Robust Molecular Property Prediction via Densifying Scarce Labeled Data

Jina Kim [* 1]   **Jeffrey Willette** [* 1]   **Bruno Andreis** [* 1]   **Sung Ju Hwang** [1 2]

Abstract

A widely recognized limitation of molecular prediction models is their reliance on structures observed in the training data, resulting in poor generalization to out-of-distribution compounds. Yet in drug discovery, the compounds most critical for advancing research often lie beyond the training set, making the bias toward the training data particularly problematic. This mismatch introduces substantial covariate shift, under which standard deep learning models produce unstable and inaccurate predictions. Furthermore, the scarcity of labeled data—stemming from the onerous and costly nature of experimental validation—further exacerbates the difficulty of achieving reliable generalization. To address these limitations, we propose a novel meta-learning-based approach that leverages unlabeled data to interpolate between in-distribution (ID) and out-of-distribution (OOD) data, enabling the model to meta-learn how to generalize beyond the training distribution. We demonstrate significant performance gains on challenging real-world datasets with substantial covariate shift, supported by t-SNE visualizations highlighting our interpolation method.

1. Introduction

Molecular property prediction plays a central role in drug discovery pipelines, enabling researchers to prioritize compounds for costly and time-consuming experimental validation. Accurate computational models have the potential to dramatically accelerate early-stage discovery by predicting critical attributes such as bioactivity, toxicity, and solubility

[*]Equal contribution [1]Korea Advanced Institute of Science and Technology (KAIST), South Korea [2]Deepauto.ai, South Korea. Correspondence to: Jina Kim <jinakim@kaist.ac.kr>, Jeffrey Willette <jwillette@kaist.ac.kr>, Bruno Andreis <andries@kaist.ac.kr>, Sung Ju Hwang <sungju.hwang@kaist.ac.kr>.
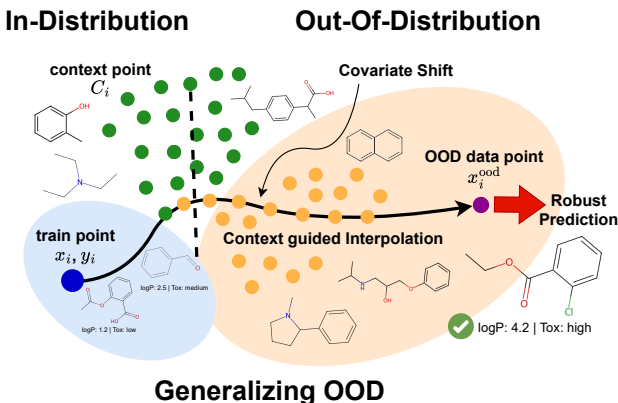
Figure 1. **Concept.** We densify the train dataset using external unlabeled data (context point) for robust generalization across covariate shift. Notation details are provided in Section 4.

before synthesis (Schneider, 2018; Vamathevan et al., 2019). However, building reliable predictive models generalizing to novel, unseen compounds remains a fundamental challenge.

Standard molecular property prediction models tend to rely heavily on patterns observed within the training distribution, resulting in poor generalization to out-of-distribution compounds (Klarner et al., 2023; Ovadia et al., 2019; Koh et al., 2021). In drug discovery, this limitation is particularly problematic, since the compounds most crucial for advancing research often lie far beyond the chemical spaces represented during training (Lee et al., 2023). The resulting covariate shift introduces significant obstacles to reliable prediction, with models frequently producing unstable outputs when extrapolating to new regions of chemical space. Further compounding these challenges, experimental validation of molecular properties is both costly and resource-intensive, leading to a scarcity of labeled data and increasing reliance on computational exploration (Altae-Tran et al., 2017). Also, available labeled data is typically concentrated in narrow regions of chemical space, introducing bias that hampers generalization to unseen compounds (Klarner et al., 2023).

While vast collections of unlabeled molecular structures are readily available (Sterling & Irwin, 2015; Kim et al., 2021), offering rich information about the structure of chemical space, existing methods often fail to fully exploit this resource to improve generalization (Klarner et al.,

2023). Therefore, we propose a novel meta-learning based method that leverages unlabeled data to densify the scarce train dataset and guide the model toward sensible behavior in unexplored regions of chemical space. Our code can be found at https://github.com/JinA0218/drugood-densify.

2. Methodology

Preliminaries. We consider the problem of molecular property prediction under covariate shift. Given a small labeled dataset $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^n$ and abundant unlabeled molecules $\mathcal{D}_{\text{unlabeled}} = \{x_j\}_{j=1}^m$, the goal is to learn a predictive model $f : \mathcal{X} \to \mathcal{Y}$ that reliably generalizes to a distributionally shifted test set $\mathcal{D}_{\text{test}}$.

Scarce Data Densification with Unlabeled Data To address this, we propose a meta-learning based framework that interpolates the training distribution $\mathcal{D}_{\text{train}}$ with an exogenous distribution $\mathcal{D}_{\text{unlabeled}}$. Our objective is to leverage the cheaper and more abundant distribution $\mathcal{D}_{\text{unlabeled}}$ to densify the scarce labeled distribution $\mathcal{D}_{\text{train}}$ in a way that encourages the model to generalize robustly under covariate shift, particularly in out-of-distribution scenarios where we have no label information and therefore high uncertainty. For this, we utilize subsets of $\mathcal{D}_{\text{unlabeled}}$ as $\mathcal{D}_{\text{context}}$ and $\mathcal{D}_{\text{mvalid}}$, where $\mathcal{D}_{\text{context}}$ is a domain-informed external task distribution for interpolating to $\mathcal{D}_{\text{train}}$, and $\mathcal{D}_{\text{mvalid}}$ is a meta-validation set used to guide the interpolation function. Inspired by (Lee et al., 2024), we introduce a permutation invariant learnable set function (Zaheer et al., 2017; Lee et al., 2019) μ_λ as a mixer (interpolator), which learns to mix each point from $x_i \sim \mathcal{D}_{\text{train}}$ with the context points $\{c_{ij}\}_{j=1}^{m_i}$ in a way that densifies $\mathcal{D}_{\text{train}}$, where

$$(x_i, y_i) \sim \mathcal{D}_{\text{train}}, \quad \{c_{ij}\}_{j=1}^{m_i} \sim \mathcal{D}_{\text{context}}, \quad i \in \{1, \ldots, B\}$$

and B denotes the minibatch size, $m_i \sim \mathcal{U}_{\text{int}}(0, M)$ where M controls the maximum number of context samples drawn from $\mathcal{D}_{\text{context}}$ for each minibatch. Given a feature dimension D, for each i, the input consists of, $x_i \in \mathbb{R}^{B \times 1 \times D}$ and $\{c_{ij}\}_{j=1}^{m_i} \in \mathbb{R}^{B \times 1 \times D}$, where the set $\{c_{ij}\}_{j=1}^{m_i}$ can be organized into a tensor $C_i \in \mathbb{R}^{B \times m_i \times D}$.

Overall, our model has two main components: (1) a meta-learner f_{θ_l}, which is a standard MLP at the lth layer, that maps input data $x_i^{(l-1)} \in \mathbb{R}^{B \times 1 \times D}$ to the feature space of the $(l+1)^{\text{th}}$ layer, producing $x_i^{(l)} = f_{\theta_l}(x_i^{(l-1)})$, and (2) a learnable set function μ_λ which mixes $x_i^{(l_{\text{mix}})}$ and $C_i^{(l_{\text{mix}})}$ as a set and outputs a single pooled representation $\tilde{x}_i^{(l_{\text{mix}})} = \mu_\lambda(\{x_i^{(l_{\text{mix}})}, C_i^{(l_{\text{mix}})}\}) \in \mathbb{R}^{B \times 1 \times H}$, where H is the hidden dimension and l_{mix} is the layer where the mixing happens. The full model structure with L layers can be expressed as

$$\hat{f}_{\theta,\lambda} := f_{\theta_L} \circ \cdots \circ f_{\theta_{l_{\text{mix}}+1}} \circ \mu_\lambda \circ f_{\theta_{l_{\text{mix}}-1}} \circ \cdots \circ f_{\theta_1}.$$

We utilize bilevel optimization for training meta-learner f_{θ_l}, and treat the set function parameter μ_λ as a hyperparameter to be optimized in the outer loop (Lorraine et al., 2019). As shown in Table 2 (w/o bilevel optimization), simply optimizing the meta-learner parameters θ and the set function parameters λ jointly can lead to overfitting to the task distribution and harms test-time generalization. Following the setting of (Lorraine et al., 2019), during training, we only update the parameter θ in the inner loop and we only update the parameter λ in the outer loop (see Figure 9b for the detailed model structure of the bilevel optimization).

In the inner loop, the model accepts $x_i \in \mathbb{R}^{B \times 1 \times H}$ and $C_i \in \mathbb{R}^{B \times m_i \times H}$ and the set encoder μ_λ mixes $\{x_i^{(l_{\text{mix}})}, C_i^{(l_{\text{mix}})}\}$ and outputs $\tilde{x}_i^{(l_{\text{mix}})} \in \mathbb{R}^{B \times 1 \times H}$. Since C_i is used to introduce a domain-informed external context to densify $\mathcal{D}_{\text{train}}$, we utilize the original label y_i from $\mathcal{D}_{\text{train}}$ to train the task learner parameters f_{θ_l}, with the mixed $\tilde{x}_i^{(l_{\text{mix}})}$.

In the outer loop, we train the set encoder using hypergradient (Lorraine et al., 2019), which aims to minimize a meta-validation loss $L_V(\lambda, \theta^*(\lambda))$, where the model parameters $\theta^*(\lambda)$ are the solution to the inner training objective $\theta^*(\lambda) = \arg\min_\theta L_T(\theta, \lambda)$ and $L_T(\theta, \lambda)$ denotes the *training loss* computed on a labeled dataset $\mathcal{D}_{\text{train}}$, potentially regularized or influenced by the hyperparameters λ. This inner loss L_T governs the optimization of model parameters θ, while L_V evaluates generalization performance on a meta-validation set $\mathcal{D}_{\text{mvalid}}$, and guides the update of λ. In order for the hypergradient to train the set encoder μ_λ in a way that guides the overall model $\hat{f}_{\theta,\lambda}$ toward robustness under covariate shift, we construct $\mathcal{D}_{\text{mvalid}}$ as

$$\{x_{i,k}^{(\text{mvalid})}\}_{k=1}^K \sim \mathcal{D}_{\text{unlabeled}}, \quad y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0, 1),$$

where K is a hyperparameter of the samples drawn from $\mathcal{D}_{\text{unlabeled}}$ for each minibatch. In Table 2, we empirically show that using a random $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0, 1)$ achieves better or comparable performance to using a labeled $y_{i,k}^{(\text{mvalid})} \sim \mathcal{D}_{\text{oracle}}$ corresponding to the real label for $x_{i,k}^{(\text{mvalid})}$ from the unlabeled dataset which has been labeled by an oracle. This setting aligns with the role of L_V in guiding the set function μ_λ to generalize to out-of-distribution data, as sampling pseudo-labels from $\mathcal{N}(0, 1)$ introduces controlled label noise that regularizes the model. The outer loop loss L_V can then be expressed as a function of two variables $L_V(\lambda, \theta^*(\lambda))$, and therefore, the hypergradient is given by,

$$\frac{dL_V}{d\lambda} = \frac{\partial L_V}{\partial \lambda} + \frac{\partial L_V}{\partial \theta} \cdot \frac{d\theta^*(\lambda)}{d\lambda}$$

and Implicit Function Theorem (IFT) is applied to compute $\frac{d\theta^*(\lambda)}{d\lambda}$ without differentiating through the entire optimization trajectory:

$$\frac{d\theta^*(\lambda)}{d\lambda} = -\left(\frac{\partial^2 L_T}{\partial \theta^2}\right)^{-1} \cdot \frac{\partial^2 L_T}{\partial \theta \partial \lambda}.$$

Table 1. Merck Molecular Activity Challenge. We report performance (MSE ↓) on datasets with the large test-time covariate shift (HIVPROT, DPP4, NK1). All entries show the mean and standard errors calculated over 10 training runs. The best performing models are highlighted in bold. 'outlier exposure' indicates whether outlier exposure is enabled, and 'bilevel' denotes the use of bilevel optimization.

Model	HIVPROT		DPP4		NK1	
	count vector	bit vector	count vector	bit vector	count vector	bit vector
L_1-Regression	$1.137_{\pm.000}$	$0.714_{\pm.000}$	$1.611_{\pm.000}$	$1.130_{\pm.000}$	$0.482_{\pm.000}$	$0.442_{\pm.000}$
L_2-Regression	$0.999_{\pm.000}$	$0.723_{\pm.000}$	$1.495_{\pm.000}$	$1.143_{\pm.000}$	$0.498_{\pm.000}$	$0.436_{\pm.000}$
Random Forest	$0.815_{\pm.009}$	$0.834_{\pm.010}$	$1.473_{\pm.008}$	$1.461_{\pm.012}$	$0.458_{\pm.002}$	$0.438_{\pm.002}$
MLP	$0.768_{\pm.014}$	$2.118_{\pm.015}$	$1.393_{\pm.024}$	$1.094_{\pm.029}$	$0.443_{\pm.007}$	$0.399_{\pm.006}$
Mixup	$0.764_{\pm0.008}$	$0.691_{\pm0.022}$	$1.439_{\pm0.021}$	$1.212_{\pm0.012}$	$0.481_{\pm0.002}$	$0.479_{\pm0.003}$
Mixup (w/ outlier exposure)	$0.748_{\pm0.01}$	$0.677_{\pm0.015}$	$1.384_{\pm0.012}$	$1.224_{\pm0.016}$	$0.442_{\pm0.005}$	$0.443_{\pm0.005}$
Manifold Mixup	$0.88_{\pm0.023}$	$0.898_{\pm0.022}$	$1.414_{\pm0.021}$	$1.367_{\pm0.04}$	$0.432_{\pm0.005}$	$0.499_{\pm0.013}$
Manifold Mixup (w/ bilevel)	$0.484_{\pm0.011}$	$0.804_{\pm0.086}$	$1.19_{\pm0.05}$	$1.217_{\pm0.068}$	$0.43_{\pm0.013}$	$0.536_{\pm0.039}$
Q-SAVI	$0.682_{\pm.019}$	$0.664_{\pm.028}$	$1.332_{\pm.017}$	$1.028_{\pm.027}$	$0.436_{\pm.007}$	$0.387_{\pm.012}$
Ours (Deepsets)	0.555 ± 0.096	$\mathbf{0.364 \pm 0.018}$	$\mathbf{0.984 \pm 0.018}$	$\mathbf{0.963 \pm 0.017}$	0.455 ± 0.016	$\mathbf{0.376 \pm 0.008}$
Ours (Set Trans.)	$\mathbf{0.39 \pm 0.011}$	0.726 ± 0.159	1.121 ± 0.037	0.986 ± 0.021	$\mathbf{0.429 \pm 0.01}$	0.397 ± 0.015

Luckily, the inverse Hessian vector product can be approximated and computed efficiently using automatic differentiation and Neuman series iterations (Lorraine et al., 2019). In Section 4, we display how the model $\hat{f}_{\theta,\lambda}$ effectively uses $\mathcal{D}_{\text{context}}$ to densify the data distribution around $\mathcal{D}_{\text{train}}$.

As the mixing of $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{context}}$ is meant to densify the sparse training data, we only perform the mixing at train time. At test time, the set function μ_λ receives an input as a singleton set and performs no mixing with any exogeneous distribution (see Figure 9a for train and test-time diagram).

3. Experiments

We evaluate on the Merck Molecular Activity Challenge (Merck) dataset (Kulkarni et al., 2012), a benchmark for molecular property prediction. The dataset contains bioactivity measurements for therapeutic targets, with molecules represented by high-dimensional chemical descriptors. It comprises 15 distinct datasets with a regression target predicting a molecular activity measurement in (possibly) different units, enabling evaluation across multiple distribution shifts. The Merck dataset reflects real-world drug discovery scenarios where promising candidates lie outside known chemical spaces, making generalization challenging. Molecular inputs are encoded either as *bit vectors*, indicating the presence or absence of substructures, or *count vectors*, which reflect the frequency of substructures—providing varying granularity of chemical information.

Following the prior work (Klarner et al., 2023), for $\mathcal{D}_{\text{train}}$, we focus on the three subsets (HIVPORT, DPP4, NK1) that exhibit the greatest distributional shift from and treat the remaining 12 datasets as $\mathcal{D}_{\text{unlabeled}}$ and exclude their labels from the training process. For fair comparison with baselines, we randomly select some portion (K for each minibatch) of $\mathcal{D}_{\text{unlabeled}}$ as $\mathcal{D}_{\text{mvalid}}$ and sample $\mathcal{D}_{\text{context}}$ from $\mathcal{D}_{\text{unlabeled}}$ (m_i for each minibatch). As $\mathcal{D}_{\text{mvalid}}$ is randomly labeled, we ensure that all baselines see the same amount of labeled data.

In addition to the baselines presented in (Klarner et al., 2023), we further implement and evaluate two related state-of-the-art interpolation methods—Mixup (Zhang et al., 2017) and Manifold Mixup (Verma et al., 2019)—to empirically demonstrate the capability of our set function μ_λ. As mixup performs the mixing procedure in the input space, there are no parameters to train for bilevel optimization. We therefore report two variants of mixup (plain and outlier exposure) where outlier exposure is exposed to the same $\mathcal{D}_{\text{mvalid}}$ with random labels as described above. Manifold mixup performs the mixup procedure in the feature space between layers, therefore we can straightforwardly apply our bilevel optimization procedure to manifold mixup. Therefore, we report two variants of manifold mixup (plain and bilevel optimization).

For Mixup, we set $l_{\text{mix}} = 1$ so we mix $\{x_i^{(l_1)}, C_i^{(l_1)}\}$ and for Manifold Mixup, we set $l_{\text{mix}} > 1$, so we mix $\{x_i^{(l_{\text{mix}})}, C_i^{(l_{\text{mix}})}\}$ after passing at least one layer f_{θ_l} (For further implementation details, please see Appendix A.1). When evaluating with bilevel optimization, we follow our setting and update θ and λ separately in each loop using hypergrad (Lorraine et al., 2019), and for the setting without bilevel optimization, we jointly update θ and λ using a single optimizer. As shown in Table 1, when either leveraging Deepsets (Zaheer et al., 2017) or Set Transformer (Lee et al., 2019) as the mixer μ_λ, our method significantly improves MSE compared to the baselines, demonstrating greater robustness of our model toward covariate shift.

Furthermore, we present a series of ablation studies in Table 2 to analyze the contributions of each component of our method (Implementation details can be found in Appendix A.2). Specifically, to validate the effectiveness of $\mathcal{D}_{\text{context}}$ and our bilevel optimization (Lorraine et al., 2019) under $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$, we compare our approach against the following variants: (1) using $\mathcal{D}_{\text{context}}$ without bilevel optimization, (2) excluding both $\mathcal{D}_{\text{context}}$ and bilevel optimization, and (3) using the real label for $y_{i,k}^{(\text{mvalid})}$. The results indicate that, in order to achieve robust generalization across

Table 2. Ablation study of **Ours**. 'ctx' indicates the use of context points, while 'bilevel' denotes the application of bilevel optimization. For $y^{(\text{mvalid})}$, 'rand' refers to sampling from a standard normal distribution, whereas 'real' refers to sampling from the oracle distribution.

Model	w/ ctx.	w/ bilevel	$y^{(\text{mvalid})}$	HIVPROT		DPP4		NK1	
				count vector	bit vector	count vector	bit vector	count vector	bit vector
Ours (Deepsets)	✓	✗	rand	1.147 ± 0.055	1.012 ± 0.025	1.342 ± 0.019	1.344 ± 0.039	0.45 ± 0.01	0.494 ± 0.006
Ours (Set Trans.)	✓	✗	rand	1.193 ± 0.038	1.008 ± 0.03	1.34 ± 0.026	1.359 ± 0.013	0.482 ± 0.002	0.487 ± 0.009
Ours (MLP)	✗	✗	rand	1.043 ± 0.013	1.049 ± 0.013	1.42 ± 0.009	1.269 ± 0.017	0.435 ± 0.002	0.466 ± 0.004
Ours (Deepsets)	✓	✓	real	0.506 ± 0.061	0.381 ± 0.018	$\mathbf{0.970 \pm 0.012}$	0.964 ± 0.024	0.495 ± 0.037	0.395 ± 0.014
Ours (Set Trans.)	✓	✓	real	0.401 ± 0.02	0.811 ± 0.111	1.123 ± 0.024	0.975 ± 0.016	0.449 ± 0.021	0.401 ± 0.017
Ours (Deepsets)	✓	✓	rand	0.555 ± 0.096	$\mathbf{0.364 \pm 0.018}$	0.984 ± 0.018	$\mathbf{0.963 \pm 0.017}$	0.455 ± 0.016	$\mathbf{0.376 \pm 0.008}$
Ours (Set Trans.)	✓	✓	rand	$\mathbf{0.39 \pm 0.011}$	0.726 ± 0.159	1.121 ± 0.037	0.986 ± 0.021	$\mathbf{0.429 \pm 0.01}$	0.397 ± 0.015

(a) **Ours**　　　　　　(b) **Mixup (w/ outlier exposure)**　　　　　　(c) **Manifold Mixup (w/ bilevel optim.)**
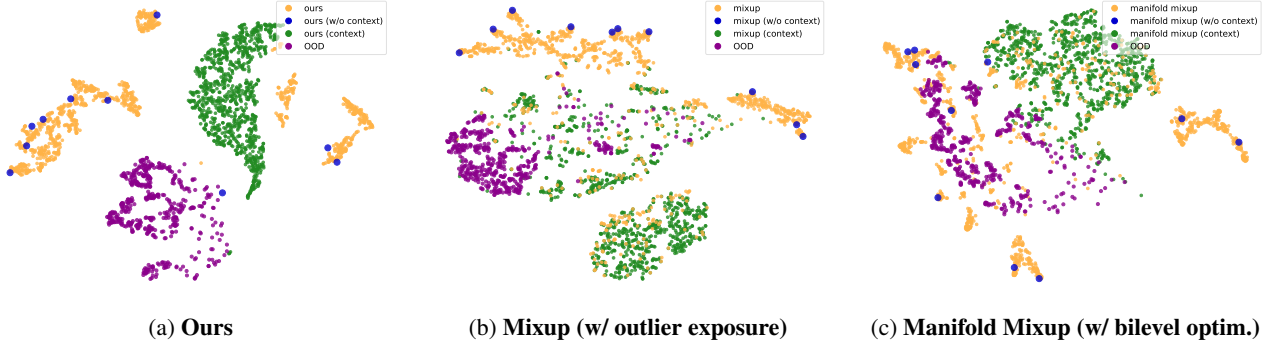
Figure 2. **t-SNE visualization of DPP4 (bit) dataset from the penultimate layer across different methods.** All models were trained on $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{context}}$, and $\mathcal{D}_{\text{mvalid}}$. At test time, we evaluate each model on four input variants (orange, blue, green, purple) to analyze how the model utilizes $\mathcal{D}_{\text{context}}$ to achieve robustness under covariate shift and how it behaves on out-of-distribution (OOD) data.

all datasets, it is crucial to leverage bilevel optimization with $\mathcal{D}_{\text{context}}$. Especially, the dataset HIVPROT, which has the highest degree of label and covariate shift compared to others (Klarner et al., 2023) (Table 3), showed a detrimental performance drop (62.6%–67.3% for the count vector, and 63.8%–65.3% for the bit vector) in settings (1,2). Moreover, using $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$ in L_V for training the set function μ_λ resulted in comparable or better performance than using the true label associated with $x_{i,k}^{(\text{mvalid})}$, validating our choice of L_V and its contribution to improved robustness under covariate shift.

4. Analysis

To visually assess how our model leverages $\mathcal{D}_{\text{context}}$ to densify $\mathcal{D}_{\text{train}}$, and to compare model behaviors on out-of-distribution (OOD) inputs, we visualize the penultimate-layer embeddings $\mathbf{z}_i = x_i^{(L-1)}$, using t-SNE (Van der Maaten & Hinton, 2008) across three models that interpolate $\mathcal{D}_{\text{train}}$ with $\mathcal{D}_{\text{context}}$: Ours, Mixup (w/ outlier exposure), and Manifold Mixup (w/ bilevel optim.), all trained on the DPP4 bit-vector dataset. We randomly sample $x_i, y_i \sim \mathcal{D}_{\text{train}}$, $C_i = \{c_{ij}\}_{j=1}^{100} \sim \mathcal{D}_{\text{context}}$, $x_i^{\text{ood}} \sim \mathcal{D}_{\text{ood}}$, where $\mathcal{D}_{\text{ood}}$ signifies an out-of-distribution dataset which was never seen during training. In Figure 2, each color indicates a penultimate-layer embedding $\mathbf{z} \in \mathbf{Z}$ under different input configurations. We denote the embeddings as follows: $\mathbf{Z}_{\text{joint}}$ for **orange** ($\{x_i, C_i\}$), $\mathbf{Z}_{\text{input}}$ for **blue** (x_i alone), $\mathbf{Z}_{\text{context}}$ for **green** (C_i alone), and $\mathbf{Z}_{\text{ood}}$ for **purple** (OOD samples x_i^{ood}). Additional t-SNE results and details are in Appendix B.

Effect of interpolating $\mathcal{D}_{\text{train}}$ with $\mathcal{D}_{\text{context}}$. In Figure 2a, we can see that our model successfully uses C_i to densify each train input x_i, based on the fact that $\mathbf{Z}_{\text{joint}}$ is concentrated in the vicinity of $\mathbf{Z}_{\text{input}}$ and is located in a completely separate cluster from both $\mathbf{Z}_{\text{context}}$ and $\mathbf{Z}_{\text{ood}}$. On the other hand, as shown in Figures 2b and 2c, both Mixup (w/ outlier exposure) and Mandifold Mixup (w/ bilevel optim.) show extensive overlap between $\mathbf{Z}_{\text{joint}}$, $\mathbf{Z}_{\text{context}}$, and $\mathbf{Z}_{\text{ood}}$.

Unlike Mixup (w/ outlier exposure) and Manifold Mixup (w/ bilevel optimization), our method shows much clearer separation between data distributions. This shows that our model effectively distinguishes seen from unseen distributions while learning structured, robust latent representations that enhance generalization under covariate shift, consistent with the results in Table 1.

5. Conclusion

To overcome the challenge of test-time covariate shift, we propose a novel meta-learning based method that densifies the training distribution using domain-informed unlabeled datasets via a learnable set function. During training, we utilize bilevel optimization which leverages noisy unlabeled data to guide our model towards robustness under covariate shift. We validate our method on challenging real-world molecular property prediction with large covariate shifts, and visually demonstrate the effectiveness of interpolation method which shows well defined separation between data distributions. Our code is available at https://github.com/JinA0218/drugood-densify.

References

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

Defazio, A., Yang, X. A., Mehta, H., Mishchenko, K., Khaled, A., and Cutkosky, A. The road less scheduled, 2024. URL https://arxiv.org/abs/2405.15682.

Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B. A., Thiessen, P. A., Yu, B., et al. Pubchem in 2021: new data content and improved web interfaces. *Nucleic acids research*, 49(D1):D1388–D1395, 2021.

Klarner, L., Rudner, T. G. J., Reutlinger, M., Schindler, T., Morris, G. M., Deane, C., and Teh, Y. W. Drug discovery under covariate shift with domain-informed prior distributions over functions, 2023. URL https://arxiv.org/abs/2307.15073.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B. A., Haque, I. S., Beery, S., Leskovec, J., Kundaje, A., Pierson, E., Levine, S., Finn, C., and Liang, P. Wilds: A benchmark of in-the-wild distribution shifts, 2021. URL https://arxiv.org/abs/2012.07421.

Kulkarni, A., BobS, joycenv, and Merck_GSC. Merck molecular activity challenge. https://kaggle.com/competitions/MerckActivity, 2012. Kaggle.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Lee, S., Jo, J., and Hwang, S. J. Exploring chemical space with score-based out-of-distribution generation, 2023. URL https://arxiv.org/abs/2206.07632.

Lee, S., Andreis, B., Kawaguchi, K., Lee, J., and Hwang, S. J. Set-based meta-interpolation for few-task meta-learning, 2024. URL https://arxiv.org/abs/2205.09990.

Lorraine, J., Vicol, P., and Duvenaud, D. Optimizing millions of hyperparameters by implicit differentiation, 2019. URL https://arxiv.org/abs/1911.02590.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift, 2019. URL https://arxiv.org/abs/1906.02530.

Schneider, G. Automating drug discovery. *Nature reviews drug discovery*, 17(2):97–113, 2018.

Sterling, T. and Irwin, J. J. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

Vamathevan, J., Clark, D., Czodrowski, P., Dunham, I., Ferran, E., Lee, G., Li, B., Madabhushi, A., Shah, P., Spitzer, M., et al. Applications of machine learning in drug discovery and development. *Nature reviews Drug discovery*, 18(6):463–477, 2019.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Courville, A., Lopez-Paz, D., and Bengio, Y. Manifold mixup: Better representations by interpolating hidden states, 2019. URL https://arxiv.org/abs/1806.05236.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL http://arxiv.org/abs/1710.09412.

A. Experimental Details

All code and experiments are publicly available at https://github.com/JinA0218/drugood-densify.

A.1. Implementation details of the baselines (Table 1)

In Table 1, we report the performance of L_1-Regression, L_2-Regression, Random Forest, MLP, and Q-SAVI by referring to the results from (Klarner et al., 2023). For Mixup (plain and with outlier exposure) and Manifold Mixup (plain and with bilevel optimization), we replace only our set function μ_λ with the linear interpolation μ_α^{lin} and reduction function, defined as

$$\tilde{x}_i^{(l_{\text{mix}})} = f_{\text{reduce}}\left(\alpha x_i + (1-\alpha)C_i\right), \quad \alpha \sim \mathcal{U}(0,1), \ \alpha \in \mathbb{R}^{B \times 1 \times 1}, \ f_{\text{reduce}} \in \{\text{mean}, \text{sum}, \text{max}\},$$

while keeping all other components unchanged. We apply the function f_{reduce} since $C_i \in \mathbb{R}^{B \times m_i \times 1}$, and therefore a final reduction must be applied in order to reduce the set dimension m_i to 1. We us $\max$ for f_{reduce} in Table 1 as a sensible default.

Mixup and Manifold Mixup. For the plain Mixup and Manifold Mixup (without bilevel or outlier exposure), we perform the mixing only utilizing $(x_i, y_i) \sim \mathcal{D}_{\text{train}}$ and $\{c_{ij}\}_{j=1}^{m_i} \sim \mathcal{D}_{\text{context}}$ without using D_{mvalid}, without any outlier exposure. Since this impelementation doesn't include bilevel optimization, we train each Mixup and Manifold Mixup with a single optimizer.

Mixup (w/ outlier exposure). To compare Mixup with outlier exposure, we implemented a variant of Mixup which is trained with $(x_i, y_i) \sim \mathcal{D}_{\text{train}}$, $\{c_{ij}\}_{j=1}^{m_i} \sim \mathcal{D}_{\text{context}}$ and $\{x_{i,k}^{(\text{mvalid})}\}_{k=1}^K \sim \mathcal{D}_{\text{unlabeled}}$, $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$. As explained in Section 3, since it peforms the mixing procedure in the input space there are no parameters to train, it is impossible to apply bilevel optimization. Therefore, when computing the loss, we do not follow the operation held in hypergradient (Lorraine et al., 2019) but instead add train loss and L_V, and update the model parameter with a single optimizer.

Manifold Mixup (w/ bilevel optim.). As described in Section 3, unlike Mixup, in Manifold Mixup, linear layers are included in μ_α^{lin}, so it is possible to straightforwardly apply bilevel optimization. Therefore, we follow the same procedure described in Section 2, and train using $\mathcal{D}_{\text{context}}$ and $\{x_{i,k}^{(\text{mvalid})}\}_{k=1}^K \sim \mathcal{D}_{\text{unlabeled}}$, with pseudo-labels $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$. Like our method, we also employ a separate optimizer for training μ_α^{lin}.

A.2. Implementation details of the ablation study (Table 2)

In Table 2, we perform three types ablation study of ours :

1. Using $\mathcal{D}_{\text{context}}$ without bilevel optimization.

2. Excluding both $\mathcal{D}_{\text{context}}$ and bilevel optimization.

3. Using the real label for $y_{i,k}^{(\text{mvalid})}$.

Since ablation studies (1) and (3) include the set function μ_λ, we report the performance of each using both DeepSets (Zaheer et al., 2017) and Set Transformer (Lee et al., 2019) as implementations of μ_λ.

For ablation study (1), we train the model with $(x_i, y_i) \sim \mathcal{D}_{\text{train}}$, $\{c_{ij}\}_{j=1}^{m_i} \sim \mathcal{D}_{\text{context}}$ and $\{x_{i,k}^{(\text{mvalid})}\}_{k=1}^K \sim \mathcal{D}_{\text{unlabeled}}$, $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$, and sum the inner loop train loss L_T and outer loop loss L_V to get the final loss for each iteration. Since there is no bilevel optimization in this setting, we update the model parameter of f_θ and μ_λ together using a single optimizer.

For ablation study (2), we train the model with $(x_i, y_i) \sim \mathcal{D}_{\text{train}}$ and $\{x_{i,k}^{(\text{mvalid})}\}_{k=1}^K \sim \mathcal{D}_{\text{unlabeled}}$, $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$, and add the inner loop train loss L_T and the outer loop L_V to get the final loss for each iteration. Since there is no context points, there is no interpolation with external distributions in this setting. Additionally, we update the model parameter of f_θ and μ_λ together using a single optimizer, so the effect of the outer loop loss L_V does not result in any bilevel optimization.

For ablation study (3), we replace $y_{i,k}^{(\text{mvalid})} \sim \mathcal{N}(0,1)$ to a labeled $y_{i,k}^{(\text{mvalid})} \sim \mathcal{D}_{\text{oracle}}$ and perform the same process as our standard setting described in Section 2.

A.3. Hyperparameter tuning details

For hyperparameter settings, all methods (including ours, Mixup, Manifold Mixup, Mixup with outlier exposure, and Manifold Mixup with bilevel optimization) reported in Table 1 used a learning rate of 1×10^{-3} for f_θ, and a learning rate of 1×10^{-5} for the mixers μ_λ and μ_α^{lin}, where the learning rate of μ_α^{lin} is only used for Manifold Mixup (w/ bilevel optim.). The number of layers (L) was set to 3, the hidden dimension (H) to 64, the dropout rate to 0.5, and the optimizer to `adamwschedulefree` (Defazio et al., 2024). For methods utilizing bilevel optimization, we set the number of inner-loop iterations to 10 and outer-loop iterations to 50. We furthure perform hyperparameter search for all the models with the same hyperparameter search space as shown in Table 3. Since Mixup, Manifold Mixup do not utilize $\mathcal{D}_{\text{mvalid}}$, the hyperparameter search was conducted only over M.

For Table 2, all ablation studies used the same hyperparameter search space as in Table 1. The hyperparameter search space is presented in Table 3.

Table 3. Hyperparameter search space.

Hyperparameter	Search Space
maximum number of context per batch (M)	1, 4, 8
number of mvalid per batch (K)	1, 6, 8

A.4. Computing resources

All experiments are conducted on GPUs including the GeForce RTX 2080 Ti, RTX 3090, and RTX 4090.

B. t-SNE Visualization Details

B.1. Embedding generation procedure

In Figure 2, we visualize the penultimate-layer embeddings $\mathbf{z}_i = x_i^{(L-1)}$ using t-SNE (Van der Maaten & Hinton, 2008), across three models that interpolate $\mathcal{D}_{\text{train}}$ with $\mathcal{D}_{\text{context}}$. For each trained model, we generate the following embeddings: $\mathbf{Z}_{\text{joint}}$: obtained by passing both x_i and C_i to the model; $\mathbf{Z}_{\text{input}}$: obtained by passing only x_i; $\mathbf{Z}_{\text{context}}$: obtained by passing only C_i. This allows us to observe how the model handles each type of input. To add an additional qualitative (OOD) evaluation, we utilize one of the Merck datasets (Kulkarni

Table 4. OOD datasets for t-SNE visualizations.

Train dataset	OOD Datasets
HIVPROT	DPP4, NK1
DPP4	HIVPROT, NK1
NK1	DPP4, HIVPROT

et al., 2012) that was not used during training (see Table 4). Using this OOD dataset, we generate the embeddings of $\mathbf{Z}_{\text{ood}}$, by only passing x_i^{ood} to the model. In addition to Figure 2, we also provide extra examples in Figures 3 to 8.

B.2. t-SNE plots across all datasets

Here, we present the t-SNE plots for all datasets used in our experiments, as listed in Table 1. For each in-distribution model, we visualize the corresponding two out-of-distribution datasets. For our method, among the DeepSets (Zaheer et al., 2017) and Set Transformer (Lee et al., 2019) variants, we visualize the model that achieved the highest performance in Table 1.
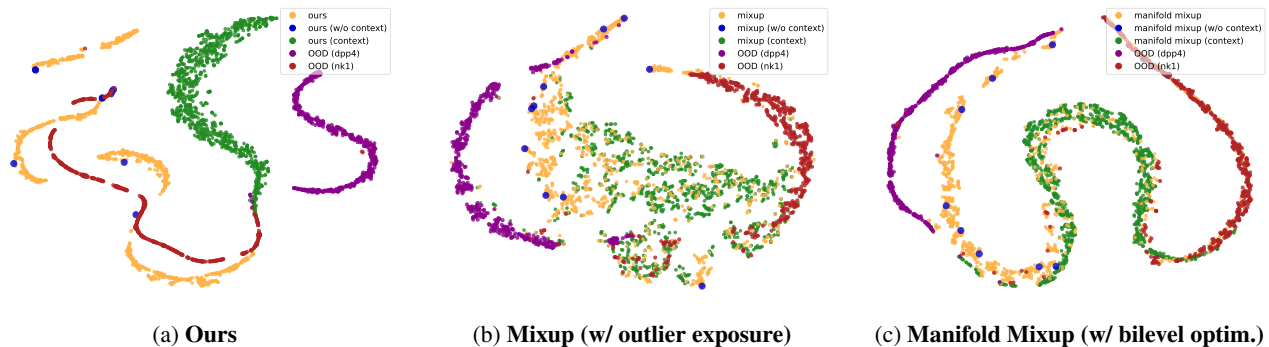
(a) **Ours**　　　　(b) **Mixup (w/ outlier exposure)**　　　　(c) **Manifold Mixup (w/ bilevel optim.)**

Figure 3. **t-SNE visualization of the model trained on the HIVPROT (count) dataset**

(a) **Ours**　　　　(b) **Mixup (w/ outlier exposure)**　　　　(c) **Manifold Mixup (w/ bilevel optim.)**
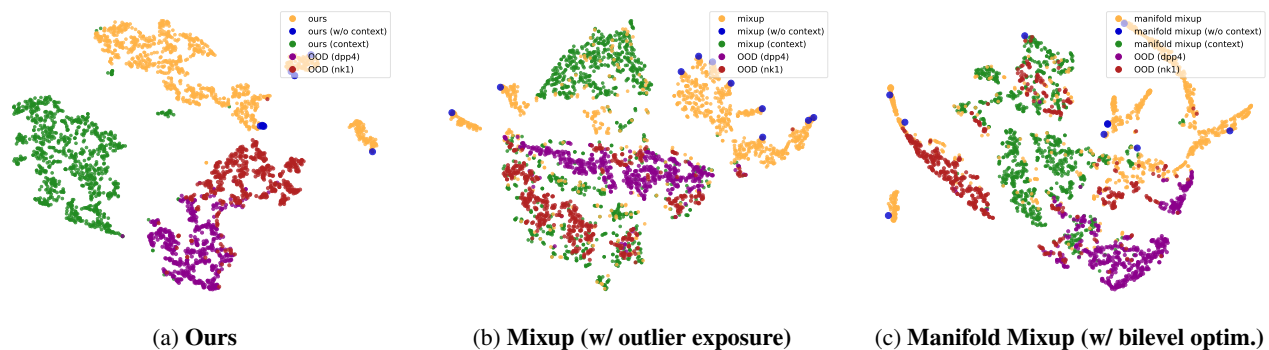
Figure 4. **t-SNE visualization of the model trained on the HIVPROT (bit) dataset**

(a) **Ours**　　　　(b) **Mixup (w/ outlier exposure)**　　　　(c) **Manifold Mixup (w/ bilevel optim.)**
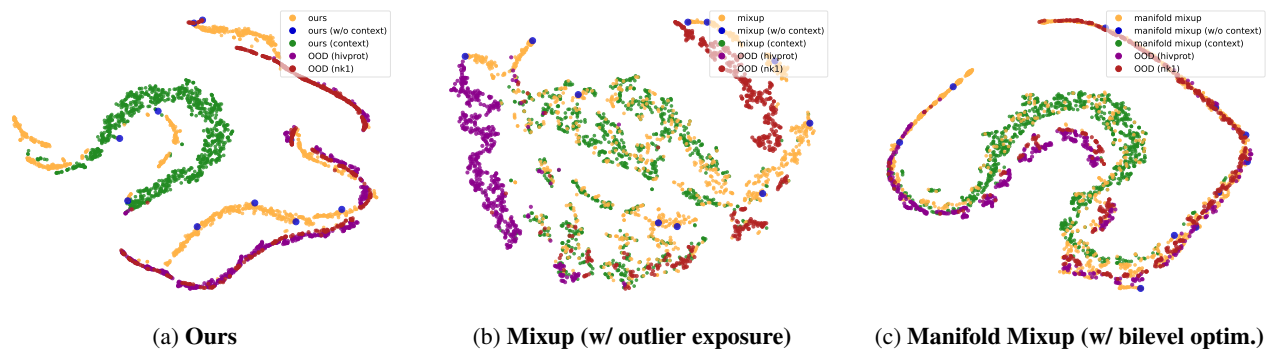
Figure 5. **t-SNE visualization of the model trained on the DPP4 (count) dataset**

(a) **Ours**　　　　(b) **Mixup (w/ outlier exposure)**　　　　(c) **Manifold Mixup (w/ bilevel optim.)**
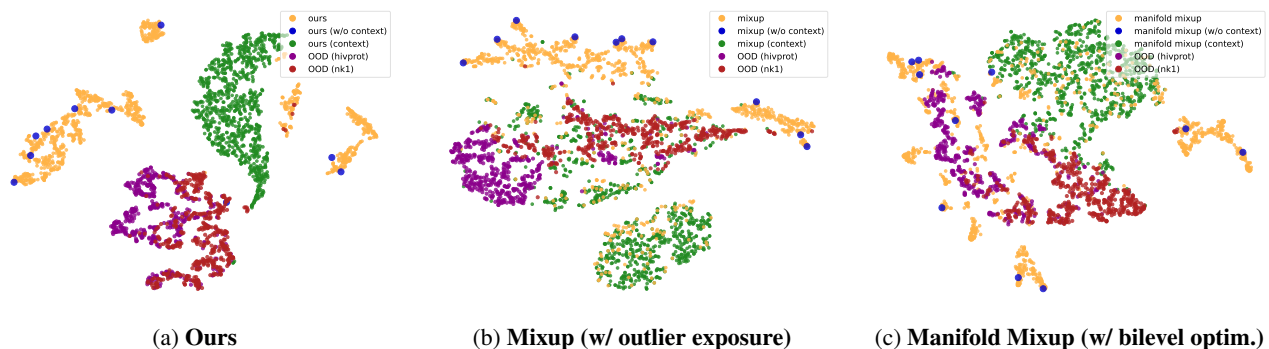
Figure 6. **t-SNE visualization of the model trained on the DPP4 (bit) dataset**

(a) **Ours** (b) **Mixup (w/ outlier exposure)** (c) **Manifold Mixup (w/ bilevel optim.)**
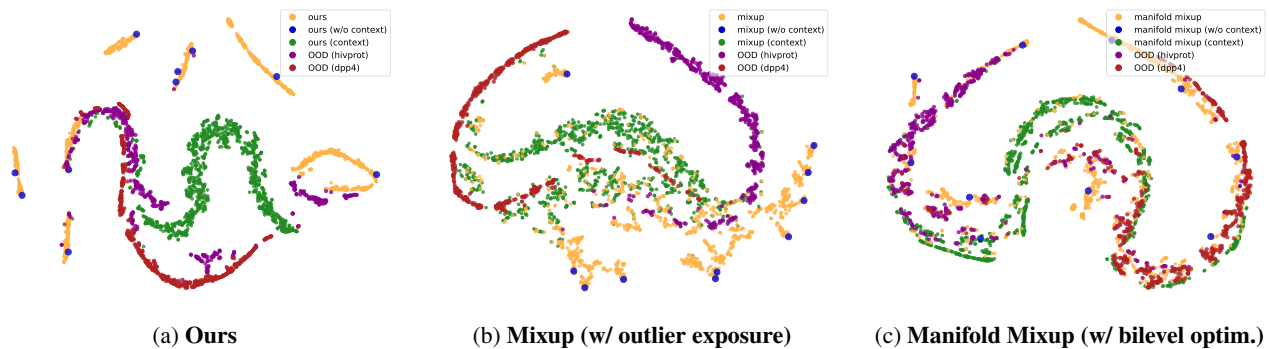
Figure 7. **t-SNE visualization of the model trained on the NK1 (count) dataset**

(a) **Ours** (b) **Mixup (w/ outlier exposure)** (c) **Manifold Mixup (w/ bilevel optim.)**
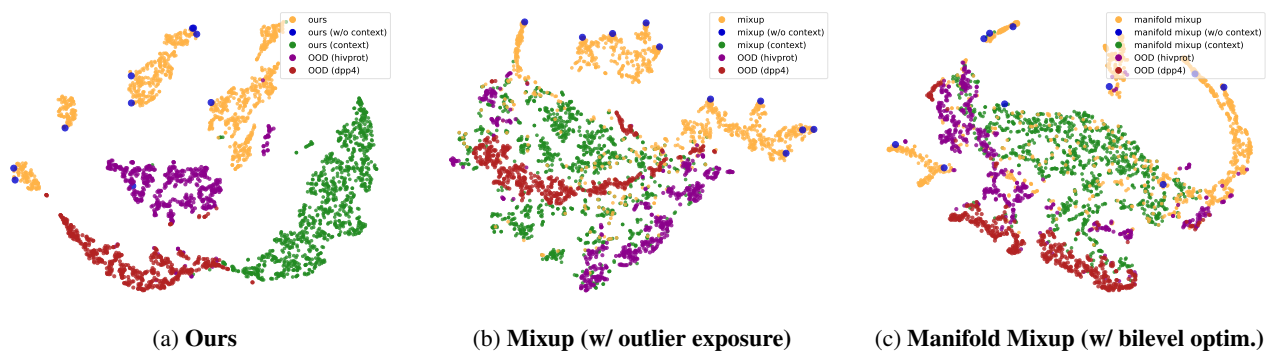
Figure 8. **t-SNE visualization of the model trained on the NK1 (bit) dataset**

C. Model Structure Overview

(a) **Training and Test-time Framework**

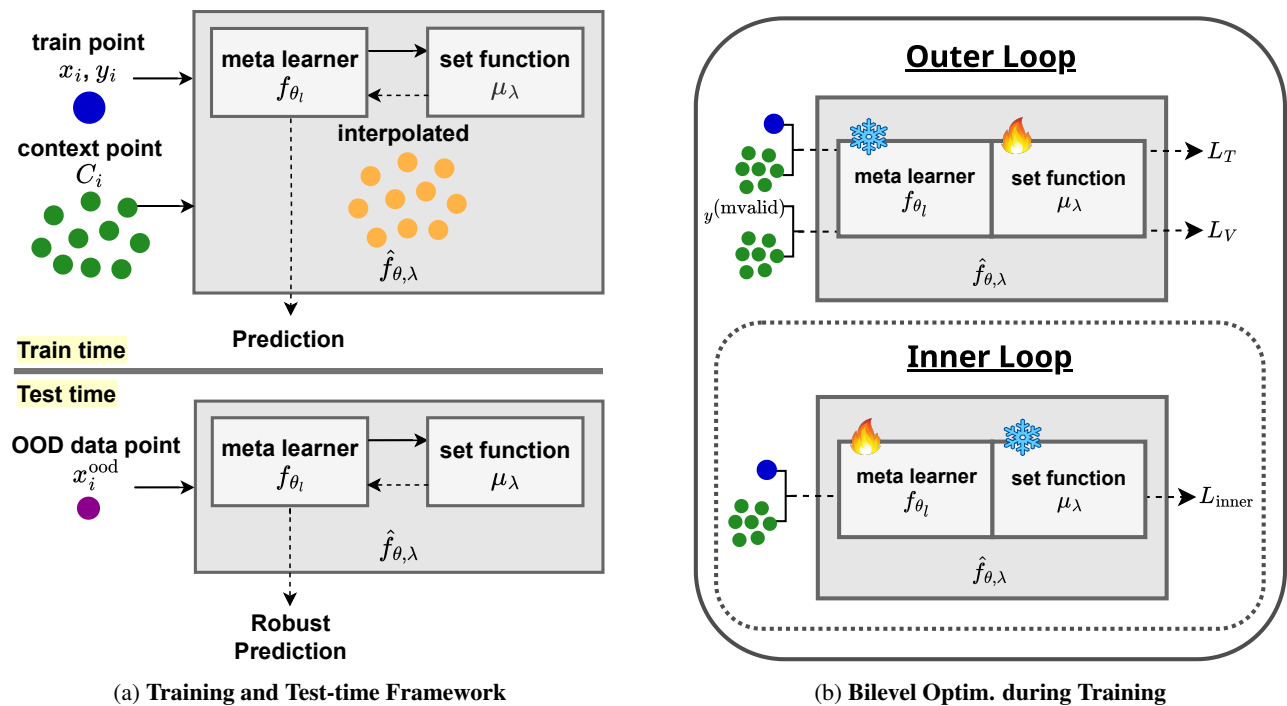(b) **Bilevel Optim. during Training**

Figure 9. **Overview of our proposed model.** (a) During training, the model interpolates between a labeled train point (x_i, y_i) and context point C_i to learn robust representations. At test time, the model predicts on an OOD input using the learned meta learner f_θ and set function μ_λ. (b) The model is trained via bilevel optimization, where the inner loop updates θ using the inner loss L_{inner}, while the outer loop updates λ using the hypergradient computed from L_T and L_V.