Generalist Equivariant Transformer Towards 3D Molecular Interaction Learning

Xiangzhe Kong[1,2] **Wenbing Huang**[3] **Yang Liu**[1,2]
[1]Dept. of Comp. Sci. & Tech., Institute for AI, BNRist Center, Tsinghua University
[2]Institute for AIR, Tsinghua University
[3]Gaoling School of Artificial Intelligence, Renmin University of China
jackie_kxz@outlook.com, hwenbing@126.com, liuyang2011@tsinghua.edu.cn

Abstract

Many processes in biology and drug discovery involve various 3D interactions between molecules, such as protein and protein, protein and small molecule, etc. Given that different molecules are usually represented in different granularity, existing methods usually encode each type of molecules independently with different models, leaving it defective to learn the universal underlying interaction physics. In this paper, we first propose to universally represent an arbitrary 3D complex as a geometric graph of sets, shedding light on encoding all types of molecules with one model. We then propose a Generalist Equivariant Transformer (GET) to effectively capture both domain-specific hierarchies and domain-agnostic interaction physics. To be specific, GET consists of a bilevel attention module, a feed-forward module and a layer normalization module, where each module is E(3) equivariant and specialized for handling sets of variable sizes. Notably, in contrast to conventional pooling-based hierarchical models, our GET is able to retain fine-grained information of all levels. Extensive experiments on the interactions between proteins, small molecules and RNA/DNAs verify the effectiveness and generalization capability of our proposed method across different domains.

1 Introduction

Molecular interactions [55], which describe attractive or repulsive forces between molecules and between non-bonded atoms, are crucial in the research of chemistry, biochemistry and biophysics, and come as foundation processes of various downstream applications, including drug discovery, material design, etc [47, 57, 58]. There are different types of molecular interactions, and this paper mainly focuses on the ones that exist in bimolecular complexes, consisting of proteins, small molecules or RNA/DNAs. Specifically, to better capture their physical effects, we study molecular interactions via 3D geometry where atom coordinates are always provided.

Modeling molecular interaction relies heavily on how to represent molecules appropriately. In recent studies, Graph Neural Networks (GNNs) are applied for this purpose [18, 27]. This is motivated by the fact that graphs naturally represent molecules, by considering atoms as nodes and inter-atom interactions or bonds as edges. When further encapsulating 3D atom coordinates, geometric graphs [16, 49, 52] are used in place of conventional graph modeling that solely encodes topology. To process geometric graphs, equivariant GNNs, a new kind of GNNs that meet E(3) equivariance regarding translation, rotation and reflection are proposed, which exhibit promising performance in molecule interaction tasks [33, 39, 56, 66].

Despite the encouraging progress, there still lacks a desirable and unified form of cross-domain molecular representation in molecular interaction. The molecules of different domains like small

The NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development (AI4D3 2023), New Orleans, LA, USA, 2023.

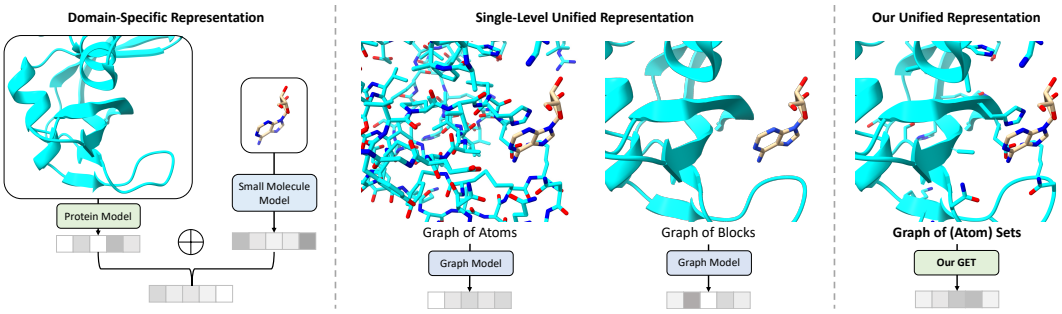

Figure 1: Domain-specific representations and unified representations in molecular interaction.

molecules, proteins, and RNA/DNAs are usually represented in different granularity, which consist of atoms, residues, and nucleobases, respectively. Existing approaches typically design domain-specific representations and model each of the interacting instances independently [51, 61], which are defective in learning the universal underlying interaction physics. Therefore, designing unified cross-domain molecular representation is demanded, which, however, is non-trivial. For one thing, directly applying unshared block-level graphs, whose nodes correspond to domain-specific building blocks, leads to limited transferability of the model from one domain to another. For another, decomposing all molecules into atom-level graphs discards the block specificity (*e.g.* which residue each atom belongs to) and overlook valuable heuristics for representation learning. It is still an open problem in designing a universal representation and a generalist model thereon to capture both the block-level specificity and atom-level shareability.

In this paper, we approache this problem by modeling a complex involved in molecular interaction as *a geometric graph of sets*. This representation follows a bilevel design: in the top level, a complex is represented as a geometric graph of blocks; in the bottom level, each block contains a set of atomic instances. It is nontrivial to process such bilevel geometric graphs, as the model should handle blocks of variable sizes and ensure certain specific geometries. To this end, we propose *Generalist Equivariant Transformer (GET)*, which consists of the three modules: bilevel attention module, feed-forward module and layer normalization module. To be specific, the bilevel attention module updates the information of each atom by adopting both sparse block-level and dense atom-level attentions. The feed-forward module is to inject the intra-block geometry to each atom, and the layer normalization module is proposed to stabilize and accelerate the training. All the modules are E(3)-equivariant regarding the 3D coordinates, permutation-invariant regarding all atoms within each block, and work regardless of the varying block size. We compare our method with other representation approaches in Figure 1.

Notably, our formulation of graph of sets is relevant to conventional pooling-based hierarchical models based on graph of graphs [28]. Nevertheless, these hierarchical architecture are usually inefficient and will blot out the fine-grained information after certain pooling-based aggregation, while our GET is able to retain both the atom-level and block-level information.

We conduct experiments on various molecular interactions between proteins, small molecules and RNA/DNAs. The results exhibit the superiority of our GET on the proposed unified representation over traditional methods including domain-specific independent models, single-level unified representations and hierarchical models. More excitingly, we identify strong potential of GET in capturing and transferring universal knowledge across different domains, and enabling zero-shot performance on RNA/DNA-ligand binding affinity prediction.

2 Method

We start by illustrating the proposed unified representation for molecules in § 2.1. Then we introduce GET in § 2.2. Each layer of GET consists of the three types of E(3)-equivariant modules: a bilevel attention module, a feed-forward module, and a layer normalization after each previous module. The overall concepts are depicted in Figure 2.

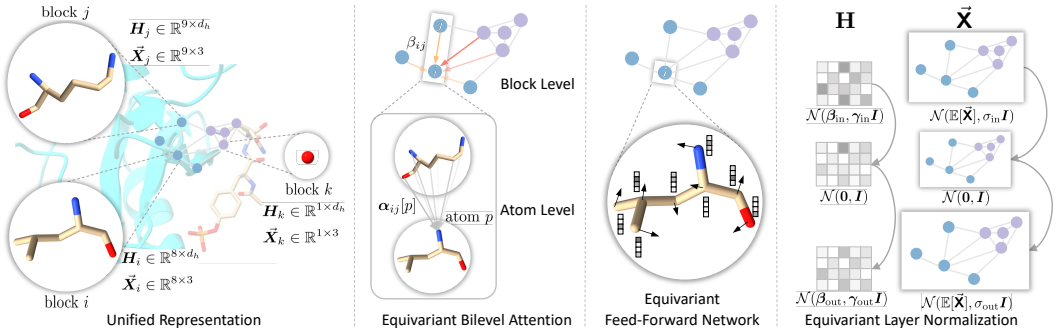

Figure 2: Overview of the unified representation and the equivariant modules in our Generalist Equivariant Transformer (GET). From left to right: The unified representation treats molecules as geometric graphs of sets according to predefined building blocks; The bilevel attention module captures both sparse block-level and dense atom-level interactions via an equivariant attention mechanism; The feed-forward network injects the block-level information into the intra-block atoms; The layer normalization transforms the input distribution with trainable scales and offsets.

2.1 Unified Representation: Geometric Graphs of Sets

Graphs come as a central tool for molecular representations, and different kind of graphs is applied in different case. For instance, small molecules can be represented as single-level graphs, where each node is an atom, while proteins (RNA/DNAs) correspond to two-level graphs, where each node is a residue (nucleobase) that consists of a variable number of atoms. To better characterize the interaction between different molecules, below we propose a unified molecular representation.

Given a complex consisting of a set of atoms $\mathbb{A}$, we first identify a set of blocks (*i.e.* subsets) from $\mathbb{A}$ according to some predefined notions (*e.g.* residues for proteins). Then the complex is abstracted as a geometric graph of sets $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{(\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i) | 1 \le i \le B\}$ includes all B blocks and $\mathcal{E} = \{(i, j, \boldsymbol{e}_{ij}) | 1 \le i, j \le B\}$ includes all edges between blocks[1], where $\boldsymbol{e}_{ij} \in \mathbb{R}^{d_e}$ distinguishes the type of the edge as intra-molecular or inter-molecular connection. In each block composed of n_i atoms, $\boldsymbol{H}_i \in \mathbb{R}^{n_i \times d_h}$ denotes a set of atom feature vectors and $\vec{\boldsymbol{X}}_i \in \mathbb{R}^{n_i \times 3}$ denotes a set of 3D atom coordinates. To be specific, the p-th row of $\boldsymbol{H}_i$, which is the feature vector of atom p, sums up the trainable embeddings of atom types $\boldsymbol{a}_i[p]$, block types b_i, and atom position codes $\boldsymbol{p}_i[p]$ (see Appendix C), namely, $\boldsymbol{H}_i[p] = \text{Embed}(\boldsymbol{a}_i[p]) + \text{Embed}(b_i) + \text{Embed}(\boldsymbol{p}_i[p]), 1 \le p \le n_i$. To reduce the computational complexity, we construct $\mathcal{E}$ via k-nearest neighbors according to the block distance which is defined as the minimum distance between inter-block atom pairs:

$$d(i, j) = \min\{\|\vec{\boldsymbol{X}}_i[p] - \vec{\boldsymbol{X}}_j[q]\|_2 \mid 1 \le p \le n_i, 1 \le q \le n_j\}. \tag{1}$$

As we will observe in the next section, the above bilevel design allows our model to capture sparse interactions for the top level and dense interactions for the bottom level, achieving a desirable integration of different granularities. We will also demonstrate in the experiments that the representation can be easily extended to arbitrary block definitions (*e.g.* subgraph-level decomposition of small molecules [33, 17]).

Connection to Single-Level Representations If we restrict the blocks to one-atom subsets only, then we obtain the **atom-level** representation where each node is one atom, and correspondingly both $\boldsymbol{H}_i$ and $\vec{\boldsymbol{X}}_i$ are downgraded to row vectors as $n_i \equiv 1$. If we retain the building blocks but replace $\boldsymbol{H}_i$ and $\vec{\boldsymbol{X}}_i$ with their centroids, then we obtain the **block-level** representation where the atoms in the same block are pooled into one single instance. Both single-level representations assign a vector and a 3D coordinate to each node, hence can be fed into most structural learning models [16, 48, 49, 54]. On the contrary, the proposed bilevel representation requires the capability of processing E(3)-equivariant feature matrices ($\boldsymbol{H}_i$ and $\vec{\boldsymbol{X}}_i$) with a variable number of rows, which cannot be directly processed by existing models. Additionally, within each block, the rows of $\boldsymbol{H}_i$ and $\vec{\boldsymbol{X}}_i$ are indeed elements in a set and their update should be unaffected by the row order. Luckily, the above challenges are well handled by our Generalist Equivariant Transformer proposed in the next section.

[1]We have added self-loops to reflect self-interactions between the atoms in each block.

Comparison with Hierarchical Representations Previous studies [28] model proteins in a hierarchical manner, where the atom-level features within each residue are first pooled as the residue-level features that will be processed via the message passing over the graph of residues. In contrast to these hierarchical methods, our bilevel representation retains the information of both the atom-level and residue-level features simultaneously for attention-based message passing.

2.2 Generalist Equivariant Transformer

Upon the unified representation, we propose GET to model the structure of the input complex. As mentioned above, one beneficial property of GET is that it can tackle blocks of variable size. Besides, GET is sophisticatedly designed to ensure E(3)-equivariance and intra-block permutation invariance to handle the symmetry. Specifically, each layer of GET first exploits an equivariant bilevel attention module to capture both sparse interactions in block level and dense interactions in atom level. Then an equivariant feed-forward module updates each atom with the geometry of its block. Finally, a novel equivariant layer normalization is implemented on both the hidden states and coordinates. We present a detailed scheme of GET in Appendix D for better understanding.

Equivariant Bilevel Attention Module Given two blocks i and j of n_i and n_j atoms, respectively, we first obtain the query, the key, and the value matrices as follows:

$$\boldsymbol{Q}_i = \boldsymbol{H}_i \boldsymbol{W}_Q, \qquad \boldsymbol{K}_j = \boldsymbol{H}_j \boldsymbol{W}_K, \qquad \boldsymbol{V}_j = \boldsymbol{H}_j \boldsymbol{W}_V, \tag{2}$$

where $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V \in \mathbb{R}^{d_h \times d_r}$ are trainable parameters. We denote $\vec{\boldsymbol{X}}_{ij} \in \mathbb{R}^{n_i \times n_j \times 3}$ and $\boldsymbol{D}_{ij} \in \mathbb{R}^{n_i \times n_j}$ as the relative coordinates and distances between any atom pair in block i and j, namely, $\vec{\boldsymbol{X}}_{ij}[p,q] = \vec{\boldsymbol{X}}_i[p] - \vec{\boldsymbol{X}}_j[q], \boldsymbol{D}_{ij}[p,q] = \|\vec{\boldsymbol{X}}_{ij}[p,q]\|_2$.

The **atom-level cross attention values** from j to i are calculated by:

$$\boldsymbol{R}_{ij}[p,q] = \phi_A(\boldsymbol{Q}_i[p], \boldsymbol{K}_j[q], \mathrm{RBF}(\boldsymbol{D}_{ij}[p,q]), \boldsymbol{e}_{ij}), \tag{3}$$

$$\boldsymbol{\alpha}_{ij} = \mathrm{Softmax}(\boldsymbol{R}_{ij}\boldsymbol{W}_A). \tag{4}$$

Here, $\boldsymbol{e}_{ij}$ is the optional edge feature to distinguish between intra-molecule edges and inter-molecule edges; ϕ_A is an Multi-Layer Perceptron (MLP); RBF embeds the distance with radial basis functions; $\boldsymbol{R}_{ij} \in \mathbb{R}^{n_i \times n_j \times d_r}$ represents the relations between each atom pair in block i and j, which are later mapped to scalars by $\boldsymbol{W}_A \in \mathbb{R}^{d_r \times 1}$ to obtain the atom-level cross attentions $\boldsymbol{\alpha}_{ij} \in \mathbb{R}^{n_i \times n_j}$ between the two blocks through Softmax alone the columns of $\boldsymbol{R}_{ij}\boldsymbol{W}_A \in \mathbb{R}^{n_i \times n_j}$.

The **block-level cross attention value** from j to i is given by:

$$\boldsymbol{r}_{ij} = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \boldsymbol{R}_{ij}[p,q], \tag{5}$$

$$\beta_{ij} = \frac{\exp(\boldsymbol{r}_{ij}\boldsymbol{W}_B)}{\sum_{j \in \mathcal{N}(i)} \exp(\boldsymbol{r}_{ij}\boldsymbol{W}_B)}, \tag{6}$$

where $\boldsymbol{W}_B \in \mathbb{R}^{d_r \times 1}$, and $\mathcal{N}(i)$ denotes the neighborhood blocks of i. Basically, $\boldsymbol{r}_{ij} \in \mathbb{R}^{d_r}$ represents the global relation between i and j after aggregating all values in $\boldsymbol{R}_{ij}$, which is then mapped to a scalar to obtain the block-level cross attentions β_{ij} through Softmax in the neighborhood of i.

With the atom-level and the block-level attentions at hand, we are ready to update both the hidden states and coordinates for each atom p in block i:

$$\boldsymbol{m}_{ij,p} = \boldsymbol{\alpha}_{ij}[p] \cdot \phi_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p])) \tag{7}$$

$$\vec{\boldsymbol{m}}_{ij,p} = \boldsymbol{\alpha}_{ij}[p] \cdot (\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p]))) \tag{8}$$

$$\boldsymbol{H}'_i[p] = \boldsymbol{H}_i[p] + \sum_{j \in \mathcal{N}(i)} \beta_{ij} \phi_m(\boldsymbol{m}_{ij,p}), \tag{9}$$

$$\vec{\boldsymbol{X}}'_i[p] = \vec{\boldsymbol{X}}_i[p] + \sum_{j \in \mathcal{N}(i)} \beta_{ij} (\vec{\boldsymbol{m}}_{ij,p} \odot \sigma_m(\boldsymbol{m}_{ij,p})), \tag{10}$$

where, $\parallel$ specifies the concatenation along the second dimension; ϕ_v,ϕ_m, σ_v, and σ_m are all MLPs, ϕ_v and σ_v are applied for each row of the input matrix independently; $\odot$ computes the element-wise

multiplication. It is verified that the shape of the updated variables $\boldsymbol{H}_i'$ and $\vec{\boldsymbol{X}}_i'$ keeps the same irregardless of the value of the block size n_j. In addition, since the attentions $\boldsymbol{\alpha}_{ij}$ and β_{ij} are E(3)-invariant, the update of $\vec{\boldsymbol{X}}_i'$ is E(3)-equivariant. It can also be observed that the update is independent to the atom permutation of each block. We provide detailed proofs in Appendix E.

Equivariant Feed-Forward Network This module updates $\boldsymbol{H}_i$ and $\vec{\boldsymbol{X}}_i$ for each atom individually. We denote each row of $\boldsymbol{H}_i$ as $\boldsymbol{h}$, and $\vec{\boldsymbol{X}}_i$ as $\vec{\boldsymbol{x}}$. We first calculate the centroids of the block:

$$\boldsymbol{h}_c = \text{centroid}(\boldsymbol{H}_i), \qquad \vec{\boldsymbol{x}}_c = \text{centroid}(\vec{\boldsymbol{X}}_i). \tag{11}$$

Then we obtain the relative coordinate $\Delta\vec{\boldsymbol{x}}$ as well as the distance representation $\boldsymbol{r}$ between each atom and the centroid:

$$\Delta\vec{\boldsymbol{x}} = \vec{\boldsymbol{x}} - \vec{\boldsymbol{x}}_c, \qquad \boldsymbol{r} = \text{RBF}(\|\Delta\vec{\boldsymbol{x}}\|_2), \tag{12}$$

The centroids and the distance representation are then integrated into the updating process of $\boldsymbol{h}$ and $\vec{\boldsymbol{x}}$ to let each atom be aware of the geometric context of its block, where ϕ_h, σ_x are MLPs:

$$\boldsymbol{h}' = \boldsymbol{h} + \phi_h(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}), \tag{13}$$

$$\vec{\boldsymbol{x}}' = \vec{\boldsymbol{x}} + \Delta\vec{\boldsymbol{x}}\sigma_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}), \tag{14}$$

Equivariant Layer Normalization Layer normalization is known to stabilize and accelerate the training of deep neural networks [5, 59]. The challenge here is that we need to additionally consider E(3)-equivariance when normalizing the coordinates. To this end, we first extract the centroid of the entire graph as $\mathbb{E}[\vec{\mathbf{X}}]$, where $\vec{\mathbf{X}}$ collects the coordinates of all atoms in all blocks. Then we exert layer normalization on the hidden vectors and coordinates of individual atoms as follows:

$$\boldsymbol{h}' = \frac{\boldsymbol{h} - \mathbb{E}[\boldsymbol{h}]}{\sqrt{\text{Var}[\boldsymbol{h}]}} \cdot \boldsymbol{\gamma} + \boldsymbol{\beta}, \tag{15}$$

$$\vec{\boldsymbol{x}}' = \frac{\vec{\boldsymbol{x}} - \mathbb{E}[\vec{\mathbf{X}}]}{\sqrt{\text{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]]}} \cdot \sigma + \mathbb{E}[\vec{\mathbf{X}}], \tag{16}$$

where $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$, and σ are learnable parameters, and $\text{Var}[\vec{\mathbf{X}}]$ calculates the variation of all atom coordinates with respect to the centroid. Therefore, the coordinates, after subtracting the centroid of all atoms, are first normalized to standard Gaussian distribution and then scaled with σ before recovering the centroid. In addition, to further reflect the rescaling of the coordinates into hidden features, we inject the following update before applying the above layer normalization:

$$\boldsymbol{h} = \boldsymbol{h} + \phi_{\text{LN}}(\text{RBF}(\sigma/\sqrt{\text{Var}[\vec{\mathbf{X}}]})), \tag{17}$$

where ϕ_{LN} is an MLP. In contrast to existing literature which only implements layer normalization on E(3)-invariant features [54, 34] or node-wise velocities [65], ours works on both E(3)-invariant features and E(3)-equivariant coordinates.

Thanks to the E(3)-equivariance of each module, GET, which is the cascading of these modules in each layer, also conforms to the symmetry of the 3D world. We provide the proof in Appendix E and complexity analysis in Appendix F.

3 Experiments

In this section, we aim to answer the following three questions via empirical experiments: (1) Does modeling complexes with unified representation better captures the geometric interactions than treating each interacting entity independently with domain-specific representations (§ 3.1)? (2) Is the proposed unified representation more expressive than vanilla single-level representations or pooling-based hierarchical methods (§ 3.2)? (3) Can the proposed method generalize to different domains by learning the universal underlying interaction physics (§ 3.3)?

We conduct experiments on prediction of binding between proteins, small molecules and nucleic acids. Thus, we adopt three widely used metrics for quantitive evaluation [56, 36, 40, 41]: **RMSE** is the Root Mean Square Error of the predicted value; **Pearson Correlation** [8] measures the linear correlation between the predicted values and the target values; **Spearman Correlation** [21] measures the correlation between the rankings given by the predicted and the target values

3.1 Comparison to Domain-Specific Representations

We evaluate our method on the prediction of binding affinity between proteins and small molecules against state-of-the-art two-branch models with domain-specific representations from existing literature [51, 61]. We follow Somnath et al. [51], Wang et al. [61] to conduct experiments on the well-established PDBbind [62, 37] and split the dataset according to sequence identity of the protein with 30% as the threshold. Details of the experiments are provided in Appendix H.

Table 1: The mean and the standard deviation of three runs on the PDBbind benchmark. The best results are marked in bold and the second best are underlined. The results of baselines are borrowed from Wang et al. [61]. Baselines encoding the complexes with one model are marked with $*$.

Model	RMSE↓	Pearson↑	Spearman↑
DeepDTA [42]	1.866 ± 0.080	0.472 ± 0.022	0.471 ± 0.024
Bepler and Berger [7]	1.985 ± 0.006	0.165 ± 0.006	0.152 ± 0.024
TAPE [45]	1.890 ± 0.035	0.338 ± 0.044	0.286 ± 0.124
ProtTrans [11]	1.544 ± 0.015	0.438 ± 0.053	0.434 ± 0.058
MaSIF [14]	1.484 ± 0.018	0.467 ± 0.020	0.455 ± 0.014
IEConv [23]	1.554 ± 0.016	0.414 ± 0.053	0.428 ± 0.032
Holoprot-Full Surface [51]	1.464 ± 0.006	0.509 ± 0.002	0.500 ± 0.005
Holoprot-Superpixel [51]	1.491 ± 0.004	0.491 ± 0.014	0.482 ± 0.032
ProNet-Amino Acid [61]	1.455 ± 0.009	0.536 ± 0.012	0.526 ± 0.012
ProtNet-Backbone [61]	1.458 ± 0.003	0.546 ± 0.007	0.550 ± 0.008
ProtNet-All-Atom [61]	1.463 ± 0.001	$\underline{0.551 \pm 0.005}$	0.551 ± 0.008
GVP* [29]	1.594 ± 0.073	-	-
Atom3D-3DCNN* [56]	$\underline{1.416 \pm 0.021}$	0.550 ± 0.021	$\underline{0.553 \pm 0.009}$
Atom3D-ENN* [56]	1.568 ± 0.012	0.389 ± 0.024	0.408 ± 0.021
Atom3D-GNN* [56]	1.601 ± 0.048	0.545 ± 0.027	0.533 ± 0.033
GET* (ours)	$\mathbf{1.364 \pm 0.009}$	$\mathbf{0.596 \pm 0.006}$	$\mathbf{0.573 \pm 0.007}$

Results Table 1 shows that our GET surpasses the baselines by a large margin. Compared to the baselines which encode proteins and small molecules independently with delicately designed domain-specific models, our unified representation enables unified geometric learning with only one model, which better captures the interactive geometric information between the protein and the small molecule. Notably, among models with one encoder [29, 56], our method also achieves significant improvement since our unified representation retains domain-specific hierarchies instead of decomposing all types of molecules into atomic graphs.

3.2 Comparison to Vanilla Unified Representations

Next, we compare the proposed unified representation with three vanilla unified representations: (1) **Atom**-level methods treats all kinds of molecules as graphs of atoms; (2) **Block**-level methods assign each building block to one node where the definition of building block is domain-specific (e.g. each residue in the proteins is one node); (3) **Hierarchical** methods first implement message passing on atom-level graphs, then obtain the block-level representations by pooling for further message passing on the block-level graphs [28].

Baselines These vanilla unified representations are compatible with most geometric graph models in existing literature [49, 48]. Therefore we adopt the following representative models as the backbone for comparison: **SchNet** [49] implements continuous-filter convolution on the 3D molecular graph to capture the geometry; **DimeNet++** [16, 15] exploits the radial basis function (RBF) and the spherical basis function (SBF) as input features to learn both the spatial distance and directions with directional message passing; **EGNN** [48] is a lightweight yet effective E(n)-equivariant graph neural network using only relative distances for geometric learning; **TorchMD** [54] utilizes RBF to define the geometry and proposes an equivariant attention mechanism to compose an equivariant graph Transformer.

Dataset To this end, we evaluate the models on prediction of protein-protein affinity and ligand-binding affinity. For **Protein-Protein Affinity** (PPA), we adopt the Protein-Protein Affinity Benchmark Version 2 [31, 60] as the test set, which categorizes 176 diversified protein-protein complexes into three difficulty levels (i.e. Rigid, Medium, Flexible) according to the conformation change of

the proteins from the unbound to the bound state [31]. The Flexible split is the most challenging as the proteins undergo large conformation change upon binding. As for training, we obtain 2,500 complexes with annotated binding affinity (K_i or K_d) from PDBbind [62] and split the dataset according to sequence identity on a threshold of 30%. For **Ligand-Binding Affinity** (LBA), we use the LBA dataset and its splits in Atom3D benchmark [56], where there are 3507, 466, and 490 complexes in the training, the validation, and the test sets. Details are provided in Appendix H.

Table 2: The mean and the standard deviation of three runs on PPA prediction. The best results are marked in bold and the second best are underlined.

Repr.	Model	Pearson↑	Spearman↑			
		All	All	Rigid	Medium	Flexible
Block	SchNet	$\underline{0.439 \pm 0.016}$	0.427 ± 0.012	0.476 ± 0.015	0.520 ± 0.013	0.068 ± 0.009
	DimeNet++	0.323 ± 0.025	0.317 ± 0.031	0.466 ± 0.088	0.368 ± 0.037	$\underline{0.171 \pm 0.054}$
	EGNN	0.381 ± 0.021	0.382 ± 0.022	0.364 ± 0.043	0.455 ± 0.026	0.080 ± 0.038
	TorchMD	0.424 ± 0.021	0.415 ± 0.027	0.552 ± 0.039	0.482 ± 0.025	0.090 ± 0.062
Atom	SchNet	0.369 ± 0.007	0.404 ± 0.016	0.546 ± 0.005	0.512 ± 0.007	0.028 ± 0.032
	DimeNet++[2]	-	-	-	-	-
	EGNN	0.302 ± 0.010	0.349 ± 0.009	0.450 ± 0.042	0.438 ± 0.021	0.027 ± 0.030
	TorchMD	0.401 ± 0.005	0.436 ± 0.004	$\underline{0.582 \pm 0.025}$	0.487 ± 0.002	0.117 ± 0.008
Hierarchical	SchNet	0.438 ± 0.017	0.424 ± 0.016	0.476 ± 0.017	$\underline{0.523 \pm 0.014}$	0.072 ± 0.021
	DimeNet++	-	-	-	-	-
	EGNN	0.386 ± 0.021	0.390 ± 0.016	0.387 ± 0.023	0.461 ± 0.020	0.078 ± 0.043
	TorchMD	0.401 ± 0.005	$\underline{0.438 \pm 0.029}$	0.547 ± 0.045	0.516 ± 0.019	0.100 ± 0.111
Unified	GET (ours)	$\mathbf{0.514 \pm 0.011}$	$\mathbf{0.533 \pm 0.011}$	$\mathbf{0.622 \pm 0.030}$	$\mathbf{0.533 \pm 0.014}$	$\mathbf{0.363 \pm 0.017}$

Table 3: The mean and the standard deviation of three runs on LBA prediction. The best results are marked in bold and the second best are underlined.

Repr.	Model	RMSE↓	Pearson↑	Spearman↑
Block	SchNet	1.406 ± 0.020	0.565 ± 0.006	0.549 ± 0.007
	DimeNet++	1.391 ± 0.020	0.576 ± 0.016	0.569 ± 0.016
	EGNN	1.409 ± 0.015	0.566 ± 0.010	0.548 ± 0.012
	TorchMD	1.367 ± 0.037	0.599 ± 0.017	0.584 ± 0.025
Atom	SchNet	1.357 ± 0.017	0.598 ± 0.011	0.592 ± 0.015
	DimeNet++	1.439 ± 0.036	0.547 ± 0.015	0.536 ± 0.016
	EGNN	1.358 ± 0.000	0.599 ± 0.002	0.587 ± 0.004
	TorchMD	1.381 ± 0.013	0.591 ± 0.007	0.583 ± 0.009
Hierarchical	SchNet	1.370 ± 0.028	0.590 ± 0.017	0.571 ± 0.028
	DimeNet++	1.388 ± 0.010	0.582 ± 0.009	0.574 ± 0.007
	EGNN	1.380 ± 0.015	0.586 ± 0.004	0.568 ± 0.004
	TorchMD	1.383 ± 0.009	0.580 ± 0.008	0.564 ± 0.004
Unified	GET (ours)	$\underline{1.327 \pm 0.005}$	$\underline{0.620 \pm 0.004}$	$\underline{0.611 \pm 0.003}$
	GET-PS (ours)	$\mathbf{1.309 \pm 0.012}$	$\mathbf{0.633 \pm 0.008}$	$\mathbf{0.642 \pm 0.009}$

Results For protein-protein affinity, we report the mean and the standard deviation of the metrics across three runs on different difficulty levels as well as on the overall test set in Table 2, with full details in Appendix J due to the space limit. Results for protein-ligand affinity prediction are presented in Table 3. Inspiringly, it reads that our GET with the proposed unified representation achieves significantly better performance compared with the baselines with either single-level representations or hierarchical pooling, no matter the interacting partners are macro molecules (*i.e.* proteins) or small molecules. This confirms the superiority of our method, which comes from a desirable integration of different granularities. Further, to show the flexibility of the proposed unified representation, as mentioned in § 2.1, we add **GET-PS**, which defines the blocks in small molecules as principal subgraphs [33] instead of atoms. GET-PS receives obvious gains over GET since fragments in small molecules usually contribute to interactions as a whole [20].

3.3 Generalization Across Different Domains

Finally, we explore whether our model is able to find universal underlying physics that can generalize across different domains.

[2]We failed to run atom-level DimeNet++ due to the high complexity of the angular SBF.

Data Augmentation from Different Domains We mix the dataset of protein-protein affinity and protein-ligand affinity for training, and evaluate the models on the test set of the two domains, respectively. We also benchmark Schnet and TorchMD, two strong baselines as shown in § 3.2, under the same setting for comparison. We present the results in Figure 3, and include detailed mean and standard deviation in Appendix N. The results demonstrate that our method obtains benefits from the mixed training set on both PPA and LBA, while the baselines receive negative impact in most cases. These phenomena well demonstrate the generalization ability of the proposed GET equipped with the unified representation.

Figure 3: Comparison of different models on the universal learning of molecular interaction affinity.

Zero-Shot Prediction of DNA/RNA-Ligand Affinity A more practical and meaningful, yet also more challenging scenario is ligand (small molecule) binding on nucleic acids (RNA/DNA), the data of which are scarce and expensive to obtain. We use the 149 data points available in PDBbind [62] as the zero-shot test set, and train a model on binding data from other domains in PDBbind (i.e. protein-protein, protein-ligand and RNA/DNA-protein). The results in Table 4 show that GET achieves amazing generalizability across different domains on molecular interaction.

Table 4: Zero-shot performance on DNA/RNA-ligand binding affinity prediction for three runs.

Repr.	Model	Pearson↑	Spearman↑
Atom	SchNet	0.329 ± 0.013	0.336 ± 0.018
	TorchMD	0.150 ± 0.034	0.198 ± 0.043
Block	SchNet	0.235 ± 0.111	0.264 ± 0.080
	TorchMD	0.217 ± 0.059	0.185 ± 0.051
Hierarchical	SchNet	0.179 ± 0.052	0.185 ± 0.087
	TorchMD	0.348 ± 0.047	0.302 ± 0.028
Unified	GET	$\mathbf{0.450 \pm 0.054}$	$\mathbf{0.362 \pm 0.041}$

Both experiments confirm the potential of our model to discover universal underlying principles of molecular interactions capable of generalizing across diverse domains.

4 Conclusion

In this paper, we explore the unified representation of molecules as geometric graphs of sets, which enables all-atom representations while preserving the heuristic building blocks of different molecules. To model the unified representaion, we propose a Generalist Equivariant Transformer (GET) to accommodate matrix-form node features and coordinates with E(3)-equivariance and permutation invariance. Each layer of GET consists of a bilevel attention module, a feed-forward module, and an equivariant layer normalization after each of the previous two modules. Experiments on molecular interactions demonstrate the superiority of learning unified representation with our GET compared to single-level representations and existing baselines. Further explorations on mixing molecular types reveal the ability of our method to learn generalizable molecular interaction mechanisms, which could inspire future research on universal representation learning of molecules.

References

[1] R. F. Alford, A. Leaver-Fay, J. R. Jeliazkov, M. J. O'Meara, F. P. DiMaio, H. Park, M. V. Shapovalov, P. D. Renfrew, V. K. Mulligan, K. Kappel, et al. The rosetta all-atom energy

function for macromolecular modeling and design. *Journal of chemical theory and computation*, 13(6):3031–3048, 2017.

[2] N. Anand and T. Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. *arXiv preprint arXiv:2205.15019*, 2022.

[3] K. Atz, F. Grisoni, and G. Schneider. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12):1023–1032, 2021.

[4] Ž. Avsec, V. Agarwal, D. Visentin, J. R. Ledsam, A. Grabska-Barwinska, K. R. Taylor, Y. Assael, J. Jumper, P. Kohli, and D. R. Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18(10):1196–1203, 2021.

[5] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[6] P. J. Ballester and J. B. Mitchell. A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking. *Bioinformatics*, 26(9):1169–1175, 2010.

[7] T. Bepler and B. Berger. Learning protein sequence embeddings using information from structure. *arXiv preprint arXiv:1902.08661*, 2019.

[8] I. Cohen, Y. Huang, J. Chen, J. Benesty, J. Benesty, J. Chen, Y. Huang, and I. Cohen. Pearson correlation coefficient. *Noise reduction in speech processing*, pages 1–4, 2009.

[9] X. Du, Y. Li, Y.-L. Xia, S.-M. Ai, J. Liang, P. Sang, X.-L. Ji, and S.-Q. Liu. Insights into protein–ligand interactions: mechanisms, models, and methods. *International journal of molecular sciences*, 17(2):144, 2016.

[10] A. A. Elfiky. Anti-hcv, nucleotide inhibitors, repurposing against covid-19. *Life sciences*, 248: 117477, 2020.

[11] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, D. Bhowmik, and B. Rost. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

[12] M. Fey and J. E. Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[13] R. A. Friesner, J. L. Banks, R. B. Murphy, T. A. Halgren, J. J. Klicic, D. T. Mainz, M. P. Repasky, E. H. Knoll, M. Shelley, J. K. Perry, et al. Glide: a new approach for rapid, accurate docking and scoring. 1. method and assessment of docking accuracy. *Journal of medicinal chemistry*, 47 (7):1739–1749, 2004.

[14] P. Gainza, F. Sverrisson, F. Monti, E. Rodola, D. Boscaini, M. Bronstein, and B. Correia. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

[15] J. Gasteiger, S. Giri, J. T. Margraf, and S. Günnemann. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. *arXiv preprint arXiv:2011.14115*, 2020.

[16] J. Gasteiger, J. Groß, and S. Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

[17] Z. Geng, S. Xie, Y. Xia, L. Wu, T. Qin, J. Wang, Y. Zhang, F. Wu, and T.-Y. Liu. De novo molecular generation via connection-aware motif mining. *arXiv preprint arXiv:2302.01129*, 2023.

[18] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[19] M. K. Gilson, J. A. Given, B. L. Bush, and J. A. McCammon. The statistical-thermodynamic basis for computation of binding affinities: a critical review. *Biophysical journal*, 72(3): 1047–1069, 1997.

[20] P. J. Hajduk and J. Greer. A decade of fragment-based drug design: strategic advances and lessons learned. *Nature reviews Drug discovery*, 6(3):211–219, 2007.

[21] J. Hauke and T. Kossowski. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. *Quaestiones geographicae*, 30(2):87–93, 2011.

[22] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[23] P. Hermosilla, M. Schäfer, M. Lang, G. Fackelmann, P. P. Vázquez, B. Kozlíková, M. Krone, T. Ritschel, and T. Ropinski. Intrinsic-extrinsic convolution and pooling for learning on 3d protein structures. *arXiv preprint arXiv:2007.06252*, 2020.

[24] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning*, pages 8867–8887. PMLR, 2022.

[25] W. Huang, J. Han, Y. Rong, T. Xu, F. Sun, and J. Huang. Equivariant graph mechanics networks with constraints. *arXiv preprint arXiv:2203.06442*, 2022.

[26] J. Jiménez, M. Skalic, G. Martinez-Rosell, and G. De Fabritiis. K deep: protein–ligand absolute binding affinity prediction via 3d-convolutional neural networks. *Journal of chemical information and modeling*, 58(2):287–296, 2018.

[27] W. Jin, R. Barzilay, and T. Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.

[28] W. Jin, R. Barzilay, and T. Jaakkola. Antibody-antigen docking and design via hierarchical equivariant refinement. *arXiv preprint arXiv:2207.06616*, 2022.

[29] B. Jing, S. Eismann, P. N. Soni, and R. O. Dror. Equivariant graph neural networks for 3d macromolecular structure. *arXiv preprint arXiv:2106.03843*, 2021.

[30] S. Jones and J. M. Thornton. Principles of protein-protein interactions. *Proceedings of the National Academy of Sciences*, 93(1):13–20, 1996.

[31] P. L. Kastritis, I. H. Moal, H. Hwang, Z. Weng, P. A. Bates, A. M. Bonvin, and J. Janin. A structure-based benchmark for protein–protein binding affinity. *Protein Science*, 20(3):482–491, 2011.

[32] X. Kong, W. Huang, and Y. Liu. Conditional antibody design as 3d equivariant graph translation. *arXiv preprint arXiv:2208.06073*, 2022.

[33] X. Kong, W. Huang, Z. Tan, and Y. Liu. Molecule generation by principal subgraph mining and assembling. *Advances in Neural Information Processing Systems*, 35:2550–2563, 2022.

[34] Y.-L. Liao and T. Smidt. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. *arXiv preprint arXiv:2206.11990*, 2022.

[35] M. Liu, Y. Luo, K. Uchino, K. Maruhashi, and S. Ji. Generating 3d molecules for target protein binding. In *International Conference on Machine Learning*, pages 13912–13924. PMLR, 2022.

[36] X. Liu, Y. Luo, P. Li, S. Song, and J. Peng. Deep geometric representations for modeling effects of mutations on protein-protein binding affinity. *PLoS computational biology*, 17(8):e1009284, 2021.

[37] Z. Liu, Y. Li, L. Han, J. Li, J. Liu, Z. Zhao, W. Nie, Y. Liu, and R. Wang. Pdb-wide collection of binding data: current status of the pdbbind database. *Bioinformatics*, 31(3):405–412, 2015.

[38] S. Luo, J. Guan, J. Ma, and J. Peng. A 3d generative model for structure-based drug design. *Advances in Neural Information Processing Systems*, 34:6229–6239, 2021.

[39] S. Luo, Y. Su, X. Peng, S. Wang, J. Peng, and J. Ma. Antigen-specific antibody design and optimization with diffusion-based generative models. *bioRxiv*, pages 2022–07, 2022.

[40] S. Luo, Y. Su, Z. Wu, C. Su, J. Peng, and J. Ma. Rotamer density estimator is an unsupervised learner of the effect of mutations on protein-protein interaction. *bioRxiv*, pages 2023–02, 2023.

[41] P. Notin, M. Dias, J. Frazer, J. M. Hurtado, A. N. Gomez, D. Marks, and Y. Gal. Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. In *International Conference on Machine Learning*, pages 16990–17017. PMLR, 2022.

[42] H. Öztürk, A. Özgür, and E. Ozkirimli. Deepdta: deep drug–target binding affinity prediction. *Bioinformatics*, 34(17):i821–i829, 2018.

[43] X. Peng, S. Luo, J. Guan, Q. Xie, J. Peng, and J. Ma. Pocket2mol: Efficient molecular sampling based on 3d protein pockets. In *International Conference on Machine Learning*, pages 17644–17655. PMLR, 2022.

[44] M. Ragoza, J. Hochuli, E. Idrobo, J. Sunseri, and D. R. Koes. Protein–ligand scoring with convolutional neural networks. *Journal of chemical information and modeling*, 57(4):942–957, 2017.

[45] R. Rao, N. Bhattacharya, N. Thomas, Y. Duan, P. Chen, J. Canny, P. Abbeel, and Y. Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

[46] J. S. Richardson. The anatomy and taxonomy of protein structure. *Advances in protein chemistry*, 34:167–339, 1981.

[47] N. Sapoval, A. Aghazadeh, M. G. Nute, D. A. Antunes, A. Balaji, R. Baraniuk, C. Barberan, R. Dannenfelser, C. Dun, M. Edrisi, et al. Current progress and open challenges for applying deep learning across the biosciences. *Nature Communications*, 13(1):1728, 2022.

[48] V. G. Satorras, E. Hoogeboom, and M. Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021.

[49] K. Schütt, P.-J. Kindermans, H. E. Sauceda Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

[50] C. Shi, C. Wang, J. Lu, B. Zhong, and J. Tang. Protein sequence and structure co-design with equivariant translation. *arXiv preprint arXiv:2210.08761*, 2022.

[51] V. R. Somnath, C. Bunne, and A. Krause. Multi-scale representation learning on proteins. *Advances in Neural Information Processing Systems*, 34:25244–25255, 2021.

[52] H. Stärk, D. Beaini, G. Corso, P. Tossou, C. Dallago, S. Günnemann, and P. Liò. 3d infomax improves gnns for molecular property prediction. In *International Conference on Machine Learning*, pages 20479–20502. PMLR, 2022.

[53] M. Steinegger and J. Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

[54] P. Thölke and G. De Fabritiis. Torchmd-net: equivariant transformers for neural network based molecular potentials. *arXiv preprint arXiv:2202.02541*, 2022.

[55] J. Tomasi and M. Persico. Molecular interactions in solution: an overview of methods based on continuous distributions of the solvent. *Chemical Reviews*, 94(7):2027–2094, 1994.

[56] R. J. Townshend, M. Vögele, P. Suriana, A. Derry, A. Powers, Y. Laloudakis, S. Balachandar, B. Jing, B. Anderson, S. Eismann, et al. Atom3d: Tasks on molecules in three dimensions. *arXiv preprint arXiv:2012.04035*, 2020.

[57] R. Tran, J. Lan, M. Shuaibi, B. M. Wood, S. Goyal, A. Das, J. Heras-Domingo, A. Kolluru, A. Rizvi, N. Shoghi, et al. The open catalyst 2022 (oc22) dataset and challenges for oxide electrocatalysts. *ACS Catalysis*, 13(5):3066–3084, 2023.

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

[59] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[60] T. Vreven, I. H. Moal, A. Vangone, B. G. Pierce, P. L. Kastritis, M. Torchala, R. Chaleil, B. Jiménez-García, P. A. Bates, J. Fernandez-Recio, et al. Updates to the integrated protein–protein interaction benchmarks: docking benchmark version 5 and affinity benchmark version 2. *Journal of molecular biology*, 427(19):3031–3041, 2015.

[61] L. Wang, H. Liu, Y. Liu, J. Kurtin, and S. Ji. Learning protein representations via complete 3d graph networks. *arXiv preprint arXiv:2207.12600*, 2022.

[62] R. Wang, X. Fang, Y. Lu, and S. Wang. The pdbbind database: Collection of binding affinities for protein- ligand complexes with known three-dimensional structures. *Journal of medicinal chemistry*, 47(12):2977–2980, 2004.

[63] J. D. Watson and F. H. Crick. The structure of dna. In *Cold Spring Harbor symposia on quantitative biology*, volume 18, pages 123–131. Cold Spring Harbor Laboratory Press, 1953.

[64] M. Xu, L. Yu, Y. Song, C. Shi, S. Ermon, and J. Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[65] S. Zaidi, M. Schaarschmidt, J. Martens, H. Kim, Y. W. Teh, A. Sanchez-Gonzalez, P. Battaglia, R. Pascanu, and J. Godwin. Pre-training via denoising for molecular property prediction. *arXiv preprint arXiv:2206.00133*, 2022.

[66] Y. Zhang, H. Cai, C. Shi, B. Zhong, and J. Tang. E3bind: An end-to-end equivariant network for protein-ligand docking. *arXiv preprint arXiv:2210.06069*, 2022.

A Related Work

Molecular Interaction and Representation Various types of molecules across different domains [9, 10, 30] can form interactions, the strength of which are usually measured by the energy gap between the unbound and bound states of the molecules (i.e. affinity) [19]. We primarily investigate interactions between two proteins [30] and between a protein and a small molecule [9], both of which are widely explored in the machine learning community [32, 40, 39, 51, 52, 61]. Furthermore, we embark on a pioneering effort to involve RNA/DNAs, which is a challenging endeavor due to the limited availability of such data. Small molecules are usually represented by graphs where nodes are atoms [3, 24, 64, 65], but there are also explorations on subgraph-level decomposition of molecules by mining motifs [17, 27, 33]. Proteins are built upon residues, which are predefined sets of atoms [46], and thus have mainly two categories of representations according to the granularity of graph nodes: atom-level and residue-level. Atom-level representations, as the name suggests, decompose proteins into single atoms [56] and discard the hierarchy of proteins. Residue-level representations either exert pooling on the atoms [28], or directly use residue-specific features [2, 50, 51, 61] which is limited to proteins. Similarly, RNA/DNAs also have atom-level and nucleobase-level representations [4, 63]. Despite the differences in building blocks, the basic units (i.e. atoms) are shared across different molecular domains, and so do the fundamental interaction physics. Therefore, it is valuable to construct a unified representation for different molecular domains, which is explored in this paper.

Equivariant Network Equivariant networks integrate the symmetry of 3D world, namely E(3)-equivariance, into the models, and thus are widely used in geometric learning [16, 48, 54]. Our work is inspired by multi-channel equivariant graph neural networks [25, 33] which assign each node with a coordinate matrix. However, they require a fixed number of channels (i.e. constant number of rows in the coordinate matrix) and lack invariance w.r.t to the permutations of the coordinates, which limits their application here as each building block is an unordered set of atoms with variable size. Moreover, the node features are still limited to single vector form, which is unable to accommodate all-atom representations in single blocks. In contrast, our proposed model is designed to handle geometric graphs of sets where each node contains an unordered set of 3D instances with a different size, which fits perfectly with the concept of building blocks in molecules.

B Analysis

We conduct ablation study by removing the following modules: the layer normalization (w/o LN); the equivariant normalization on coordinates in the LN (w/o equivLN); the reflection of rescaling information in hidden features in Eq. 17 (w/o EmbedScale); the equivariant feed-forward network (w/o FFN); both LN and FFN (w/o LN & FFN). The results are presented in Table 5.

The ablations of the modules reveal following regularities: (1) Removing either the entire layer normalization or only the equivariant normalization on coordinates introduces instability in training, which not only leads to higher variance across different experiments, but also induces adverse impacts in some tasks like PPA; (2) Not reflecting the rescaling information in the hidden features has an adverse effect on the performance as the scale of the coordinates also carries essential information for learning the interaction physics; (3) The removal of the equivariant feed-forword module incurs detriment to the overall performance, indicating the necessity of the FFN to encourage intra-block geometrical communications between atoms.

Table 5: Ablation study of each module in our proposed Generalist Equivariant Transformer (GET), where LN and FFN are abbreviations for LayerNorm and Feed-Forward Network, respectively. The best results are marked in bold and the second best are underlined.

Repr.	Model	PPA-All		LBA	
		Pearson↑	Spearman↑	Pearson↑	Spearman↑
Unified	GET-mix	$\mathbf{0.519 \pm 0.004}$	$\mathbf{0.537 \pm 0.003}$	$\mathbf{0.622 \pm 0.006}$	$\mathbf{0.615 \pm 0.008}$
	GET	$\underline{0.514 \pm 0.011}$	$\underline{0.533 \pm 0.011}$	$\underline{0.620 \pm 0.004}$	$\underline{0.611 \pm 0.003}$
	w/o LN	0.366 ± 0.024	0.426 ± 0.032	0.589 ± 0.007	0.593 ± 0.008
	w/o equivLN	0.368 ± 0.025	0.426 ± 0.032	0.591 ± 0.008	0.597 ± 0.009
	w/o EmbedScale	0.490 ± 0.027	0.507 ± 0.030	0.591 ± 0.002	0.586 ± 0.003
	w/o FFN	0.494 ± 0.010	0.510 ± 0.010	0.593 ± 0.008	0.601 ± 0.012
	w/o LN & FFN	0.360 ± 0.018	0.423 ± 0.024	0.589 ± 0.009	0.596 ± 0.008

C Atom Position Code

Certain types of molecules have conventional position codes to distinguish different status of the atoms in the same block. For example, in the protein domain, where building blocks are residues, each atom in a residue is assigned a position code $(\alpha, \beta, \gamma, \delta, \varepsilon, \zeta, \eta, ...)$ according to the number of chemical bonds between it and the alpha carbon (i.e. C_α). As these position codes provide meaningful heuristics of intra-block geometry, we also include them as a component of the embedding. For other types of molecules without such position codes (e.g. small molecules), we assign a $[\mathrm{BLANK}]$ type for positional embedding.

D Scheme of the Generalist Equivariant Transformer

We depict the overall workflow and the details of the equivariant bilevel attention module in Figure 4.

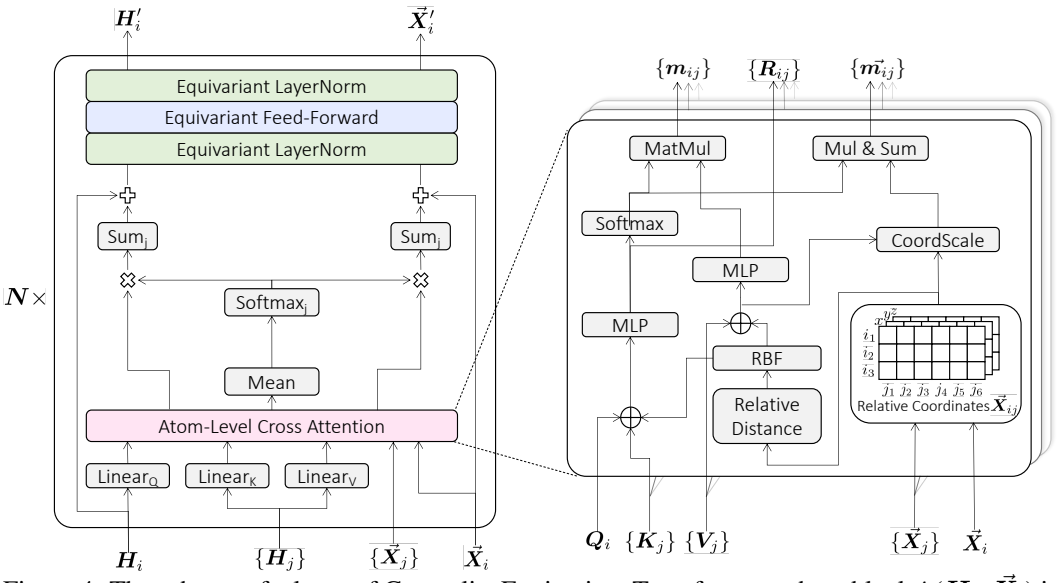

Figure 4: The scheme of a layer of Generalist Equivariant Transformer, where block i ($\boldsymbol{H}_i$, $\vec{\boldsymbol{X}}_i$) is updated by its neighbors ($\{\boldsymbol{H}_j\}$, $\{\vec{\boldsymbol{X}}_j\}$, $j \in \mathcal{N}_b(i)$). $\times$, $+$, and $\oplus$ denote multiplication, addition and concatenation, respectively. (Left) The overall workflow of a layer and details of the block-level attention. (Right) The details of the atom-level cross attention. GET is composed of N such layers.

E Proof of E(3)-Equivariance and Intra-Block Permutation Invariance

Theorem E.1 (E(3)-Equivariance and Intra-Block Permutation Invariance)**.** *Denote the proposed Equivariant Transformer as* $\{\boldsymbol{H}'_i, \vec{\boldsymbol{X}}'_i\} = \mathrm{GET}(\{\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i\})$*, then it conforms to E(3)-Equivariance and Intra-Block Permutation Invariance. Namely,* $\forall g \in E(3), \forall \{\pi_i \in S_{n_i} | 1 \le i \le B\}$*, where* B *is the number of blocks in the input and* S_{n_i} *includes all permutations on* n_i *elements, we have* $\{\pi_i \cdot \boldsymbol{H}'_i, \pi_i \cdot g \cdot \vec{\boldsymbol{X}}'_i\} = \mathrm{GET}(\{\pi_i \cdot \boldsymbol{H}_i, \pi_i \cdot g \cdot \vec{\boldsymbol{X}}_i\})$.

The Generalist Equivariant Transformer (GET) is the cascading of the three types of modules: bilevel attention, feed-forward network, and layer normalization. Further, the E(3)-equivariance and the intra-block permutation invariance are disentangled. Therefore, the proof of its E(3)-equivariance and its invariance respect to the intra-block permutations can be decomposed into proof of these two properties on each module, which we present below.

E.1 Proof of E(3)-Equivariance

First we give the definition of E(3)-equivariance as follows:

Definition E.2 (E(3)-equivariance)**.** A function $\phi : \mathbb{X} \to \mathbb{Y}$ conforms E(3)-equivariance if $\forall g \in \mathrm{E}(3)$, the equation $\rho_{\mathbb{Y}}(g)\boldsymbol{y} = \phi(\rho_{\mathbb{X}}(g)\boldsymbol{x})$ holds true, where $\rho_{\mathbb{X}}$ and $\rho_{\mathbb{Y}}$ instantiate g in $\mathbb{X}$ and $\mathbb{Y}$, respectively. A special case is E(3)-invariance where $\rho_{\mathbb{Y}}$ constantly outputs identity transformation (i.e. $\rho_{\mathbb{Y}}(g) \equiv I$).

Given $g \in \mathrm{E}(3)$ and $\vec{\boldsymbol{x}} \in \mathbb{R}^3$, we can instantiate g as $g \cdot \vec{\boldsymbol{x}} \coloneqq \boldsymbol{Q}\vec{\boldsymbol{x}} + \vec{\boldsymbol{t}}$, where $\boldsymbol{Q} \in \mathbb{R}^{3\times 3}$ is an orthogonal matrix and $\vec{\boldsymbol{t}} \in \mathbb{R}^3$ is a translation vector. Implementing g on a coordinate matrix $\vec{\boldsymbol{X}} \in \mathbb{R}^{n\times 3}$ means transforming each coordinate (i.e. each row) with g.

Then we prove the E(3)-equivariance of each module in GET as follows:

Lemma E.3. *Denote the bilevel attention module as $\{\boldsymbol{H}'_i, \vec{\boldsymbol{X}}'_i\} = \mathrm{Att}(\{\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i\})$, then it is E(3)-equivariant. Namely, $\forall g \in$ E(3), we have $\{\boldsymbol{H}'_i, g \cdot \vec{\boldsymbol{X}}'_i\} = \mathrm{Att}(\{\boldsymbol{H}_i, g \cdot \vec{\boldsymbol{X}}_i\})$.*

Proof. The key to the proof of Lemma E.3 is to prove that the propagation in Eq. 2-10 is E(3)-invariant on $\boldsymbol{H}_i$ and E(3)-equivariant on $\vec{\boldsymbol{X}}_i$. Obviously, the correlation $\boldsymbol{R}_{ij}$ between block i and block j in Eq. 3 is E(3)-invariant because all the inputs, that is, the query, the key, and the distance matrices, are not affected by the geometric transformation g. Therefore, we can immediately derive that the atom-level cross attention $\boldsymbol{\alpha}_{ij}$ in Eq. 4 is E(3)-invariant. Similarly, the block-level attention β_{ij} in Eq. 6 is E(3)-invariant because it only operates on $\boldsymbol{r}_{ij}$ in Eq. 5 which aggregates $\boldsymbol{\alpha}_{ij}$ and the edge feature. Finally, we can derive the E(3)-invariance on $\boldsymbol{H}$ and the E(3)-equivariance on $\vec{\boldsymbol{X}}$:

$$
\begin{aligned}
\boldsymbol{H}'_i[p] &= \boldsymbol{H}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij}\phi_m(\boldsymbol{m}_{ij,p}), \\
&= \boldsymbol{H}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij}\phi_m(\boldsymbol{\alpha}_{ij}[p] \cdot \phi_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p]))), \\
g \cdot \vec{\boldsymbol{X}}'_i[p] &= g \cdot \left(\vec{\boldsymbol{X}}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \left(\vec{\boldsymbol{m}}_{ij,p} \odot \sigma_m(\boldsymbol{m}_{ij,p})\right) \right) \\
&= g \cdot \left(\vec{\boldsymbol{X}}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \left(\boldsymbol{\alpha}_{ij}[p] \cdot (\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p]))) \odot \sigma_m(\boldsymbol{m}_{ij,p})\right) \right) \\
&= \boldsymbol{Q} \left(\vec{\boldsymbol{X}}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \left(\boldsymbol{\alpha}_{ij}[p] \cdot (\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p]))) \odot \sigma_m(\boldsymbol{m}_{ij,p})\right) \right) + \vec{\boldsymbol{t}} \\
&= (\boldsymbol{Q}\vec{\boldsymbol{X}}_i[p] + \vec{\boldsymbol{t}}) + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \\
&\quad \left(\boldsymbol{\alpha}_{ij}[p] \cdot \left(\begin{bmatrix} \boldsymbol{Q}(\vec{\boldsymbol{X}}_i[p] - \vec{\boldsymbol{X}}_j[1]) \\ \vdots \\ \boldsymbol{Q}(\vec{\boldsymbol{X}}_i[p] - \vec{\boldsymbol{X}}_j[n_j]) \end{bmatrix} \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p])) \right) \odot \sigma_m(\boldsymbol{m}_{ij,p}) \right) \\
&= (\boldsymbol{Q}\vec{\boldsymbol{X}}_i[p] + \vec{\boldsymbol{t}}) + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \\
&\quad \left(\boldsymbol{\alpha}_{ij}[p] \cdot \left(\begin{bmatrix} \boldsymbol{Q}\vec{\boldsymbol{X}}_i[p] + \vec{\boldsymbol{t}} - (\boldsymbol{Q}\vec{\boldsymbol{X}}_j[1] + \vec{\boldsymbol{t}}) \\ \vdots \\ \boldsymbol{Q}\vec{\boldsymbol{X}}_i[p] + \vec{\boldsymbol{t}} - (\boldsymbol{Q}\vec{\boldsymbol{X}}_j[n_j] + \vec{\boldsymbol{t}}) \end{bmatrix} \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p])) \right) \odot \sigma_m(\boldsymbol{m}_{ij,p}) \right) \\
&= g \cdot \vec{\boldsymbol{X}}_i[p] + \sum_{j\in\mathcal{N}(i)} \beta_{ij} \\
&\quad \left(\boldsymbol{\alpha}_{ij}[p] \cdot \left(\begin{bmatrix} g \cdot \vec{\boldsymbol{X}}_i[p] - g \cdot \vec{\boldsymbol{X}}_j[1] \\ \vdots \\ g \cdot \vec{\boldsymbol{X}}_i[p] - g \cdot \vec{\boldsymbol{X}}_j[n_j] \end{bmatrix} \odot \sigma_v(\boldsymbol{V}_j \parallel \mathrm{RBF}(\boldsymbol{D}_{ij}[p])) \right) \odot \sigma_m(\boldsymbol{m}_{ij,p}) \right),
\end{aligned}
$$

which concludes the proof of Lemma E.3. □

Lemma E.4. *Denote the equivariant feed-forward network as as $\{\boldsymbol{H}'_i, \vec{\boldsymbol{X}}'_i\} = \mathrm{EFFN}(\{\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i\})$, then it is E(3)-equivariant. Namely, $\forall g \in$ E(3), we have $\{\boldsymbol{H}'_i, g \cdot \vec{\boldsymbol{X}}'_i\} = \mathrm{EFFN}(\{\boldsymbol{H}_i, g \cdot \vec{\boldsymbol{X}}_i\})$.*

Proof. The proof of Lemma E.4 focuses on the single-atom updates in Eq. 11-14. First, it is easy to obtain the E(3)-equivariance of the centroid in Eq. 11:

$$g \cdot \vec{\boldsymbol{x}}_c = g \cdot \text{centroid}(\vec{\boldsymbol{X}}_i) = \text{centroid}(g \cdot \vec{\boldsymbol{X}}_i).$$

Then we have the following equation on the relative coordinate $\Delta\vec{\boldsymbol{x}}$ in Eq. 12:

$$\boldsymbol{Q}\Delta\vec{\boldsymbol{x}} = (\boldsymbol{Q}\vec{\boldsymbol{x}} + \vec{\boldsymbol{t}}) - (\boldsymbol{Q}\vec{\boldsymbol{x}}_c + \vec{\boldsymbol{t}}) = g \cdot \vec{\boldsymbol{x}} - g \cdot \vec{\boldsymbol{x}}_c.$$

We can immediately obtain the E(3)-invariance of $\boldsymbol{r}$ in Eq. 12:

$$\boldsymbol{r} = \text{RBF}(\|\boldsymbol{Q}\Delta\vec{\boldsymbol{x}}\|_2) = \text{RBF}(\sqrt{(\boldsymbol{Q}\Delta\vec{\boldsymbol{x}})^\top(\boldsymbol{Q}\Delta\vec{\boldsymbol{x}})}) = \text{RBF}(\sqrt{\Delta\vec{\boldsymbol{x}}^\top\boldsymbol{Q}^\top\boldsymbol{Q}\Delta\vec{\boldsymbol{x}}}) = \text{RBF}(\|\Delta\vec{\boldsymbol{x}}\|_2).$$

Finally we can derive the E(3)-invariance on $\boldsymbol{h}$ and the E(3)-equivariance on $\vec{\boldsymbol{x}}$:

$$\begin{aligned}
\boldsymbol{h}' &= \boldsymbol{h} + \phi_h(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}), \\
g \cdot \vec{\boldsymbol{x}}' &= g \cdot (\vec{\boldsymbol{x}} + \Delta\vec{\boldsymbol{x}}\phi_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r})) \\
&= \boldsymbol{Q}(\vec{\boldsymbol{x}} + \Delta\vec{\boldsymbol{x}}\phi_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r})) + \vec{\boldsymbol{t}} \\
&= \boldsymbol{Q}\vec{\boldsymbol{x}} + \vec{\boldsymbol{t}} + \boldsymbol{Q}\Delta\vec{\boldsymbol{x}}\phi_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}) \\
&= g \cdot \vec{\boldsymbol{x}} + (g \cdot \vec{\boldsymbol{x}} - g \cdot \vec{\boldsymbol{x}}_c)\phi_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}) \\
&= g \cdot \vec{\boldsymbol{x}} + (g \cdot \vec{\boldsymbol{x}} - \text{centroid}(g \cdot \vec{\boldsymbol{X}}_i))\phi_x(\boldsymbol{h}, \boldsymbol{h}_c, \boldsymbol{r}),
\end{aligned}$$

which concludes the proof of Lemma E.4 □

Lemma E.5. *Denote the equivariant layer normalization as as* $\{\boldsymbol{H}_i', \vec{\boldsymbol{X}}_i'\} = \text{ELN}(\{\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i\})$*, then it is E(3)-equivariant. Namely,* $\forall g \in$ *E(3), we have* $\{\boldsymbol{H}_i', g \cdot \vec{\boldsymbol{X}}_i'\} = \text{ELN}(\{\boldsymbol{H}_i, g \cdot \vec{\boldsymbol{X}}_i\})$.

Proof. Since the layer normalization is implemented on the atom level, namely each row of the coordinate matrix in a node, we again only need to concentrate on the single-atom normalization in Eq. 15-16. The key points lie in the E(3)-equivariance of $\mathbb{E}[\vec{\mathbf{X}}]$ and the E(3)-invariance of $\text{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]]$. The first one is obvious because $\mathbb{E}[\vec{\mathbf{X}}]$ is the centroid of the coordinates of all atoms:

$$g \cdot \mathbb{E}[\vec{\mathbf{X}}] = g \cdot \text{centroid}(\vec{\mathbf{X}}) = \text{centroid}(g \cdot \vec{\mathbf{X}}) = \mathbb{E}[g \cdot \vec{\mathbf{X}}].$$

Suppose there are N atoms in total, then we can prove the E(3)-invariance of the variance as follows:

$$\begin{aligned}
\text{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]] &= \frac{\sum_{i=1}^N (x_i - \bar{x})^2 + \sum_{i=1}^N (y_i - \bar{y})^2 + \sum_{i=1}^N (z_i - \bar{z})^2}{3N} \\
&= \frac{\sum_{i=1}^N [(x_i - \bar{x})^2 + (y_i - \bar{y})^2 + (z_i - \bar{z})^2]}{3N} \\
&= \frac{\sum_{i=1}^N (\vec{\boldsymbol{x}}_i - \mathbb{E}[\vec{\mathbf{X}}])^\top (\vec{\boldsymbol{x}}_i - \mathbb{E}[\vec{\mathbf{X}}])}{3N} \\
&= \frac{\sum_{i=1}^N (\vec{\boldsymbol{x}}_i - \mathbb{E}[\vec{\mathbf{X}}])^\top \boldsymbol{Q}^\top \boldsymbol{Q} (\vec{\boldsymbol{x}}_i - \mathbb{E}[\vec{\mathbf{X}}])}{3N} \\
&= \frac{\sum_{i=1}^N (\boldsymbol{Q}\vec{\boldsymbol{x}}_i - \boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}])^\top (\boldsymbol{Q}\vec{\boldsymbol{x}}_i - \boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}])}{3N} \\
&= \frac{\sum_{i=1}^N (g \cdot \vec{\boldsymbol{x}}_i - g \cdot \mathbb{E}[\vec{\mathbf{X}}])^\top (g \cdot \vec{\boldsymbol{x}}_i - g \cdot \mathbb{E}[\vec{\mathbf{X}}])}{3N} \\
&= \frac{\sum_{i=1}^N (g \cdot \vec{\boldsymbol{x}}_i - \mathbb{E}[g \cdot \vec{\mathbf{X}}])^\top (g \cdot \vec{\boldsymbol{x}}_i - \mathbb{E}[g \cdot \vec{\mathbf{X}}])}{3N} \\
&= \text{Var}[g \cdot \vec{\mathbf{X}} - \mathbb{E}[g \cdot \vec{\mathbf{X}}]].
\end{aligned}$$

Therefore, we can finally derive the E(3)-invariance on $\boldsymbol{h}$ and the E(3)-equivariance on $\vec{\boldsymbol{x}}$ in Eq. 15-16:

$$
\begin{aligned}
\boldsymbol{h} &= \frac{\boldsymbol{h} - \mathbb{E}[\boldsymbol{h}]}{\sqrt{\mathrm{Var}[\boldsymbol{h}]}} \cdot \boldsymbol{\gamma} + \boldsymbol{\beta}, \\
g \cdot \vec{\boldsymbol{x}} = g \cdot (\frac{\vec{\boldsymbol{x}} - \mathbb{E}[\vec{\mathbf{X}}]}{\sqrt{\mathrm{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]]}} \cdot \sigma + \mathbb{E}[\vec{\mathbf{X}}]) &= \frac{\boldsymbol{Q}\vec{\boldsymbol{x}} - \boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}]}{\sqrt{\mathrm{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]]}} \cdot \sigma + \boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}] + \vec{\boldsymbol{t}} \\
&= \frac{\boldsymbol{Q}\vec{\boldsymbol{x}} + \vec{\boldsymbol{t}} - (\boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}] + \vec{\boldsymbol{t}})}{\sqrt{\mathrm{Var}[\vec{\mathbf{X}} - \mathbb{E}[\vec{\mathbf{X}}]]}} \cdot \sigma + (\boldsymbol{Q}\mathbb{E}[\vec{\mathbf{X}}] + \vec{\boldsymbol{t}}) = \frac{g \cdot \vec{\boldsymbol{x}} - g \cdot \mathbb{E}[\vec{\mathbf{X}}]}{\sqrt{\mathrm{Var}[g \cdot \vec{\mathbf{X}} - \mathbb{E}[g \cdot \vec{\mathbf{X}}]]}} \cdot \sigma + g \cdot \mathbb{E}[\vec{\mathbf{X}}] \\
&= \frac{g \cdot \vec{\boldsymbol{x}} - \mathbb{E}[g \cdot \vec{\mathbf{X}}]}{\sqrt{\mathrm{Var}[g \cdot \vec{\mathbf{X}} - \mathbb{E}[g \cdot \vec{\mathbf{X}}]]}} \cdot \sigma + \mathbb{E}[g \cdot \vec{\mathbf{X}}],
\end{aligned}
$$

which concludes the proof of Lemma E.5. □

With Lemma E.3-E.5 at hand, it is obvious to deduce the E(3)-equivariance of the GET layer.

E.2 Proof of Intra-Block Permuation Invariance

Obviously, the feed-forward network and the layer normalization are invariant to intra-block permutations because they are implemented on single atoms and the only incorporated multi-atom operation is averaging, which is invariant to the permutations. Therefore, the proof narrows down to the intra-block permutation invariance of the bilevel attention module.

Lemma E.6. *Denote the bilevel attention module as $\{\boldsymbol{H}_i', \vec{\boldsymbol{X}}_i'\} = \mathrm{Att}(\{\boldsymbol{H}_i, \vec{\boldsymbol{X}}_i\})$, then it conforms to intra-block permutation invariance. Namely, $\forall\{\pi_i \in S_{n_i} | 1 \leq i \leq B\}$, where B is the number of blocks in the input and S_{n_i} includes all permutations on n_i elements, we have $\{\pi_i \cdot \boldsymbol{H}_i', \pi_i \cdot \vec{\boldsymbol{X}}_i'\} = \mathrm{Att}(\{\pi_i \cdot \boldsymbol{H}_i, \pi_i \cdot \vec{\boldsymbol{X}}_i\})$.*

Proof. Denote the the permutation of block i as π_i, then it can be instantiated as the multiplication of a series of elementary row-switching matrices $\boldsymbol{P}_i = \boldsymbol{P}_i^{(m_i)}\boldsymbol{P}_i^{(m_i-1)}\dots\boldsymbol{P}_i^{(1)}$. For example, we have $\pi_i \cdot \boldsymbol{H}_i = \boldsymbol{P}_i\boldsymbol{H}_i$, $\pi_i \cdot \vec{\boldsymbol{X}}_i = \boldsymbol{P}_i\vec{\boldsymbol{X}}_i$. Here we first prove an elegant property of $\boldsymbol{P}_i$, which we will use in the later proof:

$$
\begin{aligned}
\boldsymbol{P}_i^\top\boldsymbol{P}_i &= (\boldsymbol{P}_i^{(m_i)}\boldsymbol{P}_i^{(m_i-1)}\dots\boldsymbol{P}_i^{(1)})^\top(\boldsymbol{P}_i^{(m_i)}\boldsymbol{P}_i^{(m_i-1)}\dots\boldsymbol{P}_i^{(1)}) \\
&= {\boldsymbol{P}_i^{(1)}}^\top\dots{\boldsymbol{P}_i^{(m_i-1)}}^\top{\boldsymbol{P}_i^{(m_i)}}^\top\boldsymbol{P}_i^{(m_i)}\boldsymbol{P}_i^{(m_i-1)}\dots\boldsymbol{P}_i^{(1)} \\
&= {\boldsymbol{P}_i^{(1)}}^\top\dots{\boldsymbol{P}_i^{(m_i-1)}}^\top\boldsymbol{I}\boldsymbol{P}_i^{(m_i-1)}\dots\boldsymbol{P}_i^{(1)} \\
&= \dots \\
&= \boldsymbol{I}
\end{aligned}
$$

Given arbitrary permutations on each block, we have the permutated query, key, and value matrics:

$$
\begin{aligned}
\boldsymbol{P}_i\boldsymbol{Q}_i &= \boldsymbol{P}_i\boldsymbol{H}_i\boldsymbol{W}_Q = (\pi_i \cdot \boldsymbol{H}_i)\boldsymbol{W}_Q, \\
\boldsymbol{P}_i\boldsymbol{K}_i &= \boldsymbol{P}_i\boldsymbol{H}_i\boldsymbol{W}_K = (\pi_i \cdot \boldsymbol{H}_i)\boldsymbol{W}_K, \\
\boldsymbol{P}_i\boldsymbol{V}_i &= \boldsymbol{P}_i\boldsymbol{H}_i\boldsymbol{W}_V = (\pi_i \cdot \boldsymbol{H}_i)\boldsymbol{W}_V.
\end{aligned}
$$

The distance matrix $\boldsymbol{D}_{ij}$ is also permutated as $\boldsymbol{P}_i\boldsymbol{D}_{ij}\boldsymbol{P}_j^\top$. Therefore, the atom-level attention $\boldsymbol{\alpha}_{ij}$ in Eq. 4 is also permutated as $\boldsymbol{P}_i\boldsymbol{\alpha}_{ij}\boldsymbol{P}_j^\top$, and the messages in Eq. 7-8 are permutated as:

$$\boldsymbol{P}_i\boldsymbol{m}_{ij} = \boldsymbol{P}_i \begin{bmatrix} \boldsymbol{\alpha}_{ij}[1] \cdot \phi_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[1])) \\ \vdots \\ \boldsymbol{\alpha}_{ij}[n_i] \cdot \phi_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[n_i])) \end{bmatrix} = \boldsymbol{P}_i \begin{bmatrix} \boldsymbol{\alpha}_{ij}[1]\boldsymbol{P}_j^\top\boldsymbol{P}_j\phi_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[1])) \\ \vdots \\ \boldsymbol{\alpha}_{ij}[n_i]\boldsymbol{P}_j^\top\boldsymbol{P}_j\phi_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[n_i])) \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{P}_i\vec{\boldsymbol{m}}_{ij} &= \boldsymbol{P}_i \begin{bmatrix} \boldsymbol{\alpha}_{ij}[1] \cdot \left(\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[1])\right) \\ \vdots \\ \boldsymbol{\alpha}_{ij}[n_i] \cdot \left(\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[n_i])\right) \end{bmatrix} \\ &= \boldsymbol{P}_i \begin{bmatrix} \boldsymbol{\alpha}_{ij}[1]\boldsymbol{P}_j^\top\boldsymbol{P}_j \left(\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[1])\right) \\ \vdots \\ \boldsymbol{\alpha}_{ij}[n_i]\boldsymbol{P}_j^\top\boldsymbol{P}_j \left(\vec{\boldsymbol{X}}_{ij}[p] \odot \sigma_v(\boldsymbol{V}_j \| \mathrm{RBF}(\boldsymbol{D}_{ij}[n_i])\right) \end{bmatrix}, \end{aligned}$$

The block-level attention β_{ij} in Eq. 6 remains unchanged as the average of $\boldsymbol{R}_{ij}$ in obtaining $\boldsymbol{r}_{ij}$ eliminates the effect of permutations. Finally, we can derive the intra-block permutation invariance as follows:

$$\boldsymbol{P}_i\boldsymbol{H}_i' = \boldsymbol{P}_i\left(\boldsymbol{H}_i + \sum_{j\in\mathcal{N}(i)} \beta_{ij}\phi_m(\boldsymbol{m}_{ij})\right) = \boldsymbol{P}_i\boldsymbol{H}_i + \sum_{j\in\mathcal{N}(i)} \beta_{ij}\phi_m(\boldsymbol{P}_i\boldsymbol{m}_{ij}),$$

$$\boldsymbol{P}_i\vec{\boldsymbol{X}}_i' = \boldsymbol{P}_i\left(\vec{\boldsymbol{X}}_i + \sum_{j\in\mathcal{N}(i)} \beta_{ij}\vec{\boldsymbol{m}}_{ij} \odot \sigma_m(\vec{\boldsymbol{m}}_{ij})\right) = \boldsymbol{P}_i\vec{\boldsymbol{X}}_i + \sum_{j\in\mathcal{N}(i)} \beta_{ij}(\boldsymbol{P}_i\vec{\boldsymbol{m}}_{ij}) \odot \sigma_m(\boldsymbol{P}_i\vec{\boldsymbol{m}}_{ij})$$

which concludes Lemma E.6. □

F Complexity Analysis

To discuss the scalability of the model, we additionally provide the complexity analysis as follows. The main complexity lies in the attention-based message passing module. Suppose block i and block j have n_i and n_j atoms, respectively. Since the attention module implements bipartite cross attention between block pairs, there are a total of n_in_j attention edges between block i and block j. Therefore, the exact complexity should be $O(\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}(i)} n_in_j)$, where $\mathcal{V}$ includes all nodes and $\mathcal{N}(i)$ includes all neighbors of block i. Since we use K nearest neighbors to construct graphs in block level, we have $\mathcal{N}(i) \le K$. Denote the maximum number of atoms in a single block is C(in natural proteins we have $C = 14$), we have $n_i \le C$. Therefore, we have $\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}(i)} n_in_j \le \sum_{i\in\mathcal{V}} KC^2 = KC^2|\mathcal{V}|$, namely, the complexity should be bounded by $O(KC^2|\mathcal{V}|)$, which is linear to the number of blocks in the graph. A linear complexity means the algorithm should be easy to scale to larger molecular systems.

Practically, the complexity can be further optimized by selecting only k nearest neighbors of each atom in message passing between block i and block j. With this sparse attention, the complexity is $O(\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}(i)} kn_i) \le O(kKN)$, where N is the total number of atoms.

G Ligand Efficacy Prediction

We additionally provide the evaluation results on Ligand Efficacy Prediction (LEP). This task requires identifying a given ligand as the "activator" or the "inactivator" of a functional protein. Specifically, given the two complexes where the ligand interacts with the active and the inactive conformation of the protein respectively, the models need to distinguish which one is more favorable. To this end, we first obtain the graph-level representations of the two complexes. Then we concatenate the two representations to do a binary classification. We use two metrics for evaluation: the area under the receiver operating characteristic (**AUROC**) and the area under precision-recall curve (**AUPRC**).

Dataset We follow the LEP dataset and its splits in the Atom3D benchmark [56], which includes 27 functional proteins and 527 ligands known as activator or inactivator to a certain protein. The active and the inactive complexes are generated by Glide [13]. The splits of the training, the validation, and the test sets are based on the functional proteins to ensure generalizability.

Table 6: The mean and the standard deviation of three runs on ligand efficacy prediction. The best results are marked in bold and the second best are underlined.

Repr.	Model	AUROC↑	AUPRC↑
Block	SchNet	0.732 ± 0.022	0.718 ± 0.031
	DimeNet++	0.669 ± 0.014	0.609 ± 0.036
	EGNN	$\underline{0.746 \pm 0.017}$	$\mathbf{0.755 \pm 0.031}$
	TorchMD	0.744 ± 0.034	0.721 ± 0.052
Atom	SchNet	0.712 ± 0.026	0.639 ± 0.033
	DimeNet++	0.589 ± 0.049	0.503 ± 0.020
	EGNN	0.711 ± 0.020	0.643 ± 0.041
	TorchMD	0.677 ± 0.004	0.636 ± 0.054
Hierarchical	SchNet	0.736 ± 0.020	0.731 ± 0.048
	DimeNet++	0.579 ± 0.118	0.517 ± 0.100
	EGNN	0.724 ± 0.027	0.720 ± 0.056
	TorchMD	0.717 ± 0.033	0.724 ± 0.055
Unified	GET (ours)	$\mathbf{0.761 \pm 0.012}$	$\underline{0.751 \pm 0.012}$

Results We present the mean and the standard deviation of the metrics across three runs in Table 6. LEP requires distinguishing the active and inactive conformations of the receptor, thus it is essential to capture the block-level geometry of the protein in addition to the atom-level receptor-ligand interactions. The unified representation excels at learning the bilevel geometry, therefore, naturally, we observe obvious gains on the metrics of our method compared to the baselines.

H Implementation Details

We conduct experiments on 1 GeForce RTX 2080 Ti GPU. Each model is trained with Adam optimizer and exponential learning rate decay. To avoid unstable checkpoints from early training stages, we select the latest checkpoint from the saved top-k checkpoints on the validation set for testing. Since the number of blocks varies a lot in different samples, we set a upperbound of the number of blocks to form a dynamic batch instead of using a static batch size. We use $k = 9$ for constructing the k-nearest neighbor graph in § 2.1 and set the size of the RBF kernel to 32. We give the description of the hyperparameters in Table 7 and their values for each task in Table 8.

Table 7: Descriptions of the hyperparameters.

hyperparameter	description
d_h	Hidden size
d_r	Radial size for the attention module
lr	Learning rate
final_lr	Final learning rate
max_epoch	Maximum of epochs to train
save_topk	Number of top-k checkpoints to save
n_layers	Number of layers
max_n_vertex	Upperbound of the number of nodes in a batch

H.1 PDBbind Benchmark

We follow Somnath et al. [51], Wang et al. [61] to conduct experiments on the well-established PDBbind [62, 37] and use the split with sequence identity threshold of 30% which should barely have data leakage problem. We directly borrow the results of the baselines from Wang et al. [61]. For our model, we set $d_h = 64, d_r = 64$, lr $= 10^{-3}$, final_lr $= 10^{-4}$, and max_n_vertex $= 1500$.

H.2 Protein-Protein Affinity

Here we illustrate the setup for protein-protein affinity prediction with more details.

Table 8: Hyperparameters for our GET on each task.

hyperparameter	PPA	LBA	LEP	hyperparameter	PPA	LBA	LEP
GET							
d_h	128	64	128	d_r	16	32	64
lr	10^{-4}	10^{-3}	5×10^{-4}	final_lr	10^{-4}	10^{-6}	10^{-4}
max_epoch	20	10	90	save_topk	3	3	7
n_layers	3	3	3	max_n_vertex	1500	2000	1500
GET-mix							
d_h	128	128	-	d_r	16	16	-
lr	5×10^{-5}	5×10^{-5}	-	final_lr	5×10^{-5}	10^{-6}	-
max_epoch	20	20	-	save_topk	3	3	-
n_layers	3	3	-	max_n_vertex	1500	1500	-

We adopt the Protein-Protein Affinity Benchmark Version 2 [31, 60] as the test set, which contains 176 diversified protein-protein complexes with annotated affinity collected from existing literature. These complexes are further categorized into three difficulty levels (i.e. Rigid, Medium, Flexible) according to the conformation change of the proteins from the unbound to the bound state [31], among which the Flexible split is the most challenging as the proteins undergo large conformation change upon binding.

As for training, we first filter out 2,500 complexes with annotated binding affinity (K_i or K_d) from PDBbind [62]. Then we use MMseqs2 [53] to cluster the sequences of these complexes together with the test set by dividing complexes with sequence identity above 30% into the same cluster, where sequence identity is calculated based on the BLOSUM62 substitution matrix [22]. The complexes that shares the same clusters with the test set are dropped to prevent data leakage, after which we finally obtained 2,195 valid complexes. We split these complexes into training set and validation set with a ratio of 9:1 with respect to the number of clusters. Following previous literature [6, 26, 44], we predict the negative log-transformed value (pK) instead of direct regression on the affinity.

Table 9: Number of paramters and training speed for baselines and our GET.

Repr.	Model	PPA		LBA		LEP	
		Parameter	Sec. / Batch	Parameter	Sec. / Batch	Parameter	Sec. / Batch
Block	SchNet	0.25M	0.054	0.15M	0.040	0.14M	0.118
	DimeNet++	1.53M	0.233	0.40M	0.189	0.40M	0.263
	EGNN	0.43M	0.054	0.12M	0.035	0.11M	0.130
	TorchMD	0.71M	0.072	0.20M	0.050	0.20M	0.133
Atom	SchNet	0.25M	0.109	0.15M	0.050	0.14M	0.123
	DimeNet++	-	-	0.40M	0.435	0.40M	0.357
	EGNN	0.43M	0.145	0.12M	0.060	0.11M	0.139
	TorchMD	0.71M	0.217	0.20M	0.079	0.20M	0.145
Hierarchical	SchNet	0.37M	0.127	0.21M	0.081	0.20M	0.088
	DimeNet++	-	-	0.60M	0.622	0.60M	0.633
	EGNN	0.61M	0.143	0.17M	0.077	0.17M	0.100
	TorchMD	1.00M	0.184	0.30M	0.104	0.29M	0.119
Unified	GET (w/o FFN)	0.23M	0.291	0.09M	0.193	0.20M	0.155
	GET	2.50M	0.339	0.69M	0.237	1.60M	0.192

H.3 Number of Parameters and Training Efficiency

We further provide the number of parameters and training efficiency for the baselines as well as our GET in Table 9. Although GET involves more parameters than the baselines, the extra parameters are mainly contributed to the FFN module (after removing FFN, the parameter size of GET is even smaller than most baselines due to its smaller intermediate hidden dimension). Nevertheless, adding FFN should not harm the efficiency much as the time cost mainly comes from message passing over edges, which is propotional to the number of edges, while time complexity of FFN is propotional to the number of nodes whose value is much smaller than the number of edges.

I Baselines

In this section, we describe the implementation details of different baselines. All the baselines are designed for structural learning on graphs whose nodes are represented as one feature vector and one coordinate. Therefore, for **block-level** representation, we average the embeddings and the coordinates of the atoms in each block before feeding the graph to the baselines. For **atom-level** representation, each node is represented as the embedding and the coordinate of each atom. For **Hierarchical** methods, we first implement message passing on atom-level graphs, then average the embeddings and coordinates within each block before conducting block-level message passing [28]. For fair comparison, the number of layers in each model is set to 3. We present other hyperparameters in Table 10.

Table 10: Hyperparameters for each baseline on each task.

hyperparameter	PPA	LBA	LEP	hyperparameter	PPA	LBA	LEP
SchNet							
d_h	128	64	64	max_n_vertex	1500	1500	1500
lr	10^{-3}	5×10^{-4}	10^{-3}	final_lr	10^{-4}	10^{-5}	10^{-4}
max_epoch	20	60	65	save_topk	3	5	5
SchNet-mix							
d_h	128	128	-	max_n_vertex	1500	1500	-
lr	5×10^{-5}	5×10^{-5}	-	final_lr	5×10^{-5}	5×10^{-5}	-
max_epoch	20	20	-	save_topk	3	3	-
DimeNet++							
d_h	128	64	64	max_n_vertex	1500	1500	1500
lr	10^{-3}	5×10^{-4}	10^{-3}	final_lr	10^{-4}	10^{-5}	10^{-4}
max_epoch	20	60	65	save_topk	3	5	5
EGNN							
d_h	128	64	64	max_n_vertex	1500	1500	1500
lr	10^{-3}	5×10^{-4}	10^{-3}	final_lr	10^{-4}	10^{-5}	10^{-4}
max_epoch	20	60	65	save_topk	3	5	5
TorchMD							
d_h	128	64	64	max_n_vertex	1500[3]	1500	1500
lr	10^{-3}	5×10^{-4}	10^{-3}	final_lr	10^{-4}	10^{-5}	10^{-4}
max_epoch	20	20	65	save_topk	3	3	5
TorchMD-mix							
d_h	128	128	-	max_n_vertex	1500	1500	-
lr	5×10^{-5}	5×10^{-5}	-	final_lr	5×10^{-5}	5×10^{-5}	-
max_epoch	20	20	-	save_topk	3	3	-

SchNet [49] We use the implementation in PyTorch Geometric [12] and project the edge feature into the same shape as the distance feature expanded with radial basis functions. Then we add the edge feature to the expanded distance feature as the edge attribute for fair comparison across different models.

DimeNet++ [16, 15] We use the implementation in PyTorch Geometric. Since the spherical basis functions used in DimeNet++ model the angles between any two neighbors of each node, the complexity is much higher than other baselines. Therefore, we fail to run atom-level DimeNet++ in the PPA task due to the large number of nodes in atom-level decomposition of proteins. Also, the atom-level DimeNet++ in the LBA task need 2 GeForce RTX 2080 Ti GPU for training.

EGNN [48] We directly use the official open-source codes provided in the original paper of EGNN.

TorchMD [54] We directly use the official implementation in the original paper. Similar to the adaption of SchNet, we also add the edge feature to the expanded distance for fair comparison.

[3]The value for the atom-level model is set to be 750 due to the higher complexity.

J Detailed Results of Protein-Protein Affinity

We show the detailed mean and standard deviation of three runs on all test splits of protein-protein affinity in Table 11.

Table 11: The mean and the standard deviation of three runs on protein-protein affinity prediction. The best results are marked in bold and the second best are underlined.

Repr.	Model	Rigid	Medium	Flexible	All
		Pearson↑			
Block	SchNet	0.542 ± 0.012	0.504 ± 0.020	0.102 ± 0.019	$\underline{0.439 \pm 0.016}$
	DimeNet++	0.487 ± 0.087	0.367 ± 0.043	$\underline{0.152 \pm 0.078}$	0.323 ± 0.025
	EGNN	0.437 ± 0.023	0.436 ± 0.028	0.094 ± 0.049	0.381 ± 0.021
	TorchMD-Net	0.575 ± 0.041	0.470 ± 0.024	0.087 ± 0.024	0.424 ± 0.021
Atom	SchNet	0.592 ± 0.007	$\mathbf{0.522 \pm 0.010}$	-0.038 ± 0.016	0.369 ± 0.007
	DimeNet++[4]	-	-	-	-
	EGNN	0.497 ± 0.027	0.452 ± 0.012	-0.054 ± 0.013	0.302 ± 0.010
	TorchMD-Net	$\underline{0.609 \pm 0.023}$	0.486 ± 0.004	0.049 ± 0.009	0.401 ± 0.005
Hierarchical	SchNet	0.542 ± 0.028	0.507 ± 0.020	0.098 ± 0.011	0.438 ± 0.017
	DimeNet++	-	-	-	-
	EGNN	0.461 ± 0.018	0.440 ± 0.024	0.089 ± 0.051	0.386 ± 0.021
	TorchMD-Net	0.572 ± 0.051	0.498 ± 0.025	0.101 ± 0.093	0.438 ± 0.026
Unified	GET (ours)	$\mathbf{0.670 \pm 0.017}$	$\underline{0.512 \pm 0.010}$	$\mathbf{0.381 \pm 0.014}$	$\mathbf{0.514 \pm 0.011}$
		Spearman↑			
Block	SchNet	0.476 ± 0.015	0.520 ± 0.013	0.068 ± 0.009	0.427 ± 0.012
	DimeNet++	0.466 ± 0.088	0.368 ± 0.037	$\underline{0.171 \pm 0.054}$	0.317 ± 0.031
	EGNN	0.364 ± 0.043	0.455 ± 0.026	0.080 ± 0.038	0.382 ± 0.022
	TorchMD-Net	0.552 ± 0.039	0.482 ± 0.025	0.090 ± 0.062	0.415 ± 0.027
Atom	SchNet	0.546 ± 0.005	0.512 ± 0.007	0.028 ± 0.032	0.404 ± 0.016
	DimeNet++	-	-	-	-
	EGNN	0.450 ± 0.042	0.438 ± 0.021	0.027 ± 0.030	0.349 ± 0.009
	TorchMD-Net	$\underline{0.582 \pm 0.025}$	0.487 ± 0.002	0.117 ± 0.008	$\underline{0.436 \pm 0.004}$
Hierarchical	SchNet	0.476 ± 0.017	$\underline{0.523 \pm 0.014}$	0.072 ± 0.021	0.424 ± 0.016
	DimeNet++	-	-	-	-
	EGNN	0.387 ± 0.023	0.461 ± 0.020	0.078 ± 0.043	0.390 ± 0.016
	TorchMD-Net	0.547 ± 0.045	0.516 ± 0.019	0.100 ± 0.111	0.438 ± 0.029
Unified	GET (ours)	$\mathbf{0.622 \pm 0.030}$	$\mathbf{0.533 \pm 0.014}$	$\mathbf{0.363 \pm 0.017}$	$\mathbf{0.533 \pm 0.011}$

K Limitations

First, current evaluations mainly focus on prediction tasks in molecular interactions. Generative tasks are another major branch in learning molecular interactions [38, 35, 43]. Designing generative algorithms for the proposed unified representation is non-trivial, and we leave this for future work. Further, it is also possible to generalize atom-level knowledge in other scenarios apart from molecular interactions. For instance, universal pretraining on different molecular domains, which needs careful design of the unsupervised task, hence we also leave this for future work.

L Sensitivity to Width and Depth

We show the performance with respect to dimensions of the hidden layers and the number of layers in Figure 5 on protein-protein affinity (PPA) and ligand-binding affinity (LBA). The performance is not so sensitive to both width and depth on LBA, while it is relatively more sensitive to width on PPA. Nevertheless, the common trend is that performance increases first and then decreases with

[4]We failed to run atom-level DimeNet++ due to the high complexity of the angular SBF.

dimension getting larger or the network getting deeper, so it is still necessary to find the suitable size of the model that the dataset can support.

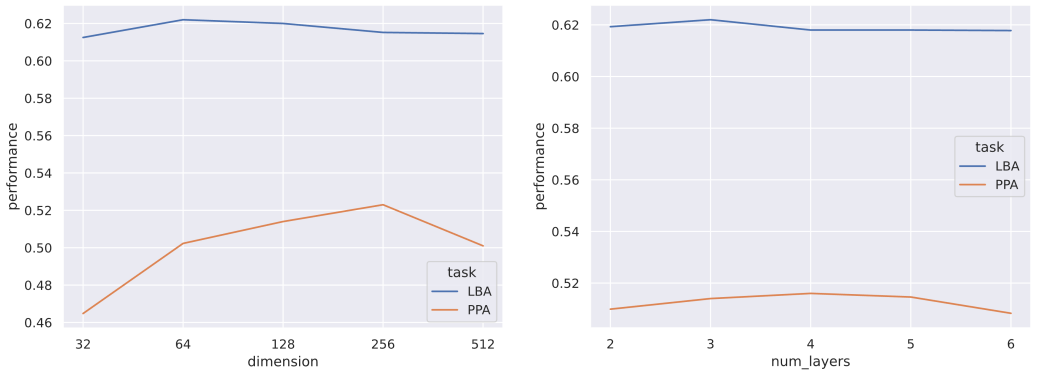

Figure 5: Performance with respect to the dimensions of the hidden layers (left) and the number of layers (right) on protein-protein affinity (PPA) and ligand-binding affinity (LBA).

M Attention Visualization

We visualize the attention weights between blocks on the interface of protein-protein complexes and compare them with the energy contributions calculated by Rosetta [1], which uses physics-based force fields. It is observed that the hot spots of attention weights largely agree with those predicted by Rosetta. We present three examples in Figure 6. The values are normalized by the maximum value (v_{max}) and the minimum value (v_{min}) in each figure (*i.e.* $v' = \frac{v - v_{\text{min}}}{v_{\text{max}} - v_{\text{min}}}$)

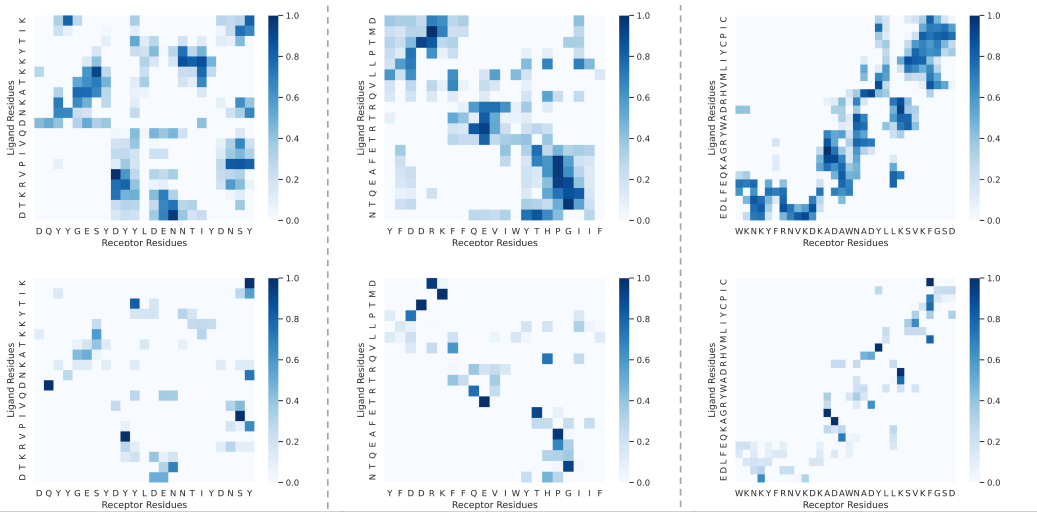

Figure 6: Attention weights of GET (upper row) and energy contributions given by Rosetta (lower row). PDB identities of the complexes are 1ahw, 1b6c, and 1gxd from left to right.

N Detailed Results of Universal Learning of Molecular Interaction Affinity

We provide the mean and the standard deviation of three parallel experiments on the universal learning of molecular interaction affinity (§ 3.3) in Tables 12 and 13.

Table 12: The mean and the standard deviation of three runs on protein-protein affinity prediction. Methods with the suffix "-mix" are trained on the mixed dataset of protein-protein affinity and ligand binding affinity. The best results are marked in bold and the second best are underlined.

Repr.	Model	Rigid	Medium	Flexible	All
		Pearson↑			
Block	SchNet	0.542 ± 0.012	0.504 ± 0.020	0.102 ± 0.019	0.439 ± 0.016
	SchNet-mix	0.553 ± 0.029	0.507 ± 0.011	0.093 ± 0.041	0.434 ± 0.011
	TorchMD-Net	0.575 ± 0.041	0.470 ± 0.024	0.087 ± 0.024	0.424 ± 0.021
	TorchMD-Net-mix	0.579 ± 0.028	0.502 ± 0.019	0.179 ± 0.044	0.457 ± 0.011
Atom	SchNet	0.592 ± 0.007	$\underline{0.522 \pm 0.010}$	-0.038 ± 0.016	0.369 ± 0.007
	SchNet-mix	0.625 ± 0.017	0.520 ± 0.021	-0.012 ± 0.049	0.421 ± 0.019
	TorchMD-Net	0.609 ± 0.023	0.486 ± 0.004	0.049 ± 0.009	0.401 ± 0.005
	TorchMD-Net-mix	0.618 ± 0.048	0.444 ± 0.027	0.057 ± 0.125	0.382 ± 0.029
Hierarchical	SchNet	0.542 ± 0.028	0.507 ± 0.020	0.098 ± 0.011	0.438 ± 0.017
	SchNet-mix	0.524 ± 0.031	0.515 ± 0.011	0.135 ± 0.077	0.429 ± 0.025
	TorchMD-Net	0.572 ± 0.051	0.498 ± 0.025	0.101 ± 0.093	0.438 ± 0.026
	TorchMD-Net-mix	0.494 ± 0.100	0.501 ± 0.007	0.130 ± 0.055	0.412 ± 0.035
Unified	GET (ours)	$\underline{0.670 \pm 0.017}$	0.512 ± 0.010	$\underline{0.381 \pm 0.014}$	$\underline{0.514 \pm 0.011}$
	GET-mix (ours)	$\mathbf{0.697 \pm 0.003}$	$\mathbf{0.533 \pm 0.004}$	$\mathbf{0.389 \pm 0.009}$	$\mathbf{0.519 \pm 0.004}$
		Spearman↑			
Block	SchNet	0.476 ± 0.015	0.520 ± 0.013	0.068 ± 0.009	0.427 ± 0.012
	SchNet-mix	0.497 ± 0.044	0.527 ± 0.009	0.042 ± 0.031	0.426 ± 0.007
	TorchMD-Net	0.552 ± 0.039	0.482 ± 0.025	0.090 ± 0.062	0.415 ± 0.027
	TorchMD-Net-mix	0.550 ± 0.039	0.524 ± 0.019	0.188 ± 0.070	0.472 ± 0.019
Atom	SchNet	0.546 ± 0.005	0.512 ± 0.007	0.028 ± 0.032	0.404 ± 0.016
	SchNet-mix	0.557 ± 0.042	0.516 ± 0.033	0.036 ± 0.010	0.428 ± 0.022
	TorchMD-Net	0.582 ± 0.025	0.487 ± 0.002	0.117 ± 0.008	0.436 ± 0.004
	TorchMD-Net-mix	0.608 ± 0.040	0.453 ± 0.037	0.058 ± 0.135	0.394 ± 0.027
Hierarchical	SchNet	0.476 ± 0.017	0.523 ± 0.014	0.072 ± 0.021	0.424 ± 0.016
	SchNet-mix	0.487 ± 0.027	0.532 ± 0.007	0.096 ± 0.053	0.412 ± 0.024
	TorchMD-Net	0.547 ± 0.045	0.516 ± 0.019	0.100 ± 0.111	0.438 ± 0.029
	TorchMD-Net-mix	0.446 ± 0.116	0.499 ± 0.009	0.143 ± 0.090	0.408 ± 0.042
Unified	GET (ours)	$\underline{0.622 \pm 0.030}$	$\underline{0.533 \pm 0.014}$	$\underline{0.363 \pm 0.017}$	$\underline{0.533 \pm 0.011}$
	GET-mix (ours)	$\mathbf{0.618 \pm 0.012}$	$\mathbf{0.555 \pm 0.008}$	$\mathbf{0.391 \pm 0.007}$	$\mathbf{0.537 \pm 0.003}$

Table 13: The mean and the standard deviation of three runs on ligand binding affinity prediction. Methods with the suffix "-mix" are trained on the mixed dataset of protein-protein affinity and ligand binding affinity. The best results are marked in bold and the second best are underlined.

Repr.	Model	LBA		
		RMSE↓	Pearson↑	Spearman↑
Block	SchNet	1.406 ± 0.020	0.565 ± 0.006	0.549 ± 0.007
	SchNet-mix	1.385 ± 0.016	0.573 ± 0.011	0.553 ± 0.012
	TorchMD-net	1.367 ± 0.037	0.599 ± 0.017	0.584 ± 0.025
	TorchMD-net-mix	1.423 ± 0.054	0.586 ± 0.012	0.567 ± 0.019
Atom	SchNet	1.357 ± 0.017	0.598 ± 0.011	0.592 ± 0.015
	SchNet-mix	1.365 ± 0.010	0.589 ± 0.006	0.575 ± 0.009
	TorchMD-net	1.381 ± 0.013	0.591 ± 0.007	0.583 ± 0.009
	TorchMD-net-mix	1.448 ± 0.122	0.566 ± 0.061	0.564 ± 0.059
Hierarchical	SchNet	1.370 ± 0.028	0.590 ± 0.017	0.571 ± 0.028
	SchNet-mix	1.403 ± 0.010	0.572 ± 0.004	0.554 ± 0.004
	TorchMD-net	1.383 ± 0.009	0.580 ± 0.008	0.564 ± 0.004
	TorchMD-net-mix	1.421 ± 0.032	0.569 ± 0.017	0.558 ± 0.017
Unified	GET (ours)	$\mathbf{1.327 \pm 0.005}$	$\underline{0.620 \pm 0.004}$	$\underline{0.611 \pm 0.003}$
	GET-mix (ours)	$\underline{1.329 \pm 0.008}$	$\mathbf{0.622 \pm 0.006}$	$\mathbf{0.615 \pm 0.008}$