8. Implementation Details

Our proposed method contains three components: hierarchical stochastic subgraph generation (Sec. 8.1), hierarchical semantic environments (Sec. 8.2), and graph invariant learning. In this section, we extend the details of our architectures. We provide a detailed description of our model in Line 18 at the end of this section.

8.1. Hierarchical Stochastic Subgraph Generation

We employ graph isomorphism networks (GINs) [42] for graph encoding, utilizing different GINs for different hierarchies. A two-layer multilayer perceptron (MLP) with ReLU activation functions serves as the scoring function for edge selection in each hierarchy, with different weight parameters. A sampler is generated from a Bernoulli distribution based on the computed edge scores. Edge selection uses the Gumbel-Softmax function with a threshold T and temperature τ. During training, we set the temperature at 0.05 for edge selection and we conduct a grid search for the threshold. Variant subgraphs are extracted by the selected edges at each hierarchy, while invariant subgraphs are obtained by directly subtracting the selected edges from the original input graph. Notably, we retain the selected edges from previous hierarchies for stable training.

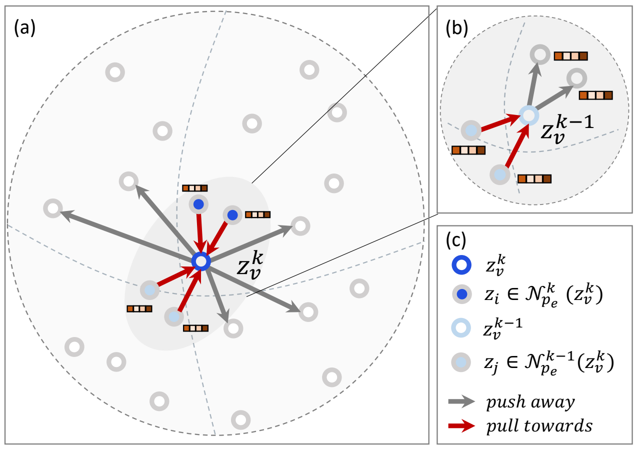
8.2. Hierarchical Semantic Environments

8.2.1 Intra-Hierarchy Environment Diversification

To predict environment labels in each hierarchy, we employ a simple neural network, denoted as f^e, for multi-classification, drawing inspiration from the work [17]. Following Eq. 6, the input of the neural network f^e is formed by concatenating the hidden embedding of the variant subgraph and the one-hot encoding of the label. The output dimension of the neural network f^e corresponds to the number of environment labels at each hierarchy. During inference, predicted environment labels at each hierarchy are obtained by the neural network's output.

8.2.2 Inter-Hierarchy Environment Augmentation

As emphasized by Huang [17], multi-class classification often outperforms clustering-based methods. However, in hierarchical settings, we encounter challenges in consistently assigning samples and their neighbors from previous hierarchies to the same class. To address this issue, we propose a contrastive objective that enhances environmental semantics in the latent embedding space by preserving neighborhood consistencies across hierarchies. Taking inspiration from prior studies [6, 7], we employ contrastive learning on two dimensions: one based on the environment and the other on labels. In Fig. 4, we illustrate the objective $\mathcal{L}_{\text{EnvCon}}$. For each anchor variant subgraph, positive neighborhoods



(a)
z_v^k
(b)
z_v^{k-1}
(c)
z_v^k
$z_i \in \mathcal{N}_{p_e}^k(z_v^k)$
z_v^{k-1}
$z_j \in \mathcal{N}_{p_e}^{k-1}(z_v^k)$
push away
pull towards

Figure 4. Illustration of objective $\mathcal{L}_{\text{EnvCon}}$ in Inter-Hierarchy Environment Augmentation. (a) We pull environment-based neighborhoods $\mathcal{N}_{p_e}^k(z_v^k)$ and $\mathcal{N}_{p_e}^{k-1}(z_v^k)$ toward anchor variant subgraph embedding z_v^k. (b) We show a simple illustration of anchor variant subgraph z_v^{k-1} in the previous hierarchy $k-1$. (c) Notations of the illustration figure.

encompass graphs with the same predicted environmental labels as the anchor graphs, including neighborhoods from the previous hierarchy. The objective $\mathcal{L}_{\text{LabelCon}}$ is computed similarly to $\mathcal{L}_{\text{EnvCon}}$, but the label-based neighborhoods of anchor subgraphs consistently remain the same. Notably, to distinguish embeddings between variant and invariant subgraphs, we employ two MLP layers with distinct parameters to embed each subgraph.

9. Data Statistics

DrugOOD datasets. We present the data statistics for the DrugOOD dataset in Tab. 4. To evaluate the performance of our model in realistic scenarios involving distribution shifts, we have integrated three datasets from the DrugOOD benchmark [20]. This benchmark offers a comprehensive out-of-distribution (OOD) evaluation in AI-driven drug discovery, specifically focusing on predicting drug-target binding affinity for both protein targets and drug compounds. The molecular data and annotations are precisely curated from the ChEMBL 29 database [35], capturing various distribution shifts across various assays, scaffolds, and molecule sizes. We specifically select datasets with three different splits, assay, scaffold, and size, from the ligand-based affinity prediction task. This task involves IC50 and EC50 measurements and encompasses annotation noise at the core level.

General datasets. In *CMNIST-sp* dataset, digits 0-4 are assigned to $y = 0$, and digits 5-9 are assigned to $y = 1$. Following previous studies, the label y will be flipped with a probability of 0.25. In addition, green and red colors will be

Algorithm 1: The training procedure.

```
Data: Dataset $\mathcal{G}^{train} = \{(G_i, y_i)\}_{i=1}^{N_{train}}$,
      hierarchical subgraph generation module $s$,
      hierarchical environment inference module $f$,
      graph invariant learning module $F$. Number
      of hierarchies $K$, number of training epochs
      for hierarchical environment inference
      module $E_1$, number of training epochs for the
      graph invariant learning $E_2$, Batch size $B$.
Result: Final estimated model $F$

1  Initialize the models $s, f, F$;
2  for $p_1 \leftarrow 1$ to $E_1$ do
3      Sample a batch of data $\mathcal{B}$ from $\mathcal{G}_{\text{train}}$ with batch size $B$;
4      for $G_j, y_j \in \mathcal{B}$ do
5          for $k \leftarrow 1$ to $K$ do
6              Obtain environment predictions $e_j^k$ from $f^k$; Find environment-based neighbors
               $\mathcal{N}_{pe}^k(G_j)$ with the same $e_j^k$ in $k$ and $k-1$-th layer;
7              Find label-based neighbors $\mathcal{N}_{py}(G_j)$ with the same $y_j$;
8              Compute loss $\mathcal{L}_{\text{hier}}^k(s^k, f^k; G_j)$ via Eq. 10;
9          Compute loss $\mathcal{L}_{\text{HEI}}$ across all $K$ hierarchies;
10         Back propagate and optimize $s, f$;
11 Freeze the parameters of module $s, f$;
12 for $p_2 \leftarrow 1$ to $E_2$ do
13     Sample a batch of data $B$ from $\mathcal{G}_{\text{train}}$ with batch size $B$;
14     for $G_j, y_j \in \mathcal{B}$ do
15         Determine the environment of each sample $(G, y)$ in $G_j$ by $\text{argmax}_{\hat{e}} f(\hat{e}|s(G), y)$ in
           last hierarchy $K-1$;
16         Compute loss $\mathcal{L}_{\text{inv}}(F; G_j, \hat{e})$ according to Eq. 11;
17         Backpropagate and optimize $F$;
18 Return the final estimated model $F$;
```

assigned to images with labels 0 and 1 on an average probability of 0.15 for the training data, respectively. In *Graph-SST* datasets, the node features are generated using BERT [22] and the edges are parsed by a Biaffine parser [11]. We follow previous works [6, 7] and split the datasets according to the averaged degrees of each graph. In the Graph-SST5 dataset, we partition graphs into the training and test sets using a sorting approach that arranges them from small to large degrees. In contrast, in the Twitter dataset, we adopt an inverse ordering strategy. This means that during training, the model is exposed to graphs with larger degrees, allowing us to assess its generalization performance on graphs with smaller degrees during evaluation.

10. Baselines and Experimental Settings

GNN encoder. For a fair comparison, we employ the same graph neural network (GNN) architecture in all methods. Consistent with prior studies, we utilize a multi-layer graph isomorphism network (GIN) [42] with Batch Normalization [18] between layers and jumping knowledge (JK) residual connections at the last layer. The hidden dimension of the GNNs is set to 128, and we explore the number of GNN layers within the range of $\{3, 4, 5\}$.

Optimization and model selection. We utilize the Adam optimizer [23] and perform a grid search for learning rates in $\{1e{-}3, 1e{-}4, 4e{-}5, 1e{-}5\}$, as well as batch sizes in $\{32, 64, 128\}$ across all models and datasets. Early stopping based on validation performance is implemented with a patience of 20 epochs. A default dropout value of 0.5 is used for all datasets. The final model is selected based on its performance on the validation set. All experiments are conducted with five different random seeds (1, 2, 3, 4, 5), and the results are reported as the mean and standard deviation from these five runs.

Euclidean OOD methods. Building on the implementation of Euclidean OOD methods (IRM [2], V-Rex [26], IB-IRM [1], and EIIL [8]) in the previous study [7], we adopt the same implementation setup to reproduce the results of these methods. As environmental information is not provided, we follow the methodology of previous works [6, 7] and randomly assign training data to two environments. The weights for the regularization term are selected from $\{1.0, 0.01, 0.001\}$ to calculate the IRM loss. Since the obtained results closely align with those reported in [7], we directly use the reported results from [7] for comparison.

Graph OOD methods. We reproduce the experiments for GREA, CIGAv1, CIGAv2, MoleOOD, and GALA using the provided codes, if available. Although the previous work GALA [7] implemented these baselines, our reproduced results differ. We adhere to the author-recommended hyperparameters for training the baselines, and consequently, some of the baselines exhibit superior results compared to the reported GALA results. For the remaining baselines, including LiSA [45], DisC [10], and GIL [27], we observe that these models struggle to fit the DrugOOD dataset with the hyperparameters provided, as indicated in Tab. 6.

Our methods. As previously mentioned, we employ the same GNN architecture to obtain graph embeddings in our

Table 4. **Statistics of the datasets used in experiments. The number of nodes is averaged numbers among all datasets.**

DATASETS	#TRAINING	#VALIDATION	#TESTING	#LABELS	#ENVS	#NODES	#METRICS
CMINST-SP	40,000	5,000	15,000	2	N/A	56.90	ACC
GRAPH-SST5	6,090	1,186	2,240	5	N/A	19.85	ACC
GRAPH-TWITTER	3,238	694	1,509	3	N/A	21.10	ACC
IC50-ASSAY	34,179	19,028	19,032	2	311	34.58	ROC-AUC
IC50-SCA	21,519	19,041	19,048	2	6,881	39.38	ROC-AUC
IC50-SIZE	36,597	17,660	16,415	2	190	37.99	ROC-AUC
EC50-ASSAY	4,540	2,572	2,490	2	47	39.81	ROC-AUC
EC50-SCA	2,570	2,532	2,533	2	850	56.84	ROC-AUC
EC50-SIZE	4,684	2,313	2,398	2	167	48.40	ROC-AUC

METHODS	CMNIST-SP	GRAPH-SST5	TWITTER
ERM	21.56±5.38	42.62±2.54	59.34±1.13
GREA	18.64±6.44	43.29±0.85	59.92±1.48
DISC	54.07±15.3	40.67±1.19	57.89±2.02
CIGAV1	23.66±8.65	44.71±1.14	63.66±0.84
CIGAV2	44.91±4.31	<u>45.25±1.27</u>	<u>64.45±1.99</u>
GIL	18.04±4.39	43.30±1.24	61.78±1.66
MOLEOOD	39.55±4.35	40.36±1.85	59.26±1.67
GALA	**59.16±3.64**	44.88±1.02	62.45±0.62
OURS	<u>46.37±4.33</u>	**46.83±0.54**	**64.51±0.10**

Table 5. Test accuracies of various models on CMNIST-SP, GRAPH-SST5, and TWITTER benchmark datasets. The mean ± standard deviation of all models is reported as an average of 5 executions of each model. The best performance in each dataset is highlighted in **bolded** and the second best methods are <u>underlined</u>.

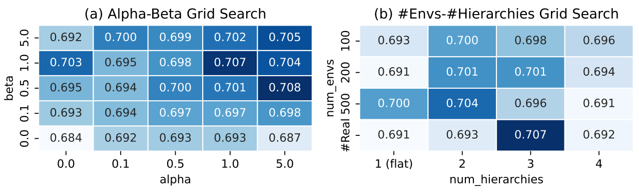


Figure 5. Hyperparameter Selection of alpha, beta, #Envs, and #Hierarchies.

methods. In stochastic subgraph generation, we explore threshold values in $\{0.5, 0.6, 0.8\}$ for different datasets. To address potential over-smoothing issues in GNNs, we set the number of GNN layers in the environment inference stage (first stage) to 1 and search the number of GNN layers in graph invariant learning (second stage) in $\{3, 4, 5\}$. For hierarchical settings, we search for the optimal number of hierarchy layers (i.e., K) from $\{1, 2, 3\}$ for different datasets. As shown in Tab. 4, some datasets, such as CMNIST-sp and Graph-SST, do not provide environmental information. For general datasets, we set the number of environment labels in the first hierarchy (i.e., E_1) to 100. For DrugOOD datasets, we set E_1 as the provided number, e.g., E_1 of the IC50-SCA dataset is 6881. We search for the optimal number of environment labels at the last hierarchy in $\{2, 5, 10\}$ for different datasets. For intermediate hierarchies, we set the median as the number of environment labels, e.g., $K = 3$ and $(E_1, E_2, E_3) = (100, 50, 2)$. We iteratively conduct hierarchical environment inference to obtain more high-level semantic environments. In graph invariant learning with inferred environments, we select the weights for the regularization term from $\{1.0, 0.01, 0.001\}$ to calculate the IRM loss. In Fig. 5 (a), we conducted experiments for two inter-hierarchy contrastive learning. The performance of the model shows an upward trend with varying values of both α and β, demonstrating the effectiveness of incorporating L_{EnvCon} and $L_{LabelCon}$ in our model. In Fig. 5 (b), the hierarchical inference of environments shows superiority over the flat inference (= existing methods) and the selection of the starting number of environments can have different performances.

11. Additional Experiments

We report results on general datasets (CMNIST-SP, Graph-SST5, and Twitter) as shown in Tab. 6. In particular, our methods consistently outperform baseline approaches on Graph-SST datasets. We acknowledge that our methods did not yield successful results on the CMNIST-SP dataset. We analyze the potential reasons as follows: (1) Our model focuses on learning variant substructures hierarchically, which can help infer more meaningful semantic environments for graph invariant learning. (2) While CMNIST-SP contains shifts in node attributes, Graph-SST encounters shifts in node degree (structure). Our method demonstrates robustness in graphs featuring more complex distribution shifts, particularly those influenced by structural variations, as indicated by results on DrugOOD. Furthermore, in Tab. 7, our method outperforms baseline performance when evaluated on three molecular property prediction datasets from the Open Graph Benchmark under fair

METHODS #ENV	EC50-ASSAY 47	EC50-SCA 850	EC50-SIZE 167
LISA [45]	68.10±2.60	64.60±1.70	61.20±1.90
DISC [10]	65.40±5.34	54.97±3.86	56.97±2.56
GIL [27]	72.13±4.70	63.05±1.04	62.08±1.06
OURS	**80.82±0.21**	**69.73±0.21**	**66.87±0.38**

Table 6. Test ROC-AUC of rest models on DrugOOD benchmark datasets. The mean ± standard deviation of all models is reported as an average of 5 executions of each model. The best methods are highlighted in **bold**.

METHODS	MOLBACE	MOLBBBP	MOLHIV
GIN	77.83±3.15	66.93±2.31	76.58±1.02
GIN+MOLEOOD	81.09±2.03	69.84±1.84	78.31±0.24
OURS	**82.26±0.88**	**71.30±0.47**	**79.65±0.14**

Table 7. Molecular property prediction from Open Graph Benchmark (OGB).

DATASETS	DIR	GSAT	GIL	**OURS**
EC50-SCA	64.45±1.69	66.02±1.13	63.05±1.04	**69.73±0.21**
MOLHIV	77.05±0.57	76.47±1.53	79.08±0.54	**79.65±0.14**

Table 8. More baseline comparisons in EC50-SCA (DrugOOD) and MOLHIV (OGB).

conditions. As shown in Tab. 8, our model outperforms the additional three baselines on EC50-SCA and MOLHIV datasets.

12. Qualitative Analysis of Hierarchical Semantic Environments.

To evaluate the practical efficacy of our proposed hierarchical environmental inference model in capturing complex environmental patterns, we utilize a visualization-based approach to compare the assignment outcomes between a flat environment model (Fig. 7) and our model (Fig. 8) for the same set of molecules. We specifically focus on molecules labeled 0 from the IC50-SCA dataset, where our model exhibits accurate predictions while other flat environment-inference models (such as the *Infer#2* model) fail to predict accurately. We then visualize the allocation of these molecules in each environment, with particular emphasis on highlighting scaffold information.

In the *Infer#2* model, the necessity of directly predicting assignments to two environments in molecules with intricate scaffold structures may result in overfitting complex structural features. In contrast, our model addresses the challenge of overfitting to molecular structural features. Through the combined learning of hierarchical environmental inference and graph invariant learning, we effectively extract spurious features, facilitating higher-level assignments of molecules to distinct environments.

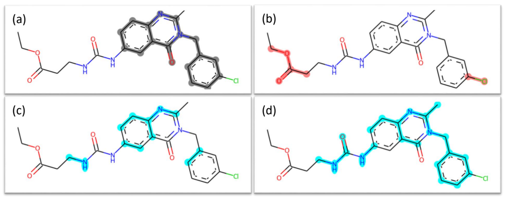


Figure 6. (a) Grey: Scaffold. (b) Red: Functional group (Chlorobenzene), and structural alert COC(=O)C from PubChem. (c, d) Blue: Learned variant subgraphs from the first and second hierarchies (the rest of the graph is considered learned invariant).

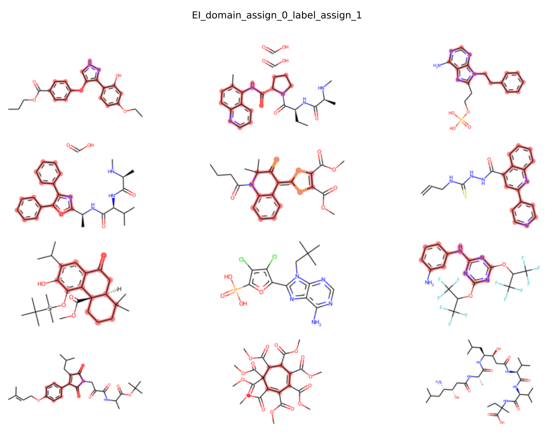
In Fig. 7, the *Infer#2* model may allocate seemingly similar molecules to the same environment, but environments inferred from our model have a higher inter-environment diversification score, which is described in the discussion section (Sec.5.4). This indicates that the *Infer#2* model fails to capture spurious features, instead blending causal features and resulting in inaccurate predictions. In summary, our hierarchical environmental assignment model generates more sophisticated semantic environments by learning hierarchical spurious features through graph invariant learning. This approach mitigates the overfitting challenges observed in models with flat environment inference, leading to a more accurate delineation of meaningful environments.

12.1. Specific Case Studies on Learned Invariant Graphs.

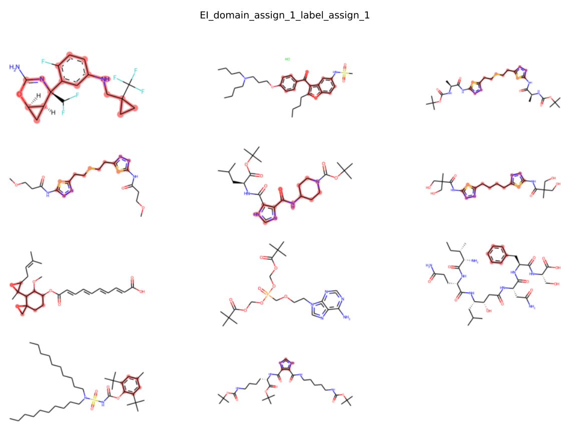
In the DrugOOD-SCA dataset, consisting of 6,881 diverse scaffolds (environments), we computed Morgan-fingerprint Tanimoto similarity for all pairwise scaffolds. The results show a long-tailed distribution with a mean of 0.13 and a standard deviation of 0.04. This reveals the importance of learning the hierarchy of complex environments, given the presence of both similar and dissimilar scaffolds. We performed a specific case study using an active molecule from DrugOOD. Fig. 6 shows our model capturing scaffold-like variants hierarchically while preserving invariant substructures. The final remaining invariant subgraphs are shown to be aligned with the functional group and structural alert.

12.2. Theoretical Analysis of Our Method.

Our method enhances out-of-distribution (OOD) performance by hierarchically stacking intra-hierarchy and inter-hierarchy losses. Specifically, grounded in the theorem of *Structrual Causal Models* [2], the intra-hierarchy loss L_{ED} satisfies the condition $H(Y|X_v) - H(Y|X_v, E_{\text{learn}}) > 0$ from EDNIL [17], thereby maximizing the diversity of inferred environments at each hierarchy. Furthermore, the L_{EnvCon} and L_{LabelCon} in the inter-hierarchy adjust the boundary of the embedding space for variant and invariant

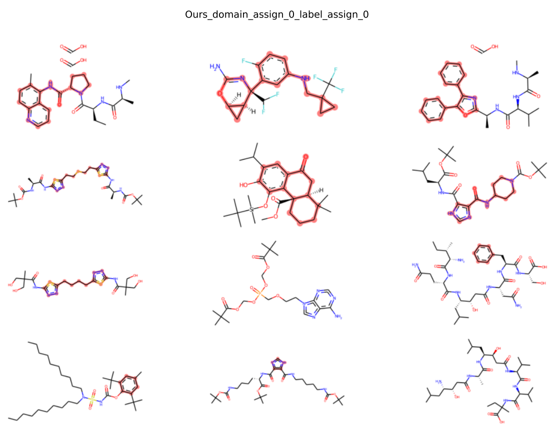


(a) True label: 0; Predicted label: 1; Env: 0

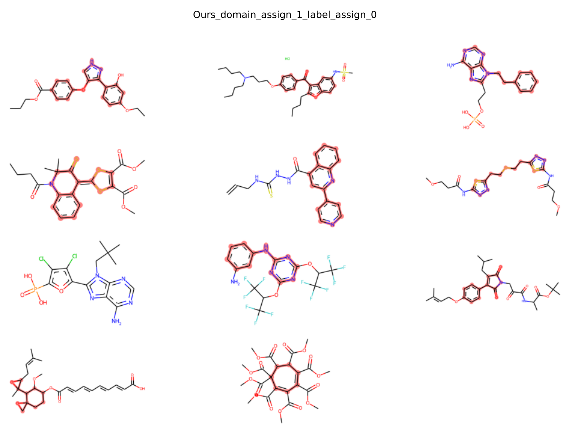


(b) True label: 0; Predicted label: 1; Env: 2

Figure 7. Visualizing molecules with incorrect predictions in their environments assigned by *Infer#2* model in IC50-SCA dataset. The highlighted part with red color represents the scaffold information of each molecule.



(a) True label: 0; Predicted label: 0; Env: 0



(b) True label: 0; Predicted label: 0; Env: 1

Figure 8. Visualizing molecules with incorrect predictions in their environments assigned by our model in IC50-SCA dataset. The highlighted part with red color represents the scaffold information of each molecule.

subgraphs, resulting in both theoretically and empirically diverse and reliable environments for graph invariant learning (Fig. 5).