Property-aware Adaptive Relation Networks for Molecular Property Prediction

Anonymous Author(s)
Affiliation
Address
email

Abstract

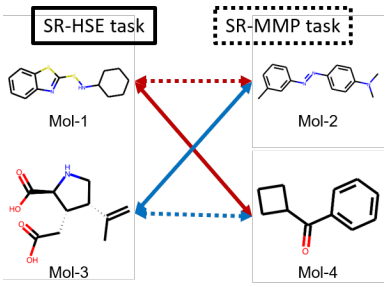
Molecular property prediction plays a fundamental role in drug discovery to discover candidate molecules with target properties. However, molecular property prediction is essentially a few-shot problem which makes it hard to obtain regular models. In this paper, we propose property-aware adaptive relation networks for the few-shot molecular property prediction problem. We first obtain molecular embeddings via a graph-based molecular encoder learned across different tasks where each of them corresponds to the prediction of one property. To capture the specific structure property/activity relationship, we particularly design property-aware embedding projection to co-adapt the embedding of each molecule w.r.t. other molecules in the current task and further select relevant substructures. The resultant embeddings are used to construct a relation graph modeling the similarity between molecules which facilitates effective information propagation and is dynamically optimized during learning. Extensive experiments on many benchmark drug property prediction datasets show that our method outperforms state-of-the-art methods under both standard few-shot learning settings and transfer learning across different datasets setting.

1 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 [34, 1, 4]. 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% [31, 26, 47]. To speedup this process, quantitative structure property/activity relationship (QSPR/QSAR) modeling uses machine learning methods to establish the connection between molecule structure and particular properties [6]. It usually consists of two components: a molecular encoder which encodes molecule structure as a fixed-length molecular representation, and a predictor which estimates the activity of a certain property based on the molecular representation. Predictive models can be leveraged in virtual screening to discover potential molecules more efficiently [16]. However, molecular property prediction 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 [35]. After a series of wet-lab experiments, most candidates still fail to be a potential drug due to the lack of any desired properties [6]. These together result in a limited number of labeled data [29].

Recently, there emerge few-shot learning (FSL) methods [4, 16, 41] for molecular property prediction. They target at learning a predictor from a set of property prediction tasks and generalize to predict new properties with a few labeled molecules. As molecules can be naturally represented as graphs, graph-based molecular representation learning methods use graph neural networks (GNNs) [22, 17] to obtain

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.



Molecules		Label (SR-)	
ID	SMILES	HSE	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

Figure 1: Illustrative examples of relation graphs for the same molecules in two tasks of Tox21.Red (blue) edges mean the connected molecules are both active (inactive) on the target property.

graph-level representation as the molecular embedding. Specifically, the pioneering IterRefLSTM [4] adopts GNN as the molecular encoder and modifies a classic FSL method [40] proposed for image classification to handle molecular prediction tasks. The recent Meta-MGNN [16] leverages a GNN pretrained from large-scale self-supervised tasks as molecular encoder. It introduces additional self-supervised learning tasks such as bond reconstruction and atom type prediction to be jointly optimized with the molecular property prediction tasks. Finally, DTCR [1] is particularly designed for few-shot transfer learning across datasets by adversarial learning.

However, aforementioned methods neglect two key facts in molecular property prediction. The first fact is that different molecular properties are attributed to different molecule substructures as found by previous QSPR studies [2]. While IterRefLSTM and Meta-MGNN use graph-based molecular encoder to encode molecules regardless of target properties whose relevant substructures can be dramatically different. The second fact is that the relationship among molecules also vary w.r.t. the target property. This can be commonly observed in benchmark molecular property prediction datasets, as shown in Figure 1. However, existing works fail to model relation graph among molecules.

To handle these problems, we propose a property-aware adaptive relation networks (PAR) which is compatible with existing graph-based molecular encoders, and are further equipped with the ability to obtain property-aware molecular embedding and model molecular relation graph adaptively. Specifically, our contribution can be summarized as follows:

- We propose a property-aware embedding function which co-adapts each molecule embedding with class prototypes and further projects it to a substructure-aware space w.r.t. the target property.
- We propose an adaptive relation graph learning module to jointly estimate molecular relation graph and refine molecular embeddings w.r.t. the target property, such that the limited labels can be efficiently propagated between similar molecules.
- We propose a new training strategy: we only fine-tune the property-aware embedding function and final classifier while keeping the other parts (graph-based molecular encoder and adaptive relation graph learning module) fixed within each task. We show it is particularly helpful to separately capture the generic knowledge shared across different tasks and those property-aware.
- We conduct extensive empirical studies on real molecular property prediction datasets. Results consistently show PAR consistently outperform the others. Further model analysis shows PAR can obtain property-aware molecular embedding and model molecular relation graph properly.

Notation. In the sequel, we denote vectors by lowercase boldface, matrices by uppercase boldface, and sets by uppercase calligraphic font. For a vector $\mathbf{x}$, $[x]_i$ denotes the ith element of $\mathbf{x}$. For a matrix $\mathbf{X}$, x^i denotes its ith row, $[X]_{ij}$ denotes the (i,j)th entry of $\mathbf{X}$. The superscript $(\cdot)^\top$ denotes the transpose operation.

2 Background and Related Works: Graph Neural Networks (GNNs)

A graph neural network (GNN) can learn expressive node/graph representation from the topological structure and associated features of a graph via neighborhood aggregation [22, 14, 18]. 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 $\mathbf{b}_{vu}^{(0)}$ for each edge

$e_{vu} \in \mathcal{E}$ between nodes v, u. At the l-th layer, GNN updates the node embedding $\mathbf{h}_v^{(l)}$ of node v as:

$$\mathbf{h}_v^{(l)} = \texttt{UPDATE}^{(l)}(\mathbf{h}_v^{(l-1)}, \texttt{AGGREGATE}^{(l)}(\{(\mathbf{h}_v^{(l-1)}, \mathbf{h}_u^{(l-1)}, \mathbf{b}_{vu}) | u \in \mathcal{N}(v)\})), \tag{1}$$

where $\mathcal{N}(v)$ is a set of neighbors of v. Existing GNNs differ on the design of aggregate function $\texttt{AGGREGATE}^{(l)}(\cdot)$ and update function $\texttt{UPDATE}^{(l)}(\cdot)$. After L iterations of aggregation, the graph-level representation $\mathbf{h}_{\mathcal{G}}$ is obtained as $\mathbf{h}_{\mathcal{G}} = \texttt{READOUT}(\{(\mathbf{h}_v)^{(L)} | v \in \mathcal{V}\})$, where $\texttt{READOUT}(\cdot)$ is a function to aggregate all node embeddings into the graph embedding, such as summation [43].

Our paper is related to GNN in two aspects: we obtain molecular representation via graph-based molecular encoder, construct and leverage relation graph by learning graph structures.

Graph-based Molecular Representation Learning. Representing molecules properly as fixed-length vectors is vital to the success of downstream biomedical applications [13]. Recently, graph-based molecular representation learning methods are popularly used and obtain state-of-the-art performance. A molecule x_i is represented as an undirected graph $\mathcal{G}_{x_i} = \{\mathcal{V}_{x_i}, \mathcal{E}_{x_i}\}$, where each node $v \in \mathcal{V}_{x_i}$ represents an atom with feature $\mathbf{h}_v^{(0)} \in \mathbb{R}^{d^n}$ and each edge $e_{vu} \in \mathcal{E}_{x_i}$ represents the bond between two nodes v, u with feature $\mathbf{b}_{vu} \in \mathbb{R}^{d^e}$. Graph-based molecular representation learning methods use graph neural networks (GNNs) to obtain graph-level representation $\mathbf{h}_{\mathcal{G}_i}$ as molecular embedding. Examples include graph convolutional networks (GCN) [7], graph attention networks (GAT) [39], message passing neural networks (MPNN) [14], Directed MPNN [44], AttentiveFP [42], Pretrained GNN [19] and GROVER [35].

In FSL methods for molecular property prediction, IterRefLSTM [4] uses a GCN while Meta-MGNN [16] uses the pretrained Pre-GNN. Using these graph-based molecular encoders cannot discover molecule substructures corresponding to the target property. Although there exist GNNs which handle subgraphs [28, 3, 11], they require predefined subgraphs. While discovering and enumerating molecule substructures is extremely hard even for domain experts [2, 46]. In this paper, we first obtain molecular embeddings using graph-based molecular encoders. We further learn to extract relevant substructure embeddings w.r.t. the target property upon these generic molecular embeddings, which is more effective and improves the performance.

Graph Structure Learning. As the provided graphs may not be optimal, a number of graph structure learning methods target at jointly learning graph structure and node embeddings [48**? ?**]. In general, they iterate over two procedures: estimating adjacency matrix (i.e., refining neighborhood $u \in \mathcal{N}(v)$) which encodes graph structure from the current node embeddings, and apply a GNN on this learned graph to obtain new node embeddings.

In FSL, there exist some works [12, 27, 20, 45, 33, 49] which learn to construct fully-connected relation graph among images in a N-way-K-shot few-shot image classification task. Their methods cannot work for the 2-way-K-shot property prediction tasks where choosing a wrong neighbor in the different class will heavily deteriorate the quality of molecular embeddings. Although we share the same spirit of learning relation graph, we introduce several regularizations to encourage our adaptive property-aware relation graph learning module to select correct neighbors effectively.

3 Proposed Method

In this section, we present the details of PAR, whose overall architecture is shown in Figure 2. Considering few-shot molecular property prediction problem, we first obtain property-aware molecular embeddings via a specially designed embedding function, and then propagate the limited labels on the adaptive molecular relation graph whose structure is jointly optimized with the molecular embeddings. To optimize PAR, we introduce a new training strategy to separately modeling generic and property-aware knowledge. Finally, we PAR can be easily extended to handle the few-shot transfer learning across datasets problem.

3.1 Problem Definition

As defined in [4, 16], a few-shot molecular property prediction task T_τ is formulated as a 2-way-K-shot classification task with a support set $\mathcal{S}_\tau = \{(x_{\tau,i}, y_{\tau,i})\}_{i=1}^{2K}$ and a query set $\mathcal{Q}_\tau = \{(x_{\tau,j}, y_{\tau,j})\}_{j=1}^{N_\tau^q}$ where K labeled samples are provided per class and K is small. Each

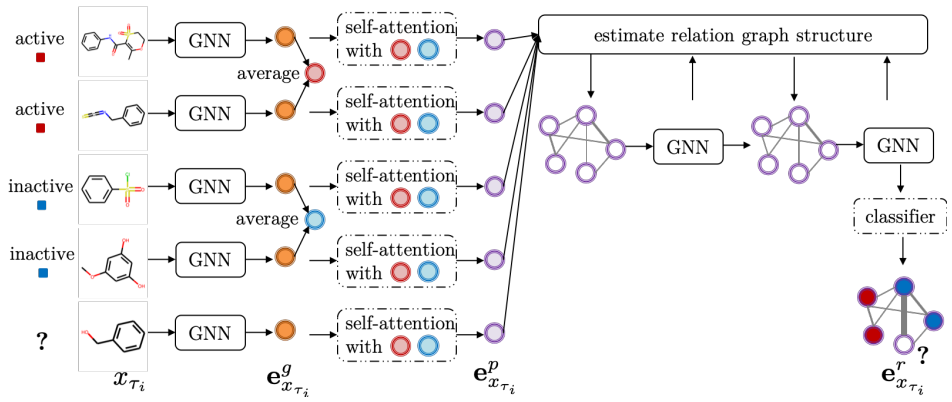


Figure 2: The architecture of the proposed PAR framework. We illustrate using a 2-way-2-shot task from Tox21. PAR is optimized over different molecular property prediction tasks. Within each task T_τ, the modules with dotted lines are fine-tuned on support set $\mathcal{S}_\tau$ and those with solid lines are fixed.

T_τ corresponds to an experimental assay testing on whether each molecule $x_{\tau,i}$ is active ($y_{\tau,i} = 1$) or inactive ($y_{\tau,i} = 0$) on a target property.

3.2 Property-aware Molecular Embedding

Our molecular encoder consists of (i) a graph-based molecular encoder which is trained from large-scale tasks to capture generic information, and (ii) a property-aware embedding function which adapts the generic molecular embeddings to be property-aware.

Recent advances in graph-based molecular encoder such as pretrained molecular encoder [19, 35] makes it possible to encode generic knowledge into molecular embedding by learning across tasks. Thus, we first obtain a generic molecular embedding $\mathbf{e}^g_{x_{\tau,i}} \in \mathbb{R}^{d^g}$ of $x_{\tau,i}$ using an existing graph-based molecular encoder introduced in Section 2, such as AttentiveFP [42] and Pre-GNN [19]. The parameter $\boldsymbol{\theta}^g$ of this graph-based molecular encoder is optimized across large-scale tasks.

However, existing graph-based molecular encoders cannot capture property-aware substructures as discussed above. When learning across tasks, a molecule will be evaluated for multiple properties. This leads to a one-to-many relationship between a molecule and properties. Thus, we are motivated to implicitly capture substructures in the embedding space w.r.t. the target property of T_τ. Let $\mathbf{p}_c$ denotes the class prototype for class $c \in \{0, 1\}$, which is computed as the average of $\mathbf{e}^p_{x_{\tau,i}}$ in $\mathcal{S}_\tau$ whose $y_{\tau,i} = c$. We model the context for $x_{\tau,i}$ as $\mathbf{C}_{\tau,i} = [(\mathbf{e}^p_{x_{\tau,i}})^\top; \mathbf{p}_0^\top; \mathbf{p}_1^\top] \in \mathbb{R}^{3\times d^g}$. We then transform $\mathbf{e}^g_{x_{\tau,i}}$ into $\mathbf{e}^p_{x_{\tau,i}}$ by:

$$\mathbf{e}^p_{x_{\tau,i}} = \texttt{MLP}_{\theta^p}(\texttt{concat}[\mathbf{e}^g_{x_{\tau,i}}, \mathbf{e}^c_{x_{\tau,i}}]) \quad \text{with} \quad \mathbf{e}^c_{x_{\tau,i}} = [\texttt{softmax}(\mathbf{C}_{\tau,i}\mathbf{C}_{\tau,i}^\top/\sqrt{d^g})\mathbf{C}_{\tau,i}]_{1:}, \tag{2}$$

where $[\cdot]_{1:}$ extracts the 1st row vector which corresponds to $x_{\tau,i}$. The $\texttt{MLP}_{\theta^p}$ denotes the multilayer perceptron, which is used to find a lower-dimensional space which encodes substructures that are more relevant to the target property of T_τ. $\mathbf{e}^c_{x_{\tau,i}}$ is computed using scaled dot-product self-attention [38], such that each $\mathbf{e}^g_{x_{\tau,i}}$ can be compared with class prototypes dimensional wise. This contextualized $\mathbf{e}^p_{x_{\tau,i}}$ is property-aware which is more predictive of the target property.

3.3 Adaptive Relation Graph Among Molecules

Apart from relevant substructures, the relationship among molecules also vary across properties. As shown in Figure 1, two molecules with a shared property can be different from each other on another property [34, 24, 32]. In this section, we introduce an adaptive relation graph learning module to capture and further leverage this property-aware relation graph among molecules, such that the limited labels can be efficiently propagated between similar molecules.

Let $\mathbf{A}_{T_\tau} \in \mathbb{R}^{(2K+1)\times(2K+1)}$ denotes the adjacency matrix encoding the the relation graph $\mathcal{G}_{T_\tau}$ among the $2K$ molecules in $\mathcal{S}_\tau$ and a single molecule in $\mathcal{Q}_\tau$. $[\mathbf{A}_{T_\tau}]_{ij} \geq 0$ if nodes $x_{\tau,i}, x_{\tau,j} \in \mathcal{V}_{T_\tau}$ are connected. Ideally, the similarity between property-aware molecular embeddings $\mathbf{e}^p_{x_{\tau,i}}, \mathbf{e}^p_{x_{\tau,j}}$

of $x_{\tau,i}, x_{\tau,j}$ reveals their relationship under the current property prediction task. Hence we set $\mathbf{h}_{\tau,i}^{(0)} = \mathbf{e}^p_{x_{\tau,i}}$ initially. At the lth iteration, we estimate $[\mathbf{A}_{T_\tau}^{(l)}]_{ij}$ between $x_{\tau,i}, x_{\tau,j}$ using the current molecular embeddings:

$$[\mathbf{A}_{T_\tau}^{(l)}]_{ij} = \frac{\exp(-[\mathbf{D}_{T_\tau}^{(l)}]_{ij})}{\sum_{j=1}^{2K+1}\exp(-[\mathbf{D}_{T_\tau}^{(l)}]_{ij})} \text{ with } [\mathbf{D}_{T_\tau}^{(l)}]_{ij} = \texttt{MLP}(|\mathbf{h}_{\tau,i}^{(l-1)} - \mathbf{h}_{\tau,i}^{(l-1)}|). \tag{3}$$

The resultant $\mathbf{A}_{T_\tau}^{(l)}$ is a dense matrix, which encodes a fully connected $\mathcal{G}_{T_\tau}^{(l)}$. However, a new molecule x_{test} only has K real neighbors in $\mathcal{G}_{T_\tau}^{(l)}$ in a 2-way-K-shot task. Choosing a wrong neighbor in the different class will heavily deteriorate the quality of molecular embeddings, especially when only one-shot is provided in each class. To avoid the interference of wrong neighbors, we further sparsify $\mathcal{G}_{T_\tau}^{(l)}$ as a K-nearest neighbor (KNN) graph, where K is set to be exactly the same as the number of labeled molecules per each class in $\mathcal{S}$. The indices of the top K largest $[A_{T_\tau}^{(l)}]_{ij}, j = 1, \ldots, 2K-1$ for $x_{\tau,i}$ is recorded in $\mathcal{N}_K^{(l)}(x_{\tau,i})$. Then, we set $[\hat{\mathbf{A}}_{T_\tau}^{(l)}]_{ij} = [\mathbf{A}_{T_\tau}^{(l)}]_{ij}$ if $x_{\tau,j} \in \mathcal{N}_K^{(l)}(x_{\tau,i})$ and 0 otherwise. With the new $\hat{\mathbf{A}}_{T_\tau}^{(l)}$, we update $\mathbf{h}_{\tau,i}^{(l)}$ by a GNN layer which instantiates (1) as

$$\mathbf{h}_{\tau,i}^{(l)} = \texttt{LeakyReLu}(\mathbf{W}^{(l)} \cdot (\texttt{SUM}(\{\mathbf{h}_{\tau,j}^{(l-1)} : x_{\tau,j} \in \mathcal{N}_K^{(l)}(x_{\tau,i}) \cup x_{\tau,i}\}))). \tag{4}$$

A full adaptive relation graph learning module consists of L GNN layers. We take $\mathbf{e}^r_{x_{\tau,i}} = \mathbf{h}_{\tau,i}^{(L)}$ as the final molecular embedding of $x_{\tau,i}$, and $\hat{\mathbf{A}}_{T_\tau}^{(L)}$ represents the optimized relation graph.

Finally, we obtain class prediction w.r.t. active/inactive as $\hat{\mathbf{y}}_{\tau,i}$ with the jth element calculated as

$$[\hat{\mathbf{y}}_{\tau,i}]_j = \exp(\mathbf{w}_j^\top \mathbf{e}_{x_{\tau,i}}) / \sum\nolimits_{c=1}^{2} \exp(\mathbf{w}_c^\top \mathbf{e}_{x_{\tau,i}}), \tag{5}$$

where $\mathbf{w}_c$ denotes the classifier parameter for class c.

3.4 Training and Inference

For simplicity, we denote PAR as $f(\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ includes all learnable parameters: $\boldsymbol{\theta}^g$ which is the parameter of graph-based molecular encoder, $\boldsymbol{\theta}^p$ which is the parameter of property-aware molecular embedding function, $\boldsymbol{\theta}^r$ which is the parameter of adaptive relation graph learning module, $\boldsymbol{\theta}^c$ which is the parameter of classifier.

In each T_τ, loss $L_{\mathcal{S}_\tau}(f_{\boldsymbol{\theta}})$ evaluated on $\mathcal{S}_\tau$ takes the form:

$$L_{\mathcal{S}_\tau}(f_{\boldsymbol{\theta}}) = \sum\nolimits_{(x_{\tau,i}, y_{\tau,i}) \in \mathcal{S}_\tau} -\mathbf{y}_{\tau,i} \cdot \log(\hat{\mathbf{y}}_{\tau,i}) + \|[\mathbf{A}^*_{T_\tau}]_{i\cdot} - [\hat{\mathbf{A}}_{T_\tau}^{(L)}]_{i\cdot}\|_2^2, \tag{6}$$

where $\mathbf{y}_i \in \mathbb{R}^2$ is a one-hot vector with all 0s but a single one denoting the index of the ground-truth class $c \in \{0, 1\}$, $[\mathbf{X}]_{i\cdot}$ means the ith row of $\mathbf{X}$, and $\mathbf{A}^*_{T_\tau}$ records the ground-truth label consistency where $[\hat{A}]_{ij} = 1$ if $y_i = y_j$ and 0 otherwise. The first term is the cross entropy for classification loss, and the second term is our specially designed neighbor alignment loss which penalizes wrong neighbors in the relation graph.

We adopt gradient-based meta-learning strategy [9] to train PAR: during meta-training, we learn from a set of tasks $\mathcal{T} = \{T_\tau\}_{\tau=1}^{N_t}$ a good initialization $\boldsymbol{\theta}$ that can be easily adapted to $\boldsymbol{\theta}_\tau$ by taking a few gradient descents. Then for each T_τ: we keep $\boldsymbol{\theta}^g, \boldsymbol{\theta}^r$ fixed while fine-tune $\boldsymbol{\theta}^p, \boldsymbol{\theta}^c$ on $\mathcal{S}_\tau$ by taking a few gradient descents. For example, $\boldsymbol{\theta}^p_\tau$ is obtained as

$$\boldsymbol{\theta}^p_\tau = \boldsymbol{\theta}^p - \alpha \nabla_{\boldsymbol{\theta}^p} L_{\mathcal{S}_\tau}(f_{\boldsymbol{\theta}^p}) \tag{7}$$

with learning rate α. By learning this way, we encourage our model to separately capture the generic knowledge shared across different tasks and those property-aware.

Then loss $L_{\mathcal{Q}_\tau}(f_{\{\boldsymbol{\theta}^g, \boldsymbol{\theta}^p_\tau, \boldsymbol{\theta}^r_\tau, \boldsymbol{\theta}^c_\tau\}})$ is calculated in the same form of (6) while using $\mathcal{Q}_\tau$ instead. $\boldsymbol{\theta}^*$ is then obtained as

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum\nolimits_{\tau=1}^{N_t} L_{\mathcal{Q}_\tau}(f_{\{\boldsymbol{\theta}^g, \boldsymbol{\theta}^p_\tau, \boldsymbol{\theta}^r_\tau, \boldsymbol{\theta}^c_\tau\}}), \tag{8}$$

Algorithm 1 Meta-training procedure for PAR.

1: initialize $\boldsymbol{\theta}^g$ randomly or adopt parameter of pretrained models, initialize $\boldsymbol{\theta}^p, \boldsymbol{\theta}^r, \boldsymbol{\theta}^c$ randomly;
2: **while** not done **do**
3: sample a batch of tasks $T_\tau \sim \mathcal{T}$;
4: **for** all T_τ **do**
5: sample a support set $\mathcal{S}_\tau$ and a query set $\mathcal{Q}_\tau$ from T_τ;
6: obtain generic molecular embedding $\mathbf{e}^g_{x_{\tau,i}}$ for each $x_{\tau,i}$ by a graph-based molecular encoder;
7: adapt molecular embedding $\mathbf{e}^g_{x_{\tau,i}}$ to be property-aware $\mathbf{e}^p_{x_{\tau,i}}$ by (2);
8: set $\mathbf{h}^{(0)}_{\tau,i} = \mathbf{e}^p_{x_{\tau,i}}$;
9: **for** $l = 1, \ldots, L$ **do**
10: estimate adjacency matrix $\mathbf{A}^{(l)}_\tau$ of relation graph among molecules using molecular embeddings $\mathbf{h}^{(l-1)}_{\tau,i}$ by (3);
11: refine molecular embeddings $\mathbf{h}^{(l)}_{\tau,i}$ on the updated relation graph $\mathbf{A}^{(l)}_\tau$ by (4);
12: **end for**
13: obtain class prediction $\hat{\mathbf{y}}_{\tau,i}$ using $\mathbf{e}^r_{x_{\tau,i}} = \mathbf{h}^{(L)}_{\tau,i}$;
14: evaluate training loss $L_{\mathcal{S}_\tau}$ on $\mathcal{S}_\tau$;
15: fine-tune $\boldsymbol{\theta}^p, \boldsymbol{\theta}^c$ as $\boldsymbol{\theta}^p_\tau, \boldsymbol{\theta}^c_\tau$ by gradient descents (e.g., by (7));
16: evaluate testing loss $L_{\mathcal{Q}_\tau}$ on $\mathcal{Q}_\tau$;
17: **end for**
18: update $\boldsymbol{\theta} = \{\boldsymbol{\theta}^g, \boldsymbol{\theta}^p, \boldsymbol{\theta}^r, \boldsymbol{\theta}^c\}$ by (8);
19: **end while**

which is also optimized by gradient descent [9]. The complete algorithm of PAR is shown in Algorithm 1. Line 6-7 correspond to property-aware embedding $\mathbf{e}^p_{x_{\tau,i}}$ which encodes substructure w.r.t the target property (see Section 3.2). Line 8-12 corresponds to adaptive relation graph learning which facilitates effective label propagation among similar molecules (see Section 3.3).

For inference, the generalization ability of PAR is evaluated on the query set $\mathcal{Q}_{\text{new}}$ of each new task T_{new} which tests on new property in meta-testing stage. Still, $\boldsymbol{\theta}^{g*}$, $\boldsymbol{\theta}^{r*}$ are fixed and $\boldsymbol{\theta}^{p*}$, $\boldsymbol{\theta}^{c*}$ are fine-tuned on $\mathcal{S}_{\text{new}}$.

3.5 Transfer Learning Across Datasets

Further, we extend PAR to handle tasks in meta-training and meta-testing come from different datasets, which requires higher generalization ability and has been considered in [1, 4, 5]. we show that our PAR can be easily adapted to conduct few-shot transfer learning across datasets by modifying Algorithm 1.

Following [1], we use the molecules from target domain task distribution $\tilde{\mathcal{T}}$ to influence the meta-training stage. After line 18, we sample a batch of tasks $T_\gamma \sim \tilde{\mathcal{T}}$ with the labeled support set $\mathcal{S}_\gamma$ and unlabeled query set $\mathcal{Q}_\gamma$ whose labels cannot be exposed during learning [1]. We then repeat line 7-16 for all T_γ. To evaluate the performance on $\mathcal{Q}_\gamma$, we propose a new loss defined as

$$\tilde{L}_{\mathcal{Q}_\gamma}(f_{\{\boldsymbol{\theta}^g, \boldsymbol{\theta}^p_\gamma, \boldsymbol{\theta}^r_\gamma, \boldsymbol{\theta}^c_\gamma\}}) = \sum\nolimits_{(x_{\gamma,j}, y_{\gamma,j}) \in \mathcal{Q}_\gamma} -[\tilde{\mathbf{y}}_{\gamma,j}]^\top \log(\tilde{\mathbf{y}}_{\gamma,j}), \tag{9}$$

$$\tilde{\mathbf{y}}_{\gamma,j} = \sum\nolimits_{(x_{\gamma,k}, y_{\gamma,k}) \in \mathcal{S}_\gamma} [\mathbf{A}^{(L)}_\gamma]_{jk} \cdot \mathbf{y}_{\gamma,k}. \tag{10}$$

This entropy-based loss can encourage the model making "confident" (low-entropy) predictions [15]. As $\tilde{\mathbf{y}}_{\gamma,j}$ is obtained by aggregating labels from its neighbors in $\mathcal{S}_\gamma$, minimizing $\tilde{L}_{\mathcal{Q}_\gamma}$ can also force the relation graph learning module learn to capture relationships in $\mathcal{G}_{T_\gamma}$ more accurately. Finally, PAR is optimized w.r.t. the accumulation of $L_{\mathcal{Q}_\tau}$ and $\tilde{L}_{\mathcal{Q}_\gamma}$ calculated over all sampled tasks.

4 Experiments

In this section, we evaluate the proposed PAR on benchmark molecular property prediction datasets, on a PC with 32GB memory, Intel-i8 CPU and a 32GB NVIDIA Tesla V100 GPU.

4.1 Experimental Settings

Datasets. We perform experiments on widely used benchmark few-shot molecular property prediction datasets[1] (Table 1): (i) Tox21 [10] contains assays each measuring the human toxicity of a biological target; (ii) SIDER [24] records the side effects for compounds used in marketed medicines. where the original 5868 side effect categories are grouped into 27 categories as in [4, 16]; (iii) MUV [34] is designed to validate virtual screening where active molecules are chosen to be structurally distinct from each another. We use public task splits provided by [4]. We also conduct experiments on ToxCast [32] whose results are put in Appendix B due to space limit.

Table 1: Summary of datasets used.

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

Baselines and Setup. We compare our PAR (Algorithm 1) with three types of baselines: (i) FSL methods originally proposed for image classification including ProtoNet [36] and MAML [9] which are classic in FSL; and GNN [12], TPN [27], and EGNN [20] which can learn relation graph during learning; (ii) FSL methods designed for molecular property prediction including IterRefLSTM [4]; and (iii) Pretained graph-based molecular encoder Pre-GNN [19] which is fined-tuned using meta-training tasks, Meta-MGNN [16] which uses trained Pre-GNN, and our PAR with trained Pre-GNN. GROVER [35] is not compared as it uses a different set of atom and bond features.

We use results of IterRefLSTM reported in [4] as its code is not available. For the other methods, we implement them using public codes of the respective authors (see Appendix A for details). We find hyperparameters using the validation set via grid search for all methods.

We use RDKit [25] to obtain molecular graphs for raw SMILES, and extract atom features (atom number and chirality tag) and bond features (bond type and bond direction) as in [19, 16]. Then, we use graph isomorphism networks (GIN) [43] as the graph-based molecular encoder to extract molecular embeddings in all methods (including ours) consistently. Following [19, 16], we use GIN provided by the authors of [19]: it consists 5 GNN layers with 300 dimensional hidden units ($d^g = 300$), take average pooling as the `READOUT` function, and set dropout rate as 0.5.

In PAR, `MLP` used in (2) and (3) both consist of two fully connected layers with hidden size 128. We iterate between relation graph estimation and molecular embedding refinement for two times. We implement PAR in PyTorch [30] and Pytorch Geometric library [8]. We train the model for a maximum number of 2000 epochs. We use Adam [21] with a learning rate 0.001 for meta training and a learning rate 0.05 for fine-tuning property-aware molecule embedding function and classifier within each task. We early stop training if the validation loss does not decrease for ten consecutive epochs. Dropout rate is 0.1 except for the graph-based molecular encoder.

Evaluation Metrics. Following [19, 16], we evaluate the binary classification performance by ROC-AUC scores calculated over 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 across all meta-testing tasks.

4.2 Results on FSL for Molecular Prediction.

Table 2 shows the results. As can be seen, our proposed PAR consistently obtains the best performance, with an average of % relative performance gain over all datasets. The previous state-of-the-art IterRefLSTM and Meta-MGNN obtain slightly better performance than Pre-GNN which is pretrained from large-scale self-supervised tasks. We also observe that FSL methods that learn relation graphs (i.e., GNN, TPN, EGNN) obtain better performance than the classic ProtoNet and MAML.

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

Table 2: ROC-AUC scores of FSL on molecule property prediction datasets. Best and comparable results are in bold, and second best ones are underlined.

Group	Method	Tox21		Sider		MUV	
		10-shot	1-shot	10-shot	1-shot	10-shot	1-shot
(i)	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
	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
	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
	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
(ii)	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
	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
	PAR(ours)	82.06±0.12	80.46±0.13	74.68±0.31	71.87±0.48	66.48±2.12	64.12±1.18
(iii)	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
	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
	Pre-PAR(ours)	**84.93±0.11**	**83.01±0.09**	**78.08±0.16**	**74.46±0.29**	**69.96±1.37**	**66.94±1.12**

4.3 Results on Few-shot Transfer Learning across Datasets

Further, we evaluate the extension of PAR (denote as PAR-TL) in Section 3.5 for few-shot transfer learning across different datasets. Following [4, 1], we consider transfer learning between Tox21 and SIDER which contain distinct tasks. In addition to baselines presented in Section 4.1, we compare with the state-of-the-art DTCR [1]. As the authors did not release codes, we use their reported results. Following [1], we compute ROC-AUC scores on query set of all tasks in the target datasets.

Table 3 presents the results of transfer learning. PAR-TL and DTCR outperform the others. PAR-TL can be easily trained while DTCR requires adversarial learning to align source and target domain. As shown, directly applying FSL methods cannot obtain satisfactory results, which is also observed in [4].

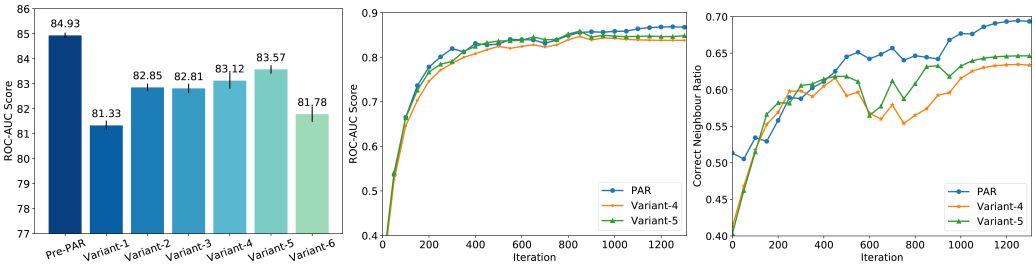
Table 3: ROC-AUC scores of few-shot transfer learning across datasets. Best and comparable results are in bold, and second best ones are underlined.

Group	Method	Tox21 → Sider		Sider → Tox21	
		10-shot	1-shot	10-shot	1-shot
Few-shot	ProtoNet	56.71±4.89	53.80±3.52	67.07±6.38	58.73±5.24
	TPN	57.50±3.97	50.31±1.43	67.80±5.52	50.02±0.87
	MAML	56.84±2.34	54.68±2.46	65.20±4.77	63.53±1.56
	Multi-task	56.00±2.00	53.00±3.00	63.00±4.00	60.00±1.00
	EGNN	57.82±2.39	55.96±2.45	68.40±1.25	65.50±1.20
	PAR	58.44±1.51	57.03±2.08	69.08±1.34	66.57±1.02
Few-shot transfer	DTCR	63.00±2.00	60.00±2.00	74.00±2.00	69.00±2.00
	PAR-TL	63.12±1.13	60.12±1.38	75.23±2.44	70.12±2.73
Few-shot with Pretrained GNN	Pre-GNN	61.12±0.82	58.29±1.78	73.77±1.52	65.62±1.77
	Meta-MGNN	61.99±1.43	58.89±2.38	74.26±2.18	67.27±1.37
	Pre-PAR	62.20±1.32	58.77±2.43	74.40±1.97	69.48±1.66
	Pre-PAR-TL	**65.19±0.97**	**61.49±2.08**	**78.05±1.33**	**71.32±1.09**

4.4 Model Analysis for PAR

We now examine the effectiveness of each elements in our proposed methods via ablation study.

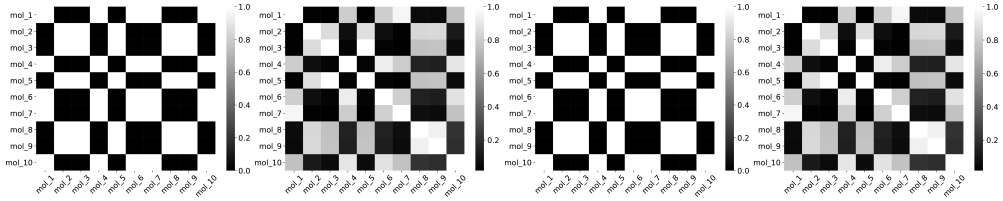
We compare the PAR with the following variants: (1) w/o property-aware embedding; (2) w/o context $\mathbf{e}^c_{x_{\tau,i}}$ in equation (2); (3) w/o adaptive relation graph learning; (4) w/o reducing $\mathcal{G}_{T_\tau}$ to KNN graph; (5) w/o the neighbor alignment loss in equation (6), and (6) fine-tune all parameters on line 15 of Algorithm 1. Figure 3 shows the results. As can be seen, the design of property-aware embedding and adaptive relation graph learning are vital to the success of PAR. We calculate the correct neighbor ratio among the top K nearest neighbors averaged over all molecules in query set of meta-testing tasks. We consider 2-way-10-shot learning on the 11th assay of Tox21, where we compare the predictions for the overlapping 10 molecules. During learning, PAR can improve both the correct neighbor ratio and the overall ROC-AUC scores. The consistency between neighbor alignment loss and ROC-AUC scores further validates the efficacy of the additional neighbor alignment loss in (6).



(a) (b) (c)

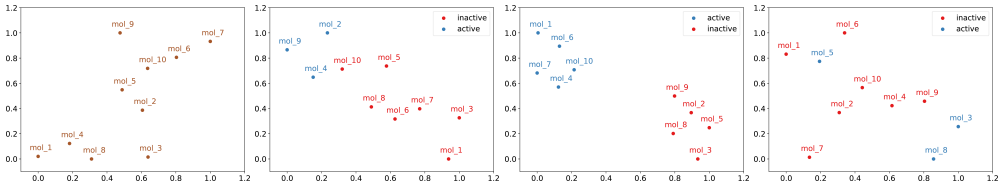
Figure 3: Ablation study of different variants (a), ROC-AUC scores vs iterations (b), and correct neighbor ratio vs iterations during training on Tox21.

Finally, we validate whether PAR can obtain different property-aware molecular embeddings and relation graphs for two tasks contain overlapping molecules but evaluate different properties, To validate this, we consider a hard case: we sample a fixed group of 10 molecules on Tox21 (whose details are in the Appendix B) and evaluate them in different meta-testing tasks. Figure 4 compares the adjacency matrices estimated by PAR with the ground truth ones. As can be seen, PAR obtain different adjacency matrices in different property-prediction tasks. Moreover, these adjacency matrices are similar to the ground truth ones. Finally, we use t-SNE [37] to visually compare $\mathbf{e}^g_{x_{\mathcal{T},i}}, \mathbf{e}^p_{x_{\mathcal{T},i}}, \mathbf{e}^r_{x_{\mathcal{T},i}}$ of these 10 molecules in Figure 5. As shown, similar molecules gradually are clustered together during passing the graph-based molecular encoder, property-aware embedding function and the adaptive relation graph learning module. Besides, the clustering patterns are different across different tasks.



(a) (b) (c) (d)

Figure 4: Comparison between adjacency matrix $\hat{\mathbf{A}}_{T_\tau}$ returned by PAR ((a) for the 11th task and (c) for the 12th task) and the ground truth $\mathbf{A}^*_\tau$ ((b) for the 11th task and (d) for the 12th task) on Tox21.



(a) $\mathbf{e}^g_{x_{\mathcal{T},i}}$ (b) 10th task (c) 11th task (d) 11th task

Figure 5: t-SNE visualization of $\mathbf{e}^r_{x_{\mathcal{T},i}}$ on Tox21.

5 Conclusion

We propose a property-aware adaptive relation network (PAR) for a few-shot molecular property prediction problem. PAR consists of three components: a graph-based molecular encoder to encode the topological structure of the molecular graph, atom features, and bond features into a molecular embedding, a property-aware embedding projection to obtain property-aware embeddings encoding context information of each task; and an adaptive relation graph learning to construct a relation graph to effectively propagate information among similar molecules. We empirically evaluate PAR on many benchmark drug property prediction datasets. Results consistently show that PAR outperforms state-of-the-art methods under both standard few-shot learning settings and transfer learning across different datasets setting.

References

[1] 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. *Journal of Chemical Information and Modeling*, 59(11):4528–4539, 2019.

[2] S. Ajmani, K. Jadhav, and S. A. Kulkarni. Group-based QSAR (G-QSAR): Mitigating interpretation challenges in QSAR. *QSAR & Combinatorial Science*, 28(1):36–51, 2009.

[3] E. Alsentzer, S. Finlayson, M. Li, and M. Zitnik. Subgraph neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 8017–8029, 2020.

[4] H. Altae-Tran, B. Ramsundar, A. S. Pappu, and V. Pande. Low data drug discovery with one-shot learning. *ACS Central Science*, 3(4):283–293, 2017.

[5] C. Cai, S. Wang, Y. Xu, W. Zhang, K. Tang, Q. Ouyang, L. Lai, and J. Pei. Transfer learning for drug discovery. *Journal of Medicinal Chemistry*, 63(16):8683–8694, 2020.

[6] G. E. Dahl, N. Jaitly, and R. Salakhutdinov. Multi-task neural networks for QSAR predictions. *arXiv preprint arXiv:1406.1231*, 2014.

[7] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2015.

[8] M. Fey and J. E. Lenssen. Fast graph representation learning with PyTorch Geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[9] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[10] N. C. for Advancing Translational Sciences. Tox21 challenge. `http://tripod.nih.gov/tox21/challenge/`, 2017. Accessed: 2016-11-06.

[11] X. Fu, J. Zhang, Z. Meng, and I. King. MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding. In *The Web Conference*, pages 2331–2341, 2020.

[12] V. Garcia and J. Bruna. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018.

[13] E. Gawehn, J. A. Hiss, and G. Schneider. Deep learning in drug discovery. *Molecular Informatics*, 35(1):3–14, 2016.

[14] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272, 2017.

[15] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Advances on Neural Information Processing Systems*, pages 529–536, 2004.

[16] Z. Guo, C. Zhang, W. Yu, J. Herr, O. Wiest, M. Jiang, and N. V. Chawla. Few-shot graph learning for molecular property prediction. In *The Web Conference*, 2021.

[17] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1025–1035, 2017.

[18] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, volume 33, pages 22118–22133, 2020.

[19] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2019.

[20] J. Kim, T. Kim, S. Kim, and C. D. Yoo. Edge-labeling graph neural network for few-shot learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11–20, 2019.

[21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016.

[23] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2. Lille, 2015.

[24] M. Kuhn, I. Letunic, L. J. Jensen, and P. Bork. The SIDER database of drugs and side effects. *Nucleic Acids Research*, 44(D1):D1075–D1079, 2016.

[25] G. Landrum. RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling, 2013.

[26] S. P. Leelananda and S. Lindert. Computational methods in drug discovery. *Beilstein journal of organic chemistry*, 12(1):2694–2718, 2016.

[27] Y. Liu, J. Lee, M. Park, S. Kim, E. Yang, S. J. Hwang, and Y. Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *International Conference on Learning Representations*, 2018.

[28] F. Monti, K. Otness, and M. M. Bronstein. MotifNet: A motif-based graph convolutional network for directed graphs. In *IEEE Data Science Workshop*, pages 225–228. IEEE, 2018.

[29] C. Q. Nguyen, C. Kreatsoulas, and K. M. Branson. Meta-learning gnn initializations for low-resource molecular property prediction. *arXiv preprint arXiv:2003.05996v2*, pages arXiv–2003, 2020.

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[31] S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg, and A. L. Schacht. How to improve R&D productivity: The pharmaceutical industry's grand challenge. *Nature Reviews Drug Discovery*, 9(3):203–214, 2010.

[32] A. M. Richard, R. S. Judson, K. A. Houck, C. M. Grulke, P. Volarath, I. Thillainadarajah, C. Yang, J. Rathman, M. T. Martin, J. F. Wambaugh, et al. ToxCast chemical landscape: Paving the road to 21st century toxicology. *Chemical Research in Toxicology*, 29(8):1225–1251, 2016.

[33] P. Rodríguez, I. Laradji, A. Drouin, and A. Lacoste. Embedding propagation: Smoother manifold for few-shot classification. In *European Conference on Computer Vision*, pages 121–138. Springer, 2020.

[34] S. G. Rohrer and K. Baumann. Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data. *Journal of Chemical Information and Modeling*, 49(2):169–184, 2009.

[35] Y. Rong, Y. Bian, T. Xu, W. Xie, Y. Wei, W. Huang, and J. Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571, 2020.

[36] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4080–4090, 2017.

[37] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[39] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[40] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3637–3645, 2016.

[41] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):1–34, 2020.

[42] Z. Xiong, D. Wang, X. Liu, F. Zhong, X. Wan, X. Li, Z. Li, X. Luo, K. Chen, H. Jiang, et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of Medicinal Chemistry*, 63(16):8749–8760, 2019.

[43] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

[44] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388, 2019.

[45] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, and Y. Liu. DPGN: Distribution propagation graph network for few-shot learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13390–13399, 2020.

[46] W. Yu, H. Xiao, J. Lin, and C. Li. Discovery of novel STAT3 small molecule inhibitors via in silico site-directed fragment-based drug design. *Journal of Medicinal Chemistry*, 56(11):4402–4412, 2013.

[47] A. Zhavoronkov, Y. A. Ivanenkov, A. Aliper, M. S. Veselov, V. A. Aladinskiy, A. V. Aladinskaya, V. A. Terentiev, D. A. Polykovskiy, M. D. Kuznetsov, A. Asadulaev, et al. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nature Biotechnology*, 37(9):1038–1040, 2019.

[48] Y. Zhu, W. Xu, J. Zhang, Q. Liu, S. Wu, and L. Wang. Deep graph structure learning for robust representations: A survey. *arXiv preprint arXiv:2103.03036*, 2021.

[49] I. Ziko, J. Dolz, E. Granger, and I. B. Ayed. Laplacian regularized few-shot learning. In *International Conference on Machine Learning*, pages 11660–11670. PMLR, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelin es and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We adopt public splits provided by [4].

(c) Did you report error bars (e.g., w.r.t. the random seed after running experiments multiple times)? [Yes]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We mention it in the beginning of Section 4.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.1.

(b) Did you mention the license of the assets? [No]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We use the public codes of the authors.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We use public benchmark molecular property prediction datasets.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]