair in experiments.

5) Case Study: In the case study, we visualize the joint relation graph in T-PAR and the molecular embeddings in PAR and T-PAR under the transferable FSL setting.

Learned Joint Relation Graphs: To display different joint relation graphs under a controlled setting, we sample a fixed group of 10 molecules on Tox21 as source domain task (mol_1 - mol_10) and another fixed group 10 molecules on SIDER that coexist in two target domain tasks (mol_11 - mol_20). The 20 molecules form two task pairs (i.e., the 1st and 2nd task pairs) and their specific information is in Appendix D.8. In order to visualize the complete joint relation graph, here we show the joint adjacency matrix $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ and do not conduct the sparsification step to update it as $\hat{\mathbf{B}}^{(\tau)}_{\omega,m}$.

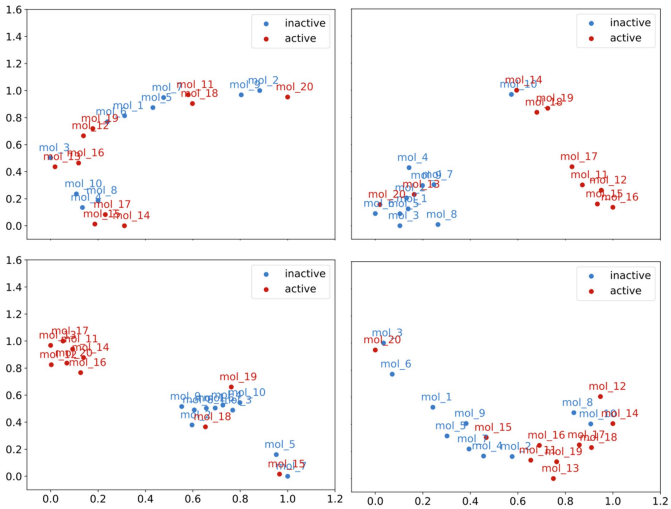
Results of joint relation graph visualization are in Fig. 11. Denote the upper left and lower right part of the matrices as inner-task parts, which represent the relation of molecules in the same task. Denote the upper left and lower right part of the matrices as inter-task parts, which represent the relation between molecules in different tasks. As can be observed, the inner-task parts in the matrices show the truth relation of the molecules, which means that T-PAR can precisely predict inner-task relations. The inter-task parts have darker color than the inner-task parts, which means that the molecules in the same task have closer relation. Moreover, inner-task parts of the two matrices is different, demonstrating that the relation between different task pairs is different.

Molecular Embeddings: For the molecular embeddings, we respectively sample a 10-shot task from source domain and target domain, and use t-SNE to visualize the molecular embedding $\mathbf{h}_{\omega,i}$ produced by PAR and T-PAR. As shown in the results in Fig. 12, T-PAR can distinguish molecules with different labels in target domain better. But PAR differentiates differently labeled molecules from source domain better, because T-PAR further considers molecules from target domain in the training process, which possibly influence the embedding learning for source domain adversely.



(a) The 1st task pair. (b) The 2nd task pair.

Fig. 11. The upper left and lower right part of the matrices in the first row show the ground-truth relation graph among molecules in the same task. The matrices in the second row show the illustration of the joint relation graph $\hat{\mathbf{A}}^{(\tau)}_{\omega,m}$ returned by T-PAR for the 20 molecules.



(a) The target domain task. (b) The source domain task.

Fig. 12. t-SNE visualization of $\mathbf{h}_{\omega,i}$ produced by PAR (the first row) and T-PAR (the second row) of source and target domain tasks from Tox21 $\rightarrow$ SIDER.

VI. Conclusion

We propose Property-Aware Relation networks (PAR) to address the few-shot MPP problem. PAR contains a graph-based molecular encoder, a property-aware molecular encoder, and a relation graph learning module. By design, PAR can obtain property-aware molecular embeddings and model molecular relation graph adaptively. We extend PAR as T-PAR to handle the transferable few-shot MPP problem. Extensive empirical results show that PAR and T-PAR obtain state-of-the-art results on typical few-shot and transferable few-shot property prediction problems, respectively.

For future work, we plan to explore hypernetwork-based methods [69] which have the potential to offer faster inference speeds and reduce the risk of overfitting, in contrast to optimization-based meta-learning methods that rely on gradient descent. Additionally, we are considering the use of automated machine learning techniques [70], [71] to search for property-specific architectures automatically. This approach could potentially free researchers and engineers from the task of manually designing models.

References

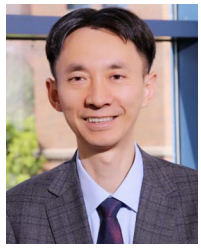
[1] S. G. Rohrer and K. Baumann, "Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data," *J. Chem. Inf. Model.*, vol. 49, no. 2, pp. 169–184, 2009.
[2] K. Abbasi, A. Poso, J. Ghasemi, M. Amanlou, and A. Masoudi-Nejad, "Deep transferable compound representation across domains and tasks for low data drug discovery," *J. Chem. Inf. Model.*, vol. 59, no. 11, pp. 4528–4539, 2019.
[3] H. Altae-Tran, B. Ramsundar, A. S. Pappu, and V. Pande, "Low data drug discovery with one-shot learning," *ACS Central Sci.*, vol. 3, no. 4, pp. 283–293, 2017.
[4] S. M. Paul et al., "How to improve R&D productivity: The pharmaceutical industry's grand challenge," *Nature Rev. Drug Discov.*, vol. 9, no. 3, pp. 203–214, 2010.
[5] S. P. Leelananda and S. Lindert, "Computational methods in drug discovery," *Beilstein J. Org. Chem.*, vol. 12, no. 1, pp. 2694–2718, 2016.
[6] A. Zhavoronkov et al., "Deep learning enables rapid identification of potent DDR1 kinase inhibitors," *Nature Biotechnol.*, vol. 37, no. 9, pp. 1038–1040, 2019.
[7] G. E. Dahl, N. Jaitly, and R. Salakhutdinov, "Multi-task neural networks for QSAR predictions," 2014, *arXiv:1406.1231*.
[8] Z. Guo et al., "Few-shot graph learning for molecular property prediction," in *Proc. Web Conf.*, 2021, pp. 2559–2567.
[9] Y. Rong et al., "Self-supervised graph transformer on large-scale molecular data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 12559–12571.
[10] C. Q. Nguyen, C. Kreatsoulas, and K. M. Branson, "Meta-learning GNN initializations for low-resource molecular property prediction," 2020, *arXiv: 2003.05996*.
[11] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.
[12] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.
[13] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4080–4090.
[14] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1199–1208.
[15] P. Rodríguez, I. Laradji, A. Drouin, and A. Lacoste, "Embedding propagation: Smoother manifold for few-shot classification," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 121–138.
[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
[17] A. A. Rusu et al., "Meta-learning with latent embedding optimization," in *Proc. Int. Conf. Learn. Representations*, 2019.
[18] K. Lee, S. Maji, A. Ravichandran, and S. Soatto, "Meta-learning with differentiable convex optimization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10657–10665.
[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2016.
[20] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.
[21] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *Proc. Int. Conf. Learn. Representations*, 2018.

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[23] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3637–3645.
[24] W. Hu et al., "Strategies for pre-training graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2019.
[25] National Center for Advancing Translational Sciences, "Tox21 challenge," 2017, Accessed: Nov. 06, 2016. [Online]. Available: http://tripod.nih.gov/tox21/challenge/
[26] C. Cai et al., "Transfer learning for drug discovery," *J. Med. Chem.*, vol. 63, no. 16, pp. 8683–8694, 2020.
[27] Y. Ji et al., "DrugOOD: Out-of-distribution dataset curator and benchmark for AI-aided drug discovery–a focus on affinity prediction problems with noise annotations," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 8023–8031.
[28] Y. Wang, A. Abuduweili, Q. Yao, and D. Dou, "Property-aware relation networks for few-shot molecular property prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 17441–17454.
[29] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[30] W. Hu et al., "Open graph benchmark: Datasets for machine learning on graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22118–22133.
[31] Y. Chen, L. Wu, and M. Zaki, "Iterative deep graph learning for graph neural networks: Better and robust node embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 19314–19326.
[32] Y. Zhu, W. Xu, J. Zhang, Q. Liu, S. Wu, and L. Wang, "Deep graph structure learning for robust representations: A survey," 2021, *arXiv:2103.03036*.
[33] Y. Wang, S. Wang, Q. Yao, and D. Dou, "Hierarchical heterogeneous graph representation learning for short text classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 3091–3101.
[34] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, and R. Turner, "Meta-learning probabilistic inference for prediction," in *Proc. Int. Conf. Learn. Representations*, 2019.
[35] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. Int. Conf. Learn. Representations*, 2017.
[36] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals, "Rapid learning or feature reuse? Towards understanding the effectiveness of MAML," in*Proc. Int. Conf. Learn. Representations*, 2019.
[37] H. Ye and W. Chao, "How to train your MAML to excel in few-shot classification," in *Proc. Int. Conf. Learn. Representations*, 2016.
[38] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales, "Learning to generalize: Meta-learning for domain generalization," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3490–3497.
[39] H.-Y. Tseng, H.-Y. Lee, J.-B. Huang, and M.-H. Yang, "Cross-domain few-shot classification via learned feature-wise transformation," 2020, *arXiv: 2001.08735*.
[40] L. Z. ZhaoA and M. Ding, "Domain-adaptive few-shot learning," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 1390–1399.
[41] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7167–7176.
[42] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1647–1657.
[43] A. R. Katritzky, V. S. Lobanov, and M. Karelson, "QSPR: The correlation and quantitative prediction of chemical and physical properties from structure," *Chem. Soc. Rev.*, vol. 24, no. 4, pp. 279–287, 1995.
[44] O. Wieder et al., "A compact review of molecular property prediction with graph neural networks," *Drug Discov. Today: Technol.*, vol. 37, pp. 1–12, 2020.
[45] J. S. Delaney, "ESOL: Estimating aqueous solubility directly from molecular structure," *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 3, pp. 1000–1005, 2004.
[46] C. Merkwirth and T. Lengauer, "Automatic generation of complementary descriptors with molecular graph networks," *J. Chem. Inf. Model.*, vol. 45, no. 5, pp. 1159–1168, 2005.
[47] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *J. Chem. Inf. Model.*, vol. 50, no. 5, pp. 742–754, 2010.
[48] D. L. Mobley and J. P. Guthrie, "FreeSolv: A database of experimental and calculated hydration free energies, with input files," *J. Comput.-Aided Mol. Des.*, vol. 28, no. 7, pp. 711–720, 2014.
[49] Z. Wu et al., "MoleculeNet: A benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, no. 2, pp. 513–530, 2018.
[50] A. Lusci, G. Pollastri, and P. Baldi, "Deep architectures and deep learning in chemoinformatics: The prediction of aqueous solubility for drug-like molecules," *J. Chem. Inf. Model.*, vol. 53, no. 7, pp. 1563–1575, 2013.
[51] D. Duvenaud et al., "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
[52] Z. Xiong et al., "Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism," *J. Med. Chem.*, vol. 63, no. 16, pp. 8749–8760, 2019.
[53] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[54] G. B. Goh, C. Siegel, A. Vishnu, and N. Hodas, "Using rule-based labels for weak supervised learning: A ChemNet for transferable chemical property prediction," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 302–310.
[55] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 201–208.
[56] S. Wang, Y. Guo, Y. Wang, H. Sun, and J. Huang, "SMILES-BERT: Large scale unsupervised pre-training for molecular property prediction," in *Proc. Int. Conf. Bioinf. Comput. Biol. Health Inform.*, 2019, pp. 429–436.
[57] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C.-K. Lee, "Motif-based graph self-supervised learning for molecular property prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 15870–15882.
[58] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.
[59] M. Kuhn, I. Letunic, L. J. Jensen, and P. Bork, "The SIDER database of drugs and side effects," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D1075–D1079, 2016.
[60] A. M. Richard et al., "ToxCast chemical LandScape: Paving the road to 21st century toxicology," *Chem. Res. Toxicol.*, vol. 29, no. 8, pp. 1225–1251, 2016.
[61] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2008.
[62] J. Oh, H. Yoo, C. Kim, and S.-Y. Yun, "BOIL: Towards representation change for few-shot learning," in *Proc. Int. Conf. Learn. Representations*, 2021.
[63] G. Landrum, "RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling," 2013.
[64] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proc. ICML Deep Learn. Workshop*, Lille, 2015.
[65] Y. Liu et al., "Learning to propagate labels: Transductive propagation network for few-shot learning," in *Proc. Int. Conf. Learn. Representations*, 2018.
[66] J. Kim, T. Kim, S. Kim, and C. D. Yoo, "Edge-labeling graph neural network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11–20.
[67] J. Klicpera, J. Groß, and S. Günnemann, "Directional message passing for molecular graphs," in *Proc. Int. Conf. Learn. Representations*, 2019.
[68] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.
[69] P. Bateni, R. Goyal, V. Masrani, F. Wood, and L. Sigal, "Improved few-shot visual classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14481–14490.
[70] Q. Yao et al., "Taking human out of learning applications: A survey on automated machine learning,", 2018, *arXiv: 1810.13306*.
[71] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, pp. 55:1–55:21, 2019.

Quanming Yao (Senior Member, IEEE) received the PhD degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST). He is a tenure-track assistant professor with the Department of Electronic Engineering, Tsinghua University. He regularly serves as area chairs for ICML, NeurIPS and ICLR. He is also a receipt of National Youth Talent Plan (China), Forbes 30 Undyer 30 (China), Young Scientist Awards (Hong Kong Institution of Science), and Google Fellowship (in machine learning).

Zhenqian Shen received the BE degree in electronic engineering from Tsinghua University, in 2022. He is currently working toward the PhD degree with the Department of Electronic Engineering, Tsinghua University. His research interest includes is few-shot learning.

Yaqing Wang (Member, IEEE) received the PhD degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, 2019. She is a staff researcher with Business Intelligence Lab, Baidu Research, Baidu Inc. Her research interests include machine learning, especially few-shot learning and its applications on biomedical applications, natural language processing, and knowledge graphs. She serves as senior program committee members at IJCAI and AAAI, and reviewers at ICML, NeurIPS, and ICLR.



Dejing Dou (Senior Member, IEEE) received the bachelor's degree from Tsinghua University, in 1996, and the PhD degree from Yale University, in 2004. He is the head of Big Data Lab (BDL) and Business Intelligence Lab (BIL) with Baidu Research. He is also a full professor (on leave) from the Computer and Information Science Department, University of Oregon and has led the Advanced Integration and Mining (AIM) Lab since 2005. His research interests include artificial intelligence, data mining, data integration, NLP, and health informatics.