PACIA: Parameter-Efficient Adapter for Few-Shot Molecular Property Prediction

Shiguang Wu[1] , **Yaqing Wang**[2*] , **Quanming Yao**[1]
[1]Department of Electronic Engineering, Tsinghua University
[2]Baidu Research, Baidu Inc.
wsg23@mails.tsinghua.edu.cn, wangyaqing01@baidu.com, qyaoaa@tsinghua.edu.cn

Abstract

Molecular property prediction (MPP) plays a crucial role in biomedical applications, but it often encounters challenges due to a scarcity of labeled data. Existing works commonly adopt gradient-based strategy to update a large amount of parameters for task-level adaptation. However, the increase of adaptive parameters can lead to overfitting and poor performance. Observing that graph neural network (GNN) performs well as both encoder and predictor, we propose PACIA, a parameter-efficient GNN adapter for few-shot MPP. We design a unified adapter to generate a few adaptive parameters to modulate the message passing process of GNN. We then adopt a hierarchical adaptation mechanism to adapt the encoder at task-level and the predictor at query-level by the unified GNN adapter. Extensive results show that PACIA obtains the state-of-the-art performance in few-shot MPP problems, and our proposed hierarchical adaptation mechanism is rational and effective.

1 Introduction

Molecular property prediction (MPP) which predicts whether desired properties will be active on given molecules, can be naturally modeled as a few-shot learning problem [Waring *et al.*, 2015; Altae-Tran *et al.*, 2017]. As wet-lab experiments to evaluate the actual properties of molecules are expensive and risky, usually only a few labeled molecules are available for a specific property. While recently, Graph Neural Network (GNN) is popularly used to learn molecular representations [Xu *et al.*, 2019; Yang *et al.*, 2019; Xiong *et al.*, 2019]. Modeling molecules as graphs, GNN can capture inherent structural information. Hence, GNN-based methods obtain better performance than classical ones [Unterthiner *et al.*, 2014; Ma *et al.*, 2015], especially when they are pretrained on self-learning tasks constructed from additional large scale corpus. As for tasks with only a few labeled molecules, the performance of existing GNN-based methods is still far from desired.

Various few-shot learning (FSL) methods have been developed to handle few-shot MPP problem. The earlier work IterRefLSTM [Altae-Tran *et al.*, 2017] builds a metric-based model upon matching network [Vinyals *et al.*, 2016]. Subsequent works mainly adopt gradient-based meta-learning strategy [Finn *et al.*, 2017], which learns parameter initialization with good generalizability across different properties and adapts parameters by gradient descent for target property. Specifically, Meta-MGNN [Guo *et al.*, 2021] brings chemical prior knowledge in the form of molecular reconstruction loss, and optimizes all parameters by gradient descents. PAR [Wang *et al.*, 2021] introduces attention and relation graph module to better utilize the labeled samples for property-adaptation with the awareness of target chemical property, and conducts a selective gradient-based meta-learning strategy. ADKF-IFT [Chen *et al.*, 2022] takes a gradient-based meta-learning strategy with implicit function theorem to avoid computing expensive hypergradients, and builds a Gaussian Process for each task as classifier. There are also works that bring auxiliary information such as additional reference molecules from large molecule database [Schimunek *et al.*, 2023] and auxiliary properties [Zhuang *et al.*, 2023] to improve the performance of few-shot MPP [Schimunek *et al.*, 2023; Zhuang *et al.*, 2023].

Two primary issues persist in existing studies. First, gradient-based meta-learning strategy requires updating a large number of parameters in order to adapt to each task. This results in poor learning efficiency and is prone to overfit given insufficient labeled samples [Rajeswaran *et al.*, 2019; Yin *et al.*, 2020], as demonstrated in Figure 1 (a). While with more task-specific parameters, the model gets easily overfit, which would be more severe in extreme few-shot cases. Effective adaptation should be made in a parameter-efficient way, i.e, without modulating a large amount of parameters. Second, query-level adaptation is absent but it is important specially for few-shot MPP. The chemical space is enormous and the representations of molecules vary in a wide range. The query-level difference should be addressed when classifying the encoded molecules. When query molecules are more similar to the labeled molecules in one class, they can be easily classified. While others exhibit comparable similarity to both categories, they will be harder to be accurately classified. Thus, a fixed predictor can not fit all molecules even in a single task.

*Corresponding author.

In this paper, we first summarize existing works into an encoder-predictor framework, where GNN performs well acting as both encoder and predictor. Upon the framework, we propose **PACIA**, a **PA**rameter-effi**CI**ent **A**dapter for few-shot MPP problem. To sum up, our contributions are as follows:

- We propose query-level adaptation for few-shot MPP problem and design a hierarchical adaptation mechanism for the encoder-predictor framework generally adopted by existing approaches.
- We design a hypernetwork-based GNN adapter to achieve parameter-efficient adaptation. This unified GNN adapter can generate a few adaptive parameters to modulate the message passing process of GNN in two aspects: node embedding and propagation depth.
- We conduct extensive results, and show PACIA obtains the state-of-the-art performance on benchmark few-shot MPP datasets MoleculeNet [Wu *et al.*, 2018] and FS-Mol [Stanley *et al.*, 2021]. We also closely examine and validate the effectiveness of our hierarchical adaptation mechanism.

2 Related Works

2.1 Few-Shot Learning

Few-shot learning (FSL) aims to generalize to a task with a few labeled samples [Wang *et al.*, 2020]. In terms of adaptation mechanism, existing FSL methods can be classified into three main categories: (i) gradient-based approaches [Finn *et al.*, 2017; Grant *et al.*, 2018] learn a model which can be generalized to new task by gradient descents, (ii) metric-based approaches [Vinyals *et al.*, 2016; Snell *et al.*, 2017] learn to embed samples into a space where similar and dissimilar samples can be easily discriminated by a distance function, and (iii) amortization-based approaches [Requeima *et al.*, 2019; Lin *et al.*, 2021; Przewiezlikowski *et al.*, 2022] use hypernetworks to map the labeled samples in the task to a few parameters to adjust the main networks to be task-specific. Recent works [Requeima *et al.*, 2019] found that amortization-based approaches can reduce the risk of overfitting compared with gradient-based approaches. They also have faster inference speed as the adapted parameters are generated by a single forward pass without taking optimization steps. Besides, the main networks can approximate various functions in addition to distance-based ones.

2.2 Hypernetworks

Hypernetworks [Ha *et al.*, 2017] refer to neural networks which learn to generate parameters of the main network which handles the target tasks. Hypernetworks have been successfully used in various applications like cold-start recommendation [Lin *et al.*, 2021] and image classification [Przewiezlikowski *et al.*, 2022]. Designing appropriate hypernetworks is challenging, requiring domain knowledge to decide what information to be fed into hypernetworks, how to adapt the main network, and what is the appropriate architecture of hypernetworks. For general GNNs, hypernetworks are developed to modulate weight matrix in aggregation function in message passing [Brockschmidt, 2020], or to facilitate node-specific message passing [Nachmani and Wolf, 2020]. In contrast to them, we particularly consider designing parameter-efficient modulators for GNNs used in encoder-predictor framework for few-shot MPP.

3 Preliminaries: Few-Shot MPP

3.1 Problem Setup

In a few-shot MPP task $\mathcal{T}_\tau$ with respect to a specific property, each sample $\mathcal{X}_{\tau,i}$ is a molecular graph and its label $y_{\tau,i} \in \{0, 1\}$ records whether the molecule is active or inactive on the target property. Only a few labeled samples are available in $\mathcal{T}_\tau$. Following earlier works [Altae-Tran *et al.*, 2017; Stanley *et al.*, 2021; Chen *et al.*, 2022; Schimunek *et al.*, 2023], we model a $\mathcal{T}_\tau$ as a 2-way classification task $\mathcal{T}_\tau$, associating with a support set $\mathcal{S}_\tau = \{(\mathcal{X}_{\tau,s}, y_{\tau,s})\}_{s=1}^{N_\tau}$ containing labeled samples from active/inactive class, and a query set $\mathcal{Q}_\tau = \{(\mathcal{X}_{\tau,q}, y_{\tau,q})\}_{q=1}^{M_\tau}$ containing M_τ samples whose labels are only used for evaluation. We consider both (i) balanced support sets, i.e., $\mathcal{S}_\tau$ contains $\frac{N_\tau}{2}$ samples per class which is consistent with the standard N-way K-shot FSL setting [Altae-Tran *et al.*, 2017], and (ii) imbalanced support sets which exist in real-world applications [Stanley *et al.*, 2021]. Our target is to learn a model from a set of tasks $\{\mathcal{T}_\tau\}_{j=1}^{N}$ that can generalize to new task given the few-shot support set. Specifically, the target properties are different across tasks.

3.2 Encoder-Predictor Framework

In the past, molecules are encoded with certain properties (fingerprint vectors [Rogers and Hahn, 2010]) and fed to deep networks for prediction [Unterthiner *et al.*, 2014; Ma *et al.*, 2015]. While recently, GNNs are popularly taken as molecular encoders [Li *et al.*, 2018; Yang *et al.*, 2019; Xiong *et al.*, 2019; Hu *et al.*, 2019] due to their superior performance on learning from of topological data. In either case, existing works can be summarized within an encoder-predictor framework.

Consider a molecular graph $\mathcal{X} = \{\mathcal{V}, \mathcal{E}\}$ with node feature $\mathbf{h}_v$ for each atom $v \in \mathcal{V}$ and edge feature $\mathbf{b}_{vu}$ for each chemical bond $e_{vu} \in \mathcal{E}$ between atoms v, u. A GNN encoder maps $\mathcal{X}$ to molecular representation $\mathbf{r}$ which is a fixed-length vector. At the lth layer, GNN updates atom embedding $\mathbf{h}_v^l$ of v as

$$\mathbf{h}_v^l = \texttt{UPD}^l\left(\mathbf{h}_v^{l-1}, \texttt{AGG}^l\left(\{(\mathbf{h}_v^{l-1}, \mathbf{h}_u^{l-1}, \mathbf{b}_{vu}) | u \in \mathcal{H}(v)\}\right)\right), \quad (1)$$

where $\mathcal{H}(v)$ contains neighbors of v. $\texttt{AGG}(\cdot)$ and $\texttt{UPD}(\cdot)$ are aggregation and updating functions respectively. After L layers, the query-level representation $\mathbf{r}$ for $\mathcal{X}$ is obtained as

$$\mathbf{r} = \texttt{READOUT}\left(\{\mathbf{h}_v^L | v \in \mathcal{V}\}\right), \quad (2)$$

where $\texttt{READOUT}(\cdot)$ function aggregates atom embeddings.

Then, a predictor $f(\cdot)$ assigns label for a query molecule $\mathcal{X}_{\tau,q}$ given support molecules in $\mathcal{S}_\tau$:

$$\hat{\mathbf{y}}_{\tau,q} = f(\mathbf{r}_{\tau,q} | \{\mathbf{r}_{\tau,s}\}_{s \in \mathcal{S}_\tau}). \quad (3)$$

The specific choice of $f(\cdot)$ is diverse, e.g., pair-wise similarity [Altae-Tran *et al.*, 2017], multi-layer perceptron (MLP) [Guo *et al.*, 2021; Wang *et al.*, 2021] and Mahalanobis distance [Stanley *et al.*, 2021]. Recently, GNN-based predictor

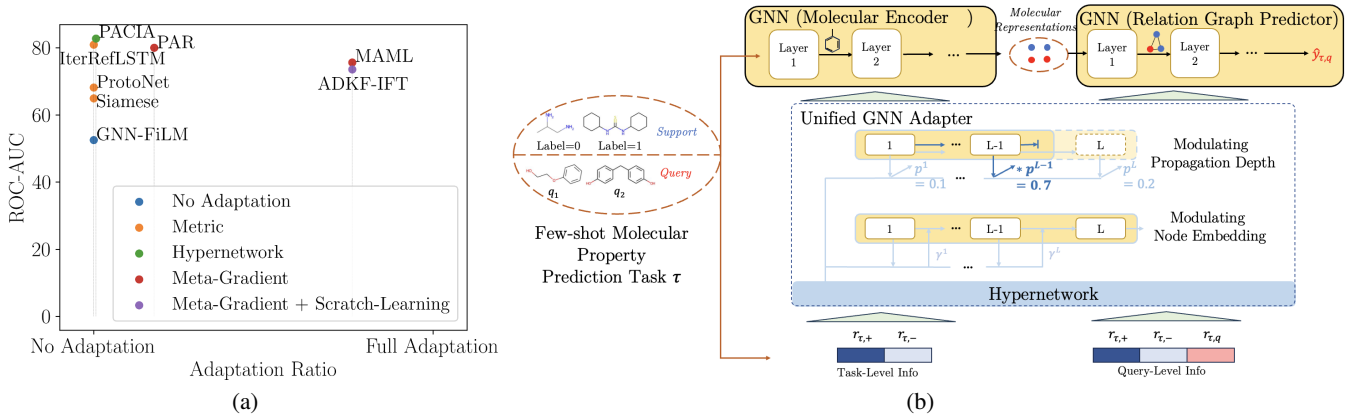


Figure 1: Illustration of the proposed PACIA. (a). The adaptation ratio ($\frac{|adaptive\ params|}{|total\ params|}$) of different methods and their testing performance on 1-shot tasks of Tox21. (b). Under the encoder-predictor framework, PACIA uses a unified GNN adapter, where hypernetworks are used to generate adaptive parameters, to modulate node embeddings and propagation depths in both GNN encoder and predictor.

which operates on relation graphs of molecules is found to effectively compensate for the lack of supervised information [Wang *et al.*, 2021]. In particular, molecular representations are refined on relation graphs such that the similar molecules cluster closer. Initialize molecular representations as the output of the encoder, i.e., $\mathbf{h}^0_{\tau,i} = \mathbf{r}_{\tau,i}$. Denote the set of $N_\tau + 1$ molecules as $\mathcal{R}_{\tau,q} = (\mathcal{X}_{\tau,q}, y_{\tau,q}) \cup \mathcal{S}_\tau$, which contains all information to make prediction for the query molecule $\mathcal{X}_{\tau,q}$. The relation graph works by recurrently estimating the adjacency matrix and updating the molecular representations. At the lth layer, each element a^l_{ij} in the adjacent matrix $\mathbf{A}^l_{\tau,q}$ of the relation graph is learned to represent pair-wise similarities between any two molecules in $\mathcal{R}_{\tau,q}$:

$$a^l_{ij} = \begin{cases} \texttt{MLP}\left(|\mathbf{h}^{l-1}_{\tau,i} - \mathbf{h}^{l-1}_{\tau,j}|\right) & \text{if } i \neq j \\ 1 & \text{otherwise} \end{cases}. \tag{4}$$

Then, each molecular representation is refined as

$$\mathbf{h}^l_{\tau,i} = \texttt{MLP}\Big(\sum\nolimits_{j=1}^{N_\tau+1} a^l_{ij}\mathbf{h}^{l-1}_{\tau,j}\Big). \tag{5}$$

After L layers of refinement, $\mathbf{h}^L_{\tau,q}$ and $\mathbf{h}^L_{\tau,s}$ (in place of $\mathbf{r}_{\tau,q}$ and $\mathbf{r}_{\tau,s}$) are fed to (3) to obtain final prediction $\hat{\boldsymbol{y}}_{\tau,q}$ for $\mathcal{X}_\tau$.

4 Hierarchical Adaptation of Encoder-Predictor Framework

To generalize across different tasks with a few labeled molecules, existing works [Wang *et al.*, 2021; Chen *et al.*, 2022] usually conduct task-level adaptation by gradient-based meta-learning. (see Appendix). However, as discussed in Section 2, gradient-based meta-learning optimizes most parameters using a few labeled molecules, which is slow to optimize and easy to overfit. As for query-level adaptation, gradient is not accessible for each query molecule in testing since the label is unknown during training. Therefore, we turn to hypernetworks to achieve parameter-efficient adaptation. Task-level adaptation is achieved in encoder since the structural features in molecular graphs needs to be captured in a property-adaptive manner, while query-level adaptation is achieved in predictor based on the property-adaptive representations.

As discussed in Section 3.2, GNNs can effectively act as both encoder and predictor. Therefore, we propose PACIA (Figure 1), a method using a unified GNN adapter to generate a few adaptive parameters to hierarchically adapt the encoder at task-level and the predictor at query-level in a parameter-efficient manner. In the sequel, we provide the details of the unified GNN adapter (Section 4.1), then describe how to learn the main networks (including encoder and predictor) with the unified GNN adapter by episodic training (Section 4.2). Finally, we present a comparative discussion of PACIA in relation to existing works (Section 4.3).

4.1 A Unified GNN Adapter

To adapt GNN's parameter-efficiently, we design a GNN adapter to modulate the node embedding and propagation depth, which are essential in message passing process.

Modulating Node Embedding. Denote the node embedding at the lth layer as $\mathbf{h}^l$, which can be atom embedding in encoder or molecular embedding in relation graph predictor. We obtain adapted embedding $\hat{\mathbf{h}}^l$ as

$$\hat{\mathbf{h}}^l = e(\mathbf{h}^l, \boldsymbol{\gamma}^l), \tag{6}$$

where $e(\cdot)$ is an element-wise function, and $\boldsymbol{\gamma}^l$ is adaptive parameter generated by the hypernetworks. This adapted embedding $\hat{\mathbf{h}}^l$ is then fed to next layer of GNN.

Modulating Propagation Depth. Further, we manage to modulate the propagation depth, i.e., layer number l of a GNN. Controlling l is challenging since it is discrete. We achieve this by training a differentiable controller, which is a hypernetwork to generate a scalar p^l corresponding to the lth layer. The value of p^l estimates how likely the message passing should stop after the lth layer. Finally, assume there are L layers in total, the vector

$$\boldsymbol{p} = \texttt{softmax}([\,p^1,\ p^2,\ \cdots,\ p^L\,]), \tag{7}$$

represents the plausibility of choosing each layer. The hypernetwork is shared across all L layers. During meta-training, $\boldsymbol{p}$ is used as differentiable weight to modulate GNN layers. Specifically, after propagation through all L layers, the final node embedding $\widetilde{\mathbf{h}}$ is obtained as

$$\widetilde{\mathbf{h}} = \sum\nolimits_{l=1}^{L} [\boldsymbol{p}]_l \mathbf{h}^l, \tag{8}$$

where $[\boldsymbol{p}]_l$ is the lth element of $\boldsymbol{p}$.

Generating Adaptive Parameters. We generate adaptive parameters $\{\boldsymbol{\gamma}^l, p^l\}_{l=1}^{L}$ by hypernetworks. In particular, note that the generated adaptive parameter should be permutation-invariant to the order of input samples in $\mathcal{S}_\tau$. Therefore, we first calculate class prototypes $\boldsymbol{r}^l_{\tau,+}$ and $\boldsymbol{r}^l_{\tau,-}$ of active class (+) and inactive class (-) for samples in $\mathcal{S}_\tau$ by

$$\begin{aligned}
\boldsymbol{r}^l_{\tau,+} &= \frac{1}{|\mathcal{S}^+_\tau||\mathcal{V}_{\tau,s}|} \sum\nolimits_{\mathcal{X}_{\tau,s}\in\mathcal{S}^+_\tau} \texttt{MLP}\big(\big[\sum\nolimits_{v\in\mathcal{X}_{\tau,s}} \mathbf{h}^l_v | \boldsymbol{y}_{\tau,s}\big]\big), \\
\boldsymbol{r}^l_{\tau,-} &= \frac{1}{|\mathcal{S}^-_\tau||\mathcal{V}_{\tau,s}|} \sum\nolimits_{\mathcal{X}_{\tau,s}\in\mathcal{S}^-_\tau} \texttt{MLP}\big(\big[\sum\nolimits_{v\in\mathcal{X}_{\tau,s}} \mathbf{h}^l_v | \boldsymbol{y}_{\tau,s}\big]\big),
\end{aligned} \tag{9}$$

where $[\cdot|\cdot]$ means concatenating, $\mathcal{S}^+_\tau$ and $\mathcal{S}^-_\tau$ are the sets of active and inactive samples in $\mathcal{S}_\tau$, and $\boldsymbol{y}_{\tau,s}$ is the one-hot encoding of label. Using $\boldsymbol{r}^l_{\tau,+}$ and $\boldsymbol{r}^l_{\tau,-}$ allows subsequent steps to keep supervised information while being permutation-invariant.

Recall that task-level adaptation and query-level adaptation is achieved in encoder and predictor respectively. For task-level adaptation, we then map $\boldsymbol{r}^l_{\tau,+}$ and $\boldsymbol{r}^l_{\tau,-}$ to $\boldsymbol{\gamma}^l_\tau, p^l_\tau$ as

$$[\boldsymbol{\gamma}^l_\tau, p^l_\tau] = \texttt{MLP}\left(\left[\boldsymbol{r}^l_{\tau,+} \mid \boldsymbol{r}^l_{\tau,-}\right]\right). \tag{10}$$

As for query-level adaptation, information comes from both $\mathcal{S}_\tau$ and the specific query molecule $\mathcal{X}_{\tau,q}$. Likewise, we use class prototypes $\boldsymbol{r}^l_{\tau,+}$ and $\boldsymbol{r}^l_{\tau,-}$ to keep permutation-invariant. We then generate $\boldsymbol{\gamma}^l_{\tau,q}, p^l_{\tau,q}$ as

$$[\boldsymbol{\gamma}^l_{\tau,q}, p^l_{\tau,q}] = \texttt{MLP}\Big(\Big[\mathbf{r}^l_{\tau,+} \mid \mathbf{r}^l_{\tau,-} \mid \sum\nolimits_{v\in\mathcal{X}_{\tau,q}} \mathbf{h}^l_v\Big]\Big), \tag{11}$$

combining information in $\mathcal{X}_{\tau,q}$ and $\mathcal{S}_\tau$. Note that parameters of these MLPs in hypernetworks are jointly meta-learned with the encoder and predictor.

4.2 Learning and Inference

Denote the collection of all model parameters in main network((1)-(5)) and hypernetwork ((9)-(11)) as $\boldsymbol{\Theta}$, excluding adaptive parameters. Our objective takes the form:

$$\min \sum\nolimits_{\tau=1}^{N} \mathcal{L}_\tau, \text{with } \mathcal{L}_\tau = -\sum\nolimits_{\boldsymbol{x}_{\tau,q}\in\mathcal{Q}_\tau} \boldsymbol{y}^\top_{\tau,q} \log\left(\hat{\boldsymbol{y}}_{\tau,q}\right). \tag{12}$$

$\mathcal{L}_\tau$ is the loss in task $\mathcal{T}_\tau$, $\boldsymbol{y}_{\tau,q}$ is one-hot ground-truth label vector and $\hat{\boldsymbol{y}}_{\tau,q}$ is prediction obtained by (3).

Note that $\boldsymbol{\Theta}$ is shared across all tasks. While the adaptive parameter $\{\boldsymbol{\gamma}^l\}_{l=1}^{L}$ in (6) and $\{p^l\}_{l=1}^{L}$ in (7) are generated by hypernetworks. The size of adaptive parameter is far smaller than the main network. This realizes parameter-efficient adaptation and mitigates the risk of overfitting.

Algorithm 1 Meta-training procedure of PACIA.

Input: meta-training task set $\boldsymbol{\mathcal{T}}_{\text{train}}$;
1: initialize $\boldsymbol{\Theta}$ randomly or use a pretrained one;
2: **while** not done **do**
3: **for** each task $\mathcal{T}_\tau \in \boldsymbol{\mathcal{T}}_{\text{train}}$ **do**
4: **for** $l \in \{1, 2, \cdots, L_{\text{enc}}\}$ **do**
5: + generate $[\boldsymbol{\gamma}^l_\tau, p^l_\tau]$ by (10);
6: modulate atom embedding $\mathbf{h}^l_v \leftarrow e(\mathbf{h}^l_v, \boldsymbol{\gamma}^l_\tau)$;
7: * update atom embedding $\mathbf{h}^l_v$ by (1);
8: **end for**
9: obtain atom embedding after message passing $\mathbf{h}^{L_{\text{enc}}}_v \leftarrow \sum_{l=1}^{L_{\text{enc}}} [\boldsymbol{p}_\tau]_l \mathbf{h}^l_v$ and obtain molecular embeddings by (2);
10: **end for**
11: **for** each query $(\mathcal{X}_{\tau,q}, y_{\tau,q}) \in \mathcal{Q}_\tau$ **do**
12: **for** $l \in \{1, 2, \cdots, L_{\text{rel}}\}$ **do**
13: + generate $[\boldsymbol{\gamma}^l_{\tau,q}, p^l_{\tau,q}]$s by (11);
14: modulate molecular embedding $\mathbf{h}^l_{\tau,i} \leftarrow e(\mathbf{h}^l_{\tau,i}\boldsymbol{\gamma}^l_{\tau,q})$;
15: * update molecular embedding by (4)-(5);
16: **end for**
17: obtain molecular embedding after message passing $\mathbf{h}^{L_{\text{rel}}}_{\tau,i} \leftarrow \sum_{l=1}^{L_{\text{rel}}} [\boldsymbol{p}_{\tau,q}]_l \mathbf{h}^l_{\tau,i}$;
18: obtain prediction $\hat{\mathbf{y}}_{\tau,q}$ by (3);
19: **end for**
20: calculate loss by (12);
21: update $\boldsymbol{\Theta}$ by gradient descent;
22: **end while**
23: **return** learned $\boldsymbol{\Theta}^*$.

Algorithm 1[1] summarizes the training procedure of PACIA. As mentioned above, our unified GNN adapter can simultaneously modulate the node embedding and propagation depth, and be cascaded to adapt both encoder and predictor. During training, molecular graph $\mathcal{X}_{\tau,i}$ is first processed by encoder (line 4-9). At each layer, adaptive parameters $[\boldsymbol{\gamma}^l_\tau, p^l_\tau]$ are obtained by (10) (line 5). Then, (6) modulates all atom embeddings $\mathbf{h}^l_v$ (line 6). After L_{enc} layers of message passing (1), (8) is applied before (2), to get property-adaptive molecular representations and initialize node embeddings $\mathbf{h}^0_{\tau,i} = \mathbf{r}_{\tau,i}$ in relation graph (line 9). Then in predictor, at each layer of GNN on relation graph, adaptive parameters $[\boldsymbol{\gamma}^l_{\tau,q}, p^l_{\tau,q}]$ are obtained with (11) (line 13) and (6) modulates all node embeddings $\mathbf{h}^l_{\tau,i}$ (line 14). After L_{rel} layers of message passing by (4)(5), (8) is applied (line 17). The final prediction $\hat{\mathbf{y}}_{\tau,q}$ is obtained by (3).

Testing procedure is provided in Appendix. The process is similar. A noteworthy difference is the propagation depth is adapted by selecting the layer with maximal plausibility:

$$l' = \text{argmax}_{l\in\{1,2,\cdots,L\}}\ p^l. \tag{13}$$

Only $\mathbf{h}^{l'}$, rather than (8), is fed forward to the next module.

[1] In Algorithm 1,"*" (resp. "+") indicates the step is executed by the main network (resp. hypernetwork).

Method	Tox21		SIDER		MUV		ToxCast	
	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot	10-shot	1-shot
GNN-ST	$61.23_{(0.89)}$	$55.49_{(2.31)}$	$56.25_{(1.50)}$	$52.98_{(2.12)}$	$54.26_{(3.61)}$	$51.42_{(5.11)}$	$55.66_{(1.47)}$	$51.80_{(1.99)}$
MAT	$64.84_{(0.93)}$	$54.90_{(1.89)}$	$57.45_{(1.26)}$	$52.97_{(3.00)}$	$56.19_{(2.88)}$	$52.01_{(4.05)}$	$58.50_{(1.62)}$	$52.41_{(2.34)}$
GNN-MT	$69.56_{(1.10)}$	$62.08_{(1.25)}$	$60.97_{(1.02)}$	$55.39_{(1.83)}$	$66.24_{(2.40)}$	$60.78_{(2.91)}$	$65.72_{(1.19)}$	$62.38_{(1.67)}$
ProtoNet	$72.99_{(0.56)}$	$68.22_{(0.46)}$	$61.34_{(1.08)}$	$57.41_{(0.76)}$	$68.92_{(1.64)}$	$64.81_{(1.95)}$	$65.29_{(0.82)}$	$63.73_{(1.18)}$
MAML	$79.59_{(0.33)}$	$75.63_{(0.18)}$	$70.49_{(0.54)}$	$68.63_{(1.51)}$	$68.38_{(1.27)}$	$65.82_{(2.49)}$	$68.43_{(1.85)}$	$66.75_{(1.62)}$
Siamese	$80.40_{(0.29)}$	$65.00_{(11.69)}$	$71.10_{(1.68)}$	$51.43_{(2.83)}$	$59.96_{(3.56)}$	$50.00_{(0.19)}$	-	-
EGNN	$80.11_{(0.31)}$	$75.71_{(0.21)}$	$71.24_{(0.37)}$	$66.36_{(0.29)}$	$68.84_{(1.35)}$	$62.72_{(1.97)}$	$66.42_{(0.77)}$	$63.98_{(1.20)}$
IterRefLSTM	$81.10_{(0.10)}$	$80.97_{(0.06)}$	$69.63_{(0.16)}$	$71.73_{(0.06)}$	$49.56_{(2.32)}$	$48.54_{(1.48)}$	-	-
PAR	$82.13_{(0.26)}$	$\underline{80.02}_{(0.30)}$	$\underline{75.15}_{(0.35)}$	$\underline{72.33}_{(0.47)}$	$68.08_{(2.23)}$	$65.62_{(3.49)}$	$70.01_{(0.85)}$	$\underline{68.22}_{(1.34)}$
ADKF-IFT	$\underline{82.43}_{(0.60)}$	$77.94_{(0.91)}$	$67.72_{(1.21)}$	$58.69_{(1.44)}$	$\mathbf{98.18}_{(3.05)}$	$\underline{67.04}_{(4.86)}$	$\underline{72.07}_{(0.81)}$	$67.50_{(1.23)}$
PACIA	$\mathbf{84.25_{(0.31)}}$	$\mathbf{82.77_{(0.15)}}$	$\mathbf{82.40_{(0.26)}}$	$\mathbf{77.72_{(0.34)}}$	$\underline{72.58}_{(2.23)}$	$\mathbf{68.80}_{(4.01)}$	$\mathbf{72.38_{(0.96)}}$	$\mathbf{69.89_{(1.17)}}$

Table 1: Test ROC-AUC (%) obtained on MoleculeNet. The best results are bolded, second-best results are underlined.

4.3 Comparison with Existing Works

From the perspective of hypernetworks, the usage of hypernetworks for encoder is related to GNN-FiLM [Brockschmidt, 2020], which considers a GNN as main network. It builds hypernetworks with target node as input to generate parameters of FiLM layers, to equip different nodes with different aggregation functions in the GNN. What and how to adapt are similar to ours, but it is different that the input of our hypernetworks for encoder is $\mathcal{S}_\tau$ and how we encode a set of labeled graphs.

As for few-shot learning, some recent studies [Requeima *et al.*, 2019; Lin *et al.*, 2021] use hypernetworks to transform the support set into parameters that modulate the main network. The functionality of their hypernetworks is akin to that of our hypernetwork for the encoder. However, there is a distinct difference in the architecture of main networks: while their approaches employ convolutional neural networks and MLPs, our method uniquely modulates the message passing process of GNN. Furthermore, the application of hypernetworks for the predictor, aimed at adjusting the model architecture based on a query sample (without label information) and a set of labeled samples, has not yet been explored in the literature. A detailed comparison of PACIA w.r.t. existing few-shot MPP works is in Appendix.

5 Experiments

In this section, we evaluate the proposed PACIA[2] on few-shot MPP problems. We run all experiments with 10 random seeds, and report the mean and standard deviations (in the subscript bracket). Appendix provides more information of datasets, baselines, and implementation details.

5.1 Performance Comparison on MoleculeNet

We use Tox21 [National Center for Advancing Translational Sciences, 2017], SIDER [Kuhn *et al.*, 2016], MUV [Rohrer and Baumann, 2009] and ToxCast [Richard *et al.*, 2016] from MoleculeNet [Wu *et al.*, 2018], which are commonly used to evaluate the performance on few-shot MPP [Altae-Tran *et al.*, 2017; Wang *et al.*, 2021]. We adopt the public data split provided by [Wang *et al.*, 2021]. The support sets are balanced, each of them contains K labeled molecules per class, where $K = 1$ and $K = 10$ are considered. The performance is evaluated by ROC-AUC calculated on the query set of each meta-testing task and averaged across all meta-testing tasks.

We compare PACIA with the following baselines: 1) single-task method **GNN-ST** [Gilmer *et al.*, 2017]; 2) multi-task pretraining method **GNN-MT** [Corso *et al.*, 2020; Gilmer *et al.*, 2017]; 3) self-supervised pretraining method **MAT** [Maziarka *et al.*, 2020]; 4) meta-learning methods, including **Siamese** [Koch *et al.*, 2015], **ProtoNet** [Snell *et al.*, 2017], **MAML** [Finn *et al.*, 2017], **EGNN** [Kim *et al.*, 2019]; and 5) methods proposed for few-shot MPP, including **IterRefLSTM** [Altae-Tran *et al.*, 2017], **PAR** [Wang *et al.*, 2021] and **ADKF-IFT** [Chen *et al.*, 2022]. Note that MHNfs [Schimunek *et al.*, 2023] is not included as it uses additional reference molecules from external datasets, which leads to unfair comparison. GS-META [Zhuang *et al.*, 2023] has not been compared since that approach requires multiple properties of each molecule, which is not applicable when a molecule is only evaluated w.r.t. one property. Following earlier works [Guo *et al.*, 2021; Wang *et al.*, 2021], we use GIN [Xu *et al.*, 2019] as encoder, which is trained from scratch.

Table 1 shows the results. Results of Siamese and IterRefLSTM are copied from [Altae-Tran *et al.*, 2017] as their codes are unavailable, and their results on ToxCast are unknown. GNN-FiLM is a general GNN whose target is not few-shot MPP, which explains its bad performance. PACIA obtains the highest ROC-AUC scores on all cases except the 10-shot case on MUV, where ADKF-IFT outperforms the others by a large margin. This can be a special case where ADKF-IFT works well but may not be generalizable. Moreover, depending on the number of local-update steps of ADKF-IFT, PACIA is about 5 times faster than ADKF-IFT (both meta-training and inference time is about 1/5). In terms of average performance, PACIA significantly outperforms the second-best method ADKF-IFT by 3.25%. We also provide results obtained with a pretrained encoder in Appendix. Similar observations can be made: our PACIA with pretrained encoder (Pre-PACIA) performs the best, and its performance

[2]Code is available at https://github.com/LARS-research/PACIA.

gain is more significant when fewer labeled samples are provided. Further, results in Appendix shows the performance comparison between PACIA and a fine-tuned GNN with varying support set size. This shows that PACIA nicely achieves its goal: handling few-shot MPP problem in a parameter-efficient way.

5.2 Performance Comparison on FS-Mol

We also use FS-Mol [Stanley *et al.*, 2021], a new benchmark consisting of a large number of diverse tasks for model pre-training and a set of few-shot tasks with imbalanced classes. We adopt the public data split [Stanley *et al.*, 2021]. Each support set contains 64 labeled molecules, and can be imbalanced where the number of labeled molecules from active and inactive is not equal. All remaining molecules in the task form the query set. Testing tasks are divided into categories with support size 16 [Schimunek *et al.*, 2023], which is close to real-world scenario. The performance is evaluated by ΔAUPRC (change in area under the precision-recall curve) w.r.t. a random classifier [Stanley *et al.*, 2021], averaged across meta-testing tasks.

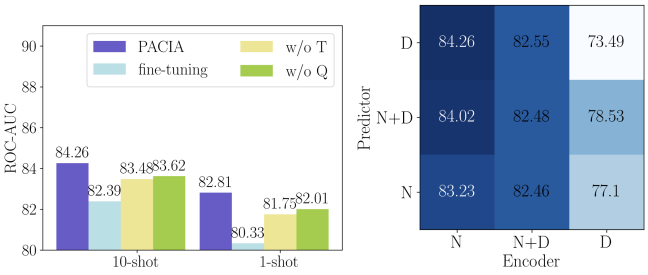
We use the same baselines that were applied to MoleculeNet. Table 2 shows the results. We find that PACIA performs the best. Besides, the time-efficiency of PACIA is much higher since its adaptation only needs a single forward pass. While IterRefLSTM and ADKF-IFT take multiple local-update steps, they are much slower to generalize.

Method	All [157]	Kinases [125]	Hydrolases [20]	Oxido-reductases[7]
GNN-ST	$2.9_{(0.4)}$	$2.7_{(0.4)}$	$4.0_{(1.8)}$	$2.0_{(1.6)}$
MAT	$5.2_{(0.5)}$	$4.3_{(0.5)}$	$9.5_{(1.9)}$	$6.2_{(2.4)}$
GNN-MT	$9.3_{(0.6)}$	$9.3_{(0.6)}$	$10.8_{(2.5)}$	$5.3_{(1.8)}$
MAML	$15.9_{(0.9)}$	$17.7_{(0.9)}$	$10.5_{(2.4)}$	$5.4_{(2.8)}$
PAR	$16.4_{(0.8)}$	$18.2_{(0.9)}$	$10.9_{(2.0)}$	$3.9_{(0.8)}$
ProtoNet	$20.7_{(0.8)}$	$21.5_{(0.9)}$	$20.9_{(3.0)}$	$9.5_{(2.9)}$
EGNN	$21.2_{(1.1)}$	$22.4_{(1.0)}$	$20.5_{(2.4)}$	$9.7_{(2.2)}$
Siamese	$22.3_{(1.0)}$	$24.1_{(1.0)}$	$17.8_{(2.6)}$	$8.2_{(2.5)}$
IterRefLSTM	$\underline{23.4}_{(1.0)}$	$\mathbf{25.1}_{(1.0)}$	$19.9_{(2.6)}$	$9.8_{(2.7)}$
ADKF-IFT	$\underline{23.4}_{(0.9)}$	$24.8_{(2.0)}$	$\underline{21.7}_{(1.7)}$	$\mathbf{10.6}_{(0.8)}$
PACIA	$\mathbf{23.6}_{(0.8)}$	$\mathbf{25.1}_{(1.6)}$	$\mathbf{21.9}_{(2.9)}$	$\mathbf{10.6}_{(1.0)}$

Table 2: Test ΔAUPRC (%) obtained on FS-Mol. Tasks are categorized by target protein type. The number of tasks per category is reported in brackets. The best results are bolded, second-best results are underlined.

5.3 Ablation Study

We consider various variants of PACIA, including (i) **fine-tuning**: using the same model structure and fine-tuning all parameters to adapt to each property without hypernetworks; (ii) **w/o T**: removing task-level adaptation, thus the GNN encoder will not be adapted by hypernetworks w.r.t. each property; and (iii) **w/o Q**: removing query-level adaptation, such that all molecules are processed by the same predictor.



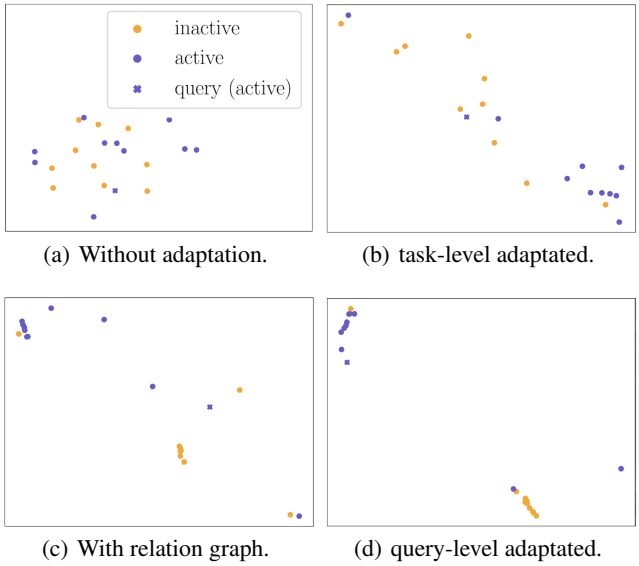
(a) Adaptation strategies. (b) Modulation functions.

Figure 2: Ablation study on 10-shot tasks of Tox21.

Figure 2(a) provides performance comparison on Tox21. As shown, the performance gain of PACIA over "w/o Q" shows the necessity of query-level adaptation. The gap between PACIA and "w/o T" indicates the effect of adapting the model to be task-specific. One can also notice that without query-level adaptation, "w/o Q" still obtains better performance than gradient-based baselines like PAR, which indicates the advantage of designing the amortization-based hypernetwork. The poor performance of "fine-tuning" is possibly because of the overfitting caused by updating all parameters with only a few samples. In sum, every component of PACIA is important for achieving good performance.

Now that the effectiveness of task-level and query-level adaptation are validated, we further investigate modulation functions, i.e, modulating node embedding (**N**), modulating propagation depth (**D**), modulating both (**ND**), for encoder and predictor. There are 3×3 combinations, whose performance is reported in Figure 2(b). We find that only modulating node embedding in encoder while only modulating propagation depth in predictor obtains the best performance. As the GIN encoder has highly non-linearity across layers, truncation would lead to non-explainability and somehow perturb the black-box. While the operation of relation graph in predictor updates node embedding in a linear way (5), adapting the propagation depth is harmonious with message passing process.

Figure 3 shows the t-SNE visualization [Van der Maaten and Hinton, 2008] of molecular representations learned on a 10-shot support set and a query molecule with ground-truth label "active" in task SR-p53 from Tox21. As shown, molecular representations obtained without adaptation (Figure 3(a)) are mixed up, since the encoder has not been adapted to the target property of the task. Molecular representations being processed by our property-adaptive encoder (Figure 3(b)) becomes more distinguishable, indicating that adapting molecular representation in task-level takes effect. Molecular representations in Figure 3(c) and Figure 3(d) form clear clusters as we encourage similar molecules to be connected during relation graph refinement by (5). The difference is that molecular representations in Figure 3(c) are refined by the best propagation depth number for all tasks in 10-shot case, while molecular representations in Figure 3(d) are refined by 4 depths of propagation which are selected for the specific query molecule. As shown, we can conclude that our molecular-adaptive refinement steps help better separate



(a) Without adaptation. (b) task-level adaptated.

(c) With relation graph. (d) query-level adaptated.

Figure 3: Molecular representation visualization for 10-shot case in task SR-p53 of Tox21.

molecules of different classes. Ablation study of configurations of hypernetworks is in Appendix.

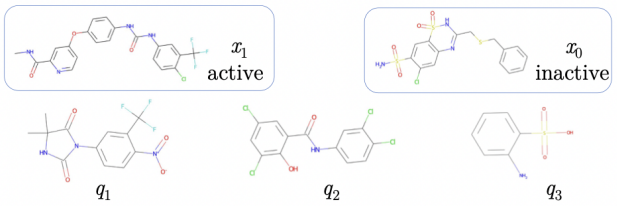
5.4 Study of Hierarchical Adaptation Mechanism

Task-Level Adaptation. In PACIA, parameter-efficient task-level adaptation is achieved by using hypernetworks to modulate the node embeddings during message passing. We compared this amortization-based adaptation with gradient-based adaptation in PAR which has similar main network with PACIA. We record their adaptation process, i.e., time required to process the support set and the test performance. Table 3 shows the results. PAR uses molecules in support set to take gradient steps, and updates all parameters in GNN. We record each of a maximum five steps, where we can find that it easily overfits as the testing ROC-AUC keeps dropping with more steps. The time consumption also grows. In contrast, PACIA processes molecules in support set by hypernetworks, which is much more efficient as only one single forward pass is needed. PACIA can obtain better performance due to the reduction of adaptive parameters, which also leads to better generalization and alleviates the risk of overfitting to a few shots. Table 3 and Figure 1(a) both indicate that the underlying overfitting problem can be mitigated by PACIA.

	PACIA	PAR				
# Total para.	3.28M	2.31M				
# Adaptive para.	3.00K	0.38M				
# Gradient steps	-	1	2	3	4	5
ROC-AUC (%)	84.26	82.07	81.85	80.32	79.09	77.25
Time (secs)	1.09	2.02	3.62	5.34	6.76	8.10

Table 3: Comparison of task-level adaptation approaches.

Query-Level Adaptation. Finally, we present a case study on query-level adaptation. More experimental results on validating the design of query-level adaptation are in Appendix. We use a 1-shot support set and 3 query molecules in task SR-p53 of Tox21. In Figure 4(a), x_1 and x_0 are support molecules with different labels, q_1, q_2 and q_3 are query molecules. As shown, classifying q_1 and q_3 is relatively easy and the propagation depth will be 1, while classifying q_2 is hard and requires 4 depth of propagation. Considering the shared substructures (function groups), q_1 and x_1 are visually similar, q_3 and x_0 are visually similar. While both x_1 and x_0 share substructures with q_2, it is hard to tell which of them is more similar to q_2. Figure 4(a) provides the cosine similarity based on the molecule representations generated by Pre-GNN , which confirms our observation: q_1 is much more similar to x_1, q_3 is much more similar to x_0, and q_2 shows large similarities with both samples. Intuitively, classifying q_1 and q_3 will be easier while q_2 will be hard. In the dynamic propagation of PACIA, we find different depths are taken: 1 for both q_1 and q_3 while 4 for q_2. PACIA achieves effective query-level adaptation by assigning more complex models for query molecules that are difficult to classify.



(a)

Query molecule	q_1		q_2		q_3	
Support molecule	x_1	x_0	x_1	x_0	x_1	x_0
Cosine similarity	0.183	-0.024	0.278	0.252	-0.031	0.129
Propagation depth	1		4		1	

(b)

Figure 4: Illustration of query-level adaptation. (a), Molecular graphs of support molecules x_1, x_0 and query molecules q_1, q_2, q_3. (b), Cosine similarities between query molecules and support molecules, and propagation depth taken to classify each query molecule.

6 Conclusion

We propose PACIA to handle few-shot MPP in a parameter-efficient manner. We investigate two key factors in few-shot molecular property prediction under the common encoder-predictor framework: adaptation-efficiency and query-level adaptation. Evidence shows that too much adaptive parameter would lead to overfitting, thus we design a parameter-efficient GNN adapter, which can modulate node embedding and propagation depth of message passing of GNN in a unified way. We also notice the importance of capturing query-level difference and therefore propose hierarchical adaptation mechanism, which is achieved by using a unified GNN adapter in both encoder and predictor. Empirical results show that PACIA achieves the best performance on both MoleculeNet and FS-Mol.

Acknowledgments

Q. Yao is supported by research fund of National Natural Science Foundation of China (No. 92270106), and Independent Research Plan of the Department of Electronic Engineering Department at Tsinghua University.

References

[Altae-Tran *et al.*, 2017] Han Altae-Tran, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. Low data drug discovery with one-shot learning. *ACS Central Science*, 3(4):283–293, 2017.

[Brockschmidt, 2020] Marc Brockschmidt. GNN-FiLM: Graph neural networks with feature-wise linear modulation. In *International Conference on Machine Learning*, pages 1144–1152, 2020.

[Chen *et al.*, 2022] Wenlin Chen, Austin Tripp, and José Miguel Hernández-Lobato. Meta-learning adaptive deep kernel gaussian processes for molecular property prediction. In *International Conference on Learning Representations*, 2022.

[Corso *et al.*, 2020] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. In *Advances in Neural Information Processing Systems*, pages 13260–13271, 2020.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

[Gilmer *et al.*, 2017] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272, 2017.

[Grant *et al.*, 2018] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical Bayes. In *International Conference on Learning Representations*, 2018.

[Guo *et al.*, 2021] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. Few-shot graph learning for molecular property prediction. In *The Web Conference*, pages 2559–2567, 2021.

[Ha *et al.*, 2017] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations*, 2017.

[Hu *et al.*, 2019] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2019.

[Kim *et al.*, 2019] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang 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.

[Koch *et al.*, 2015] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015.

[Kuhn *et al.*, 2016] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The SIDER database of drugs and side effects. *Nucleic Acids Research*, 44(D1):D1075–D1079, 2016.

[Li *et al.*, 2018] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *AAAI Conference on Artificial Intelligence*, pages 3546–3553, 2018.

[Lin *et al.*, 2021] Xixun Lin, Jia Wu, Chuan Zhou, Shirui Pan, Yanan Cao, and Bin Wang. Task-adaptive neural process for user cold-start recommendation. In *The Web Conference*, pages 1306–1316, 2021.

[Ma *et al.*, 2015] Junshui Ma, Robert P Sheridan, Andy Liaw, George E Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.

[Maziarka *et al.*, 2020] Lukasz Maziarka, Tomasz Danel, Slawomir Mucha, Krzysztof Rataj, Jacek Tabor, and Stanislaw Jastrzebski. Molecule attention transformer. *arXiv preprint arXiv:2002.08264*, 2020.

[Nachmani and Wolf, 2020] Eliya Nachmani and Lior Wolf. Molecule property prediction and classification with graph hypernetworks. *arXiv preprint arXiv:2002.00240*, 2020.

[National Center for Advancing Translational Sciences, 2017] National Center for Advancing Translational Sciences. Tox21 challenge. http://tripod.nih.gov/tox21/challenge/, 2017. Accessed: 2016-11-06.

[Przewiezlikowski *et al.*, 2022] Marcin Przewiezlikowski, P Przybysz, Jacek Tabor, Maciej Zieba, and Przemyslaw Spurek. HyperMAML: Few-shot adaptation of deep models with hypernetworks. *arXiv preprint arXiv:2205.15745*, 2022.

[Rajeswaran *et al.*, 2019] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pages 113–124, 2019.

[Requeima *et al.*, 2019] James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, pages 7957–7968, 2019.

[Richard *et al.*, 2016] Ann M Richard, Richard S Judson, Keith A Houck, Christopher M Grulke, Patra Volarath, Inthirany Thillainadarajah, Chihae Yang, James Rathman, Matthew T Martin, John F Wambaugh, et al. ToxCast chemical landscape: Paving the road to 21st century toxicology. *Chemical Research in Toxicology*, 29(8):1225–1251, 2016.

[Rogers and Hahn, 2010] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010.

[Rohrer and Baumann, 2009] Sebastian G Rohrer and Knut 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.

[Schimunek *et al.*, 2023] Johannes Schimunek, Philipp Seidl, Lukas Friedrich, Daniel Kuhn, Friedrich Rippmann, Sepp Hochreiter, and Günter Klambauer. Context-enriched molecule representations improve few-shot drug discovery. *arXiv preprint arXiv:2305.09481*, 2023.

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4080–4090, 2017.

[Stanley *et al.*, 2021] Megan Stanley, John F Bronskill, Krzysztof Maziarz, Hubert Misztela, Jessica Lanini, Marwin Segler, Nadine Schneider, and Marc Brockschmidt. FS-Mol: A few-shot learning dataset of molecules. In *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[Unterthiner *et al.*, 2014] Thomas Unterthiner, Andreas Mayr, Günter Klambauer, Marvin Steijaert, Jörg K Wegner, Hugo Ceulemans, and Sepp Hochreiter. Deep learning as an opportunity in virtual screening. In *NIPS Deep Learning Workshop*, volume 27, pages 1–9, 2014.

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

[Vinyals *et al.*, 2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[Wang *et al.*, 2020] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):1–34, 2020.

[Wang *et al.*, 2021] Yaqing Wang, Abulikemu Abuduweili, Quanming Yao, and Dejing Dou. Property-aware relation networks for few-shot molecular property prediction. In *Advances in Neural Information Processing Systems*, pages 17441–17454, 2021.

[Waring *et al.*, 2015] Michael J Waring, John Arrowsmith, Andrew R Leach, Paul D Leeson, Sam Mandrell, Robert M Owen, Garry Pairaudeau, William D Pennie, Stephen D Pickett, Jibo Wang, et al. An analysis of the attrition of drug candidates from four major pharmaceutical companies. *Nature Reviews Drug discovery*, 14(7):475–486, 2015.

[Wu *et al.*, 2018] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018.

[Xiong *et al.*, 2019] Zhaoping Xiong, Dingyan Wang, Xiaohong Liu, Feisheng Zhong, Xiaozhe Wan, Xutong Li, Zhaojun Li, Xiaomin Luo, Kaixian Chen, Hualiang 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.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[Yang *et al.*, 2019] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388, 2019.

[Yin *et al.*, 2020] Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. In *International Conference on Learning Representations*, 2020.

[Zhuang *et al.*, 2023] Xiang Zhuang, Qiang Zhang, Bin Wu, Keyan Ding, Yin Fang, and Huajun Chen. Graph sampling-based meta-learning for molecular property prediction. *arXiv preprint arXiv:2306.16780*, 2023.