Hierarchical Protein Representation for Interface Co-design with HICON

Anonymous Author(s)
Affiliation
Address
email

Abstract

Protein-protein interactions (PPIs) are essential for many biological processes, but their design is challenging due to their complex and dynamic nature. We propose a new model called Hierarchical Interface CO-design Network (HICON) that can jointly generate the sequence and 3D structure of protein interfaces. HICON uses a novel hierarchical architecture that combines atomic and amino acid resolutions in an equivariant manner and leverages Large Protein Language Models for sequence initialization. We evaluate HICON on a variety of biological interfaces, including protein-protein, enzyme-ligand, and antibody paratope-epitope interfaces. Our results show that HICON outperforms state-of-the-art models on sequence prediction and paratope co-design on several computational metrics.

1 Introduction and related work

Recent advances in generative models for biology[14, 21, 6] have revolutionized the field of protein interface design, enabling the development of novel binding proteins with unprecedented experimental success rates. This has opened up new possibilities for the design of protein interfaces with tailored properties, such as increased binding affinity, specificity, and stability.

There are three main challenges in designing novel interfaces: The first is generation scope. Existing models either generate only the sequence or only the structure (e.g., RFDiffusion[17]). It is desirable to perform *interface co-design*, or the joint generation of sequence and structure, as both are highly interdependent. The second is the applicability domain: Existing interface co-design models (e.g., RefineGNN[18]) focus on antibody CDRs. However, structural antibody datasets[1] are limited, and larger protein datasets contain a more diverse set of natural interfaces. It is desirable to optimize an architecture for generalized protein-protein interface design. The third challenge is model representation and efficiency: binding interactions occur at the atomic scale however modeling proteins at an all-atom level is computationally expensive and prone to learning noise. Hierarchical message-passing[5] is an effective strategy to introduce inductive bias and ensure learning efficacy. ProNet[10] and IEConv[11] leverage the hierarchical structure of proteins, but are non-generative. HSRN[19] provides a framework for generative hierarchical co-design networks, but its all-atom model does not scale well with large proteins.

In this paper, we propose a new architecture called Hierarchical Interface CO-design Network (HICON) to address the above challenges. HICON simultaneously generates the sequence and structure of a protein interface in a one-shot manner, leveraging Large Protein Language Models (LPLMs) for sequence initialization. Our architecture is optimized for generalized interface design, including enzyme pockets. Finally, HICON introduces a novel Equivariant Hierarchical message-passing network, leveraging atomic features in a scalable and efficient framework. We show that HICON outperforms state-of-the-art inverse folding and paratope co-design models on various computational metrics. We also demonstrate that HICON can be applied for PPI design as well as protein-small molecules in an enzyme test case.

2 Methods

2.1 Hierarchical 3D Graphs

Proteins can be naturally modeled as Hierarchical 3D graphs. A two-level Hierarchical 3D graph can be represented as $G = (V, E, P)$. Here, $V = \{v_i\}_{i=1}^{n}$ is the set of node features, where each $v_i \in \mathbb{R}^{k_i \times d_v}$ denotes the feature matrix for node i. $E = \{e_{ij}\}_{i,j=1}^{n}$ is the set of edge features, where $e_{ij} \in \mathbb{R}^{k^e_{ij} \times d_e}$ represents the edge feature matrix for edge (i, j). Furthermore, $P = \{P_i\}_{i=1}^{n}$ is the set of position matrices, where $P_i \in \mathbb{R}^{k_i \times 3}$ denotes the position matrix for node i. The parameters k_i, and k^e_{ij} can vary for different applications. For example, if we treat each atom in a molecule as a node, then $k_i = 1$ and $k^e_{ij} = 1$ for each node i, and each edge (i, j) respectively. Conversely, in the context of proteins, where each amino acid serves as a node, k_i signifies the number of atoms in amino acid i. And k^e_{ij} signifies the number of edges between atoms within amino acid i, and atoms within amino acid j, and between the atoms across both.

2.2 Hierarchical Message Passing

We generalize GNNs message-passing[8] on simple graphs for Hierarchical Graphs by considering all nodes and edges in the subgraphs as follows:

$$\begin{aligned} m^{(t)}_{v_{ip}} &= \sum_{j=1}^{n} \mathbb{1}_{k^e_{ij}>0} \sum_{q=1}^{k^e_{ij}} m^{(t)}_{pq} \\ m^{(t)}_{pq} &= M_t(h^{(t)}_{v_{ip}}, h^{(t)}_{v_{jq}}, e^{pq}_{ij}) \end{aligned}$$

Where $1 \leq p \leq k_i$ is a node in the subgraph i, and M_t is a Multi-Layer Perceptron. Our framework allows us to perform message-passing on the subgraph level and the simple graph level ($k_i = 1$ and $k^e_{ij} \leq 1$). In order to ensure end-to-end learning on both levels sequentially, we take a representative node (alpha carbon position and latent embedding) from each subgraph after the message-passing on the subgraph level.

2.3 Model architecture

The main assumption of this work is that the interface's sequence and 3D position primarily depend on atom-level interactions, which are often neglected when considering an amino acid-level abstraction. This statement holds especially true in interfaces involving small metabolites such as small molecules Fig. 1. Details are presented in Appendix S2.

HICON receives 3 main streams of information:

1. The complex graph is first passed through the **Complex Module** to encode the global embeddings of the complex as separate entities.
2. The atomic/chemical graph of the interface is then fed into the **Encoder** to embed the atomic-level geometrical and chemical structure of the interface.
3. The sequence information is embedded separately and fed into the **Decoder**.

Masking and Noising We train the model by masking amino acid types in the interface. We also remove sidechain atoms and noise the coordinates of backbone atoms using the

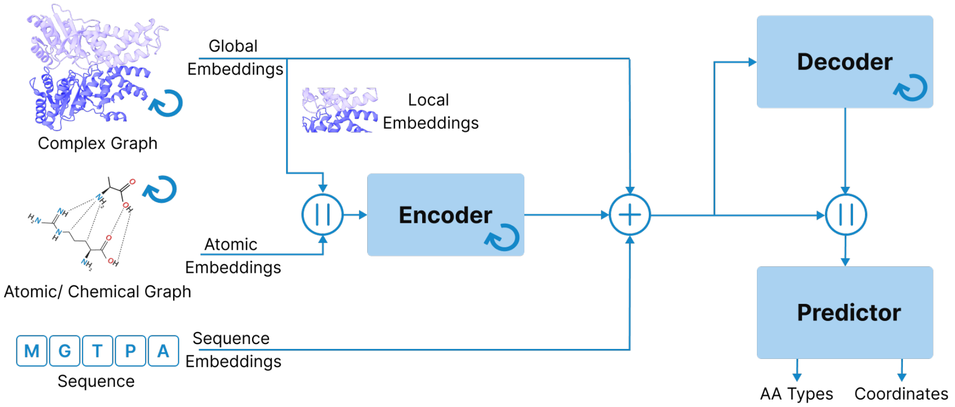

Figure 1: HICON's hierarchical framework: circular arrows indicate message-passing steps.

following noising scheme:

$$P_{trans} = P_{init} + C_{translation} \times AA\_translation^{AA}_{[0,1]} + C_{internal} \times internal\_noise^{Atomic}_{[0,1]}$$

$$P_{noised} = \left(P_{trans} - \overline{P}_{trans} \right) \cdot R^{AA}_{[-\frac{\pi}{2},\frac{\pi}{2}]} + \overline{P}_{trans}$$

ESM Initialization We leverage the sequential dependencies using Large Language Models trained on sequence data. Specifically, we use ESM-2[3] predictions to initialize the masked amino acids, as opposed to random or zero initialization.

3 Results

3.1 Interface Sequence Prediction

We evaluate our architecture on the inverse folding problem given partial sequence information using Proteinflow[9]. Masked amino acids are stripped of their side chain atoms. The length of the masked portion is $l = 20$. We provide non-masked amino acid types as node features for PiFold[22] and ProteinMPNN[2] and train them in a one-shot manner.

Table 1: Protein-protein interface sequence prediction benchmark: **First**, Second

Model	Accuracy	Perplexity
HICON-ESM35M	**50.9%**	**1.92**
ESM35M	18%	8.35
HICON	49.3%	2.38
PiFold	49.1%	2.37
ProteinMPNN	45.1%	2.95

3.2 PPI Co-design

We assess the codesign framework of HICON, with 2 cycles, using a general PPI dataset. As a baseline, we consider ProNet, with similar blocks as HICON, and equivariant layers.

Table 2: Protein-protein interface co-design benchmark: **First**, Second

Model	Accuracy	ESM Acc.	C_α RMSD	C RMSD	N RMSD
HICON-ESM150M	**29.1%**	23.2%	**1.48**	1.80	**1.76**
HICON-ESM35M	26.2%	18%	1.52	**1.79**	1.77
HICON	21.9%	-	1.61	1.99	1.96
ProNet	16.2%	-	2.79	3.51	3.42

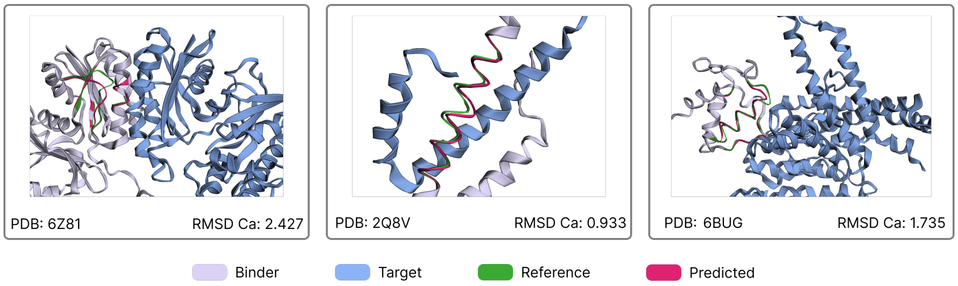

Figure 2: 3 examples of HICON-ESM35M predicted coordinates. Individual C_α RMSDs for subfigures 1, 2, and 3 are respectively 2.427, 0.933, and 1.735

3.3 Enzyme Co-design

The atomic features of the HICON model enable its use for another application: enzyme interface redesign. We construct 3 datasets from BRENDA[16] with 3 splits: Tanimoto, MMseqs[15] sequence similarity, and EC numbers.
We define the interface with a geometric radius around the ligand. We also add a separate chemical message passing block for the ligand. Details about the Protein-Ligand adaptation, and ablation studies, are presented in the supplementary materials.

Table 3: HICON-ESM35M results on enzyme co-design: **First**, Second

Split	Accuracy	ESM Acc.	Exp. Res.	C_α RMSD	C RMSD	N RMSD
Tanimoto	**45%**	29.7%	2.06 ± 0.47	**1.65**	1.97	1.99
MMseqs	26.8%	25.5%	2.15 ± 0.48	1.67	**1.91**	**1.94**
EC number	27.3%	26.7%	2.01 ± 0.46	1.73	1.95	**1.94**

Results show that splitting has a significant impact on sequence prediction accuracy. We suggest that ligands with highly dissimilar scaffolds might bind to similar pockets.

3.4 CDR-H3 Paratope Co-design

We test HSRN[19] with their position initialization module (original init), and with our noising (our init). We consider HSRN's dataset: the test set is a manually curated dataset used in RefineGNN[18], and in Diffab[13], and the training set is extracted from SabDAb[1]. We also train MEAN[20] using our initialization, and the original Diffab[13].

Table 4: CDR-H3 paratope co-design benchmark: **First**, Second

Model	Accuracy	Perplexity	C_α RMSD	C RMSD	N RMSD
HICON-ESM35M	**31.20%**	**2.97**	1.86	2.13	2.08
HSRN (original init)	28.02%	7.59	2.96	2.85	2.8
HSRN (our init)	27.71%	8.47	2.78	3.18	2.66
MEAN	28.70%	4.10	**1.53**	**1.43**	**1.45**
Diffab	26.93%	-	3.44	3.38	2.90
ESM35M	7.4%	8.36	-	-	-

4 Conclusion and Further Work

This paper presents HICON, a model for protein interface co-design. We introduce a hierarchical message-passing framework and LPLMs for sequence initialization. Our experiments show that HICON outperforms existing methods in PPI sequence prediction and CDR-H3 co-design, and can design general PPIs and enzymes using atomic information.
For further research, we can explore a position initialization module and test its effects on robustness and stability. Additionally, experimental validation can further assess the model's capacity to improve binding affinity in various use cases.

5 Appendix

5.1 Ablation Studies

To test our main assumption regarding the architecture and optimal level of representation for encoding protein interaction information (full atom or residue level), we evaluated the impact of different architecture choices on the Protein-Protein Interface Co-Design dataset. We consider 3 experiments that evaluate the validity of our hierarchical framework and the atomic-level representation. We also compare the sizes of different models in terms of the number of parameters, as an indicator of the models' representation efficiency, and inference speed.

- The first model HICON_{AA}, operates using the same complex+interface framework at a full amino acid resolution; i.e.: removing the chemical, and atomic message passing blocks.
- The second model $\text{HICON}_{\text{interface}}$, only operates on the interface (with atomic and amino-acid message passing), but without the complex block.
- The third model $\text{HICON}_{\text{AA\_complex}}$ performs amino-acid-level message passing on the whole complex. We achieve this by removing the complex block and setting the interface-defining radius to 100Å: the geometric area where the message passing is performed in the encoder and decoder.

Table 1: Ablation experiments results (HICON-ESM35M) **First** [1]

Model	Nb. params*	Accuracy	C_α RMSD	C RMSD	N RMSD
HICON	20.2M	**26.2%**	1.52	**1.79**	1.77
HICON_{AA}	11.9M	24.9%	1.58	2.9	2.83
$\text{HICON}_{\text{interface}}$	17.4M	**26.2%**	**1.46**	**1.79**	**1.75**
$\text{HICON}_{\text{AA\_complex}}$	9.1M	23.5%	2.08	2.75	3.08

Results show that the atomic hierarchical message passing ($\text{HICON}_{\text{interface}}$, HICON) improves results significantly on both sequence prediction accuracy, as well as RMSD, when compared to models with only amino-acid resolution (HICON_{AA}, $\text{HICON}_{\text{AA\_complex}}$). Additionally, $\text{HICON}_{\text{interface}}$ slightly outperforms the baseline on the structure prediction, while having 2.8M fewer parameters. This result suggests that complex-level information does not have a significant impact on interface co-design, which validates restricting most of the HICON's computations around the interface. Note that due to hardware limitations, we could not test a fully atomic model.

Enzyme ablation: In the absence of other models to benchmark our enzyme co-design model, we resorted to testing if the model used ligand information to reach its structural and sequential performance. We perform a test consisting of training the same model on an enzyme dataset, with or without the ligand information, enabling us to test the contribution of the ligand in the design of the pocket.
For this experiment, we curated a cleaner enzyme dataset, from NLDB[**?**], which provides the active compound associated with the enzyme for each pdb id. This avoids retrieving the ligand from raw pdbs where co-factors and other small molecules might be picked up as ligands, adding noise to the input. We cluster and split using MMseqs[15] similarity measure with a 30% threshold.
In the table below, we compare HICON-ESM35M, and HICON-ESM35M_nolig, on the NLDB dataset.

[1] Excluding frozen ESM-35M parameters

Table 2: HICON-ESM35M results on NLDB enzyme co-design **First** [2]

Split	Accuracy	Perplexity	Exp. Res.*	C_α RMSD	C RMSD	N RMSD
HICON	**29.6%**	**4.13**	2.13 ± 0.45	**1.65**	**2.03**	**2.01**
HICON_nolig	29.1%	4.43	2.13 ± 0.45	1.85	2.22	2.17
ESM35M	21.8%	8.36	-	-	-	-

We observe that removing the ligand information has a small effect on the sequence retrieval accuracy and a more significant impact on the structure RMSD. Due to the large diversity of enzymes present in our datasets, we expect that limiting the pocket diversity could enhance the impact of the ligand class in the codesign results. This could be done by limiting the dataset to a specific enzyme class or by curating a new dataset where similar enzyme sequences appear with several substrate types.

Size ablation: In our experiments, we tested HICON with 20 missing amino acids for fair comparison, and showed results for CDR-H3 co-design, which is a smaller interface. To test the performance with bigger interfaces, we tested our model on the general PPI dataset with a larger masked interface. The results are shown below:

Table 3: HICON results with larger interface sizes **First**

Split	Accuracy	C_α RMSD	C RMSD	N RMSD
20 AA	**21.9%**	**1.61**	**1.99**	**1.96**
40 AA	20.4%	1.92	2.27	2.24

5.2 Model architecture

5.2.1 Complex Module

The complex module encodes global information about the pair of molecules that might affect the interface information. It performs message passing on both graphs separately at an amino acid resolution. Namely, input node features are

$$v_i = \Big(OneHot(Amino\ Acid\ Type)\ ,\ Dihedrals(pos_{C\alpha}, pos_C, pos_N) \Big) \in \mathbb{R}^{d_v} \tag{1}$$

Where $d_v = 21 + 6$, as in [7], is the node input size, formed of a one-hot encoding of the amino-acid type, and the dihedral angles of the backbone. Input Node coordinates are $P_i = pos^i_{C_\alpha}$. While input edge attributes are:

$$e_{ij} = \Bigg(PosEmb_{16}(i,j),\ O_i^T \frac{P_j - P_i}{\|P_j - P_i\|},\ q(O_i^T O_j)\ ,\ \bigoplus_{(k,l)\in backbone_i \times backbone_j}^{n} RBF_{16}(\|P_k - P_l\|) \Bigg) \in \mathbb{R}^{d_e} \tag{2}$$

The different terms in 5.2.1 are denoted respectively as follows [7]:

- $PosEmb_{16}$ is a positional embedding that represents distances between residues in the sequence (rather than space). This edge feature indicates if connecting nodes are in close proximity in the sequence (adjacent amino acids).
- The second vector is a direction encoding that corresponds to the relative direction of P_j in the reference frame of (P_i, O_i). O_i being the relative orientation of node i to node $i+1$.
- The third vector is an orientation encoding $q(.)$ of the quaternion representation of the spatial rotation matrix $O_i^T O_j$. Quaternions represent 3D rotations as four-element vectors that can be efficiently and reasonably compared by an inner product. Both the second and third terms describe the relative geometrical orientation of the nodes.

[2] *Exp. Res.: mean PDB experimental resolution threshold in resp. test-sets

- The fourth term, as in [2], is formed of the distance encoding in radial basis 16 between all possible pairs of backbone atoms in amino acids i and j. The concatenation of all distance vectors gives a full representation of the relative positions of the pair of amino acids involved in the edge.

The resulting total input edge dimension is $d_e = 423$.

Linear Message Passing The first level of message passing in the complex graph occurs on the linear graph defined by the protein sequence. Thus, the set of edges in the linear graph is defined as follows:

$$E = \bigcup_{seq \in \{sequence_1, sequence_2\}} \{(k, k+1[N]), k \in 0...N-1, N = |seq|\} \tag{3}$$

Where $sequence_1$ and $sequence_2$ represent the amino acid sequences of protein 1 and protein 2 respectively. The message-passing operation uses the EGNN layer introduced in the methods section, with a depth of 4.

Amino-Acid Message Passing The output node, and edge embeddings, as well as node coordinates from the linear block, are passed to a second message-passing block that operates on the global 3D structure of the complex at an amino-acid resolution. Namely, we construct the graph edges as follows:

$$E = \{(i,j) \in [1...n]^2, \|P_i - P_j\| \leq radius_{gnn}, \\ |E_{i*}| \leq 32, |E_{j*}| \leq 32, Protein[i] = Protein[j]\} \tag{4}$$

We connect nodes from the same protein within a geometric radius ($radius_{complex\_gnn} = 20$), with a maximum number of neighbors set to 32.

5.2.2 Encoder

The encoder module operates on the interface only. Message-passing occurs on both the atomic and amino-acid resolutions. We introduce the first operator that selects the interface using the masked amino acids $Chain\_mask_l$:

$$S_{interface}(X) = \{i \in X, \exists j \in Chain\_mask_l, \|P_i^{(0)} - P_j^{(0)}\| \leq radius_{interface}\} \tag{5}$$

Where $radius_{interface} = 15$ in most experiments. $S_{interface}$ selects a geometric region around the masked amino acids in the interface using input positions. We, therefore, select the global node embeddings, after message-passing in the complex block, as initialization for the encoder's amino acid level vectors.

$$V_{encoder}^{global} = S_{interface}(MLP_v^{enc}(h_v^{complex})) \tag{6}$$

$$P_{encoder}^{C\alpha} = S_{interface}(P^{complex}) \tag{7}$$

$$P_{encoder}^{atoms} = P_{encoder}^{atoms} + (S_{interface}(P^{complex}) - S_{interface}(P_{init}^{complex})) \tag{8}$$

Where MLP_v^{enc} compresses the complex embeddings from $\mathbb{R}^{hidden\_dim}$ to $\mathbb{R}^{hidden\_dim/2}$. And the sidechain atoms' positions are translated by a vector in the direction of the new $C\alpha$ coordinates.

To compute the atomic-level embeddings, we perform message passing sequentially on two atomic blocks described below:

Chemical Message Passing This module considers the full graph on the subgraph level i.e.: all atoms. No side chain atoms and information are given in the masked amino acids. This is done by initializing masked amino acids as Glycine, which only contains 4 backbone atoms, to ignore any sidechain information. Node input features are the biological atom type ($C\alpha$, $C\beta$, $NE1$...). Moreover, the edges are defined as the chemical bonds between these atoms. Input edges features are defined using rdkit [12]: type of bond between two atoms, such as single, double, or triple. The stereo property indicates whether the bond is cis or trans, and the is_conjugated property indicates whether the bond is part of a conjugated system.

We project these features to a latent space in $\mathbb{R}^{hidden\_dim/2}$. Then, 4 layers of message passing update the atom positions and embeddings, which learn to fix bond lengths of the initial noisy structure.

Atomic Message Passing This module performs geometric message-passing on an atomic level. The input node embeddings, as well as atom positions, are the output of the chemical message-passing block. Then, It passes messages to nearby atoms in the graph, including neighboring atoms of other protein chains or ligands. We construct the graph edges as below:

$$E \quad = \quad \left\{ \begin{array}{r} (p,q) \in [1...k_i] \times [1...k_j], (i,j) \in [1...n]^2, \\ \|P_i^p - P_j^q\| \leq radius_{atom\_gnn}, |E_{ij}^{q*}| \leq 24, |E_{ij}^{p*}| \leq 24 \end{array} \right\} \tag{9}$$

$$e_{ij}^{pq} = \left(PosEmb_{16}(i,j),\ RBF_{16}(\|P_i^p - P_j^q\|) \right) \in \mathbb{R}^{d_e'}, d_e' = 32 \tag{10}$$

Where $radius_{atom\_gnn} = 10$. After 4 layers of message-passing, we introduce the following operator to select an amino acid representative, allowing us to transition to the amino acid level:

$$S_{C\alpha}(X) = \{x_i^{index_{C\alpha}} \in X, i \in [1...n]\} \tag{11}$$

Therefore, we define $V_{encoder}^{atom}$, and $P_{encoder}^{AA}$ as:

$$h_{encoder}^{atom} = S_{C\alpha}(h_v^{atom}) \in \mathbb{R}^{hidden\_dim/2} \tag{12}$$

$$P_{encoder}^{AA} = S_{C\alpha}(P^{atom}) \tag{13}$$

Afterward, we concatenate $h_{encoder}^{atom}$, and $V_{encoder}^{global}$ along the last dimension, defining $V_{encoder}^{AA}$:

$$V_{encoder}^{AA} = \left(h_{encoder}^{global}, h_{encoder}^{atom} \right) \in \mathbb{R}^{hidden\_dim} \tag{14}$$

Taking $V_{encoder}^{AA}$, and $P_{encoder}^{AA}$ as input, we perform 4 layers of message passing on an amino acid level, similar to the complex's 5.2.1, with a lower cutoff $radius_{AA\_gnn} = 10$, and allowing for messages to pass between nodes across both proteins or protein-ligand in the interface.

5.2.3 Decoder

This module aggregates information from 3 different inputs: the complex module's embeddings, encoders atomic, and amino acid embeddings and coordinates, as well as the input sequence embeddings, defined as:

$$V_{sequence} = MLP_{seq}(sequence) \in \mathbb{R}^{hidden\_dim} \tag{15}$$

Similar to the encoder, the decoder operates only on the interface. The input to this module are:

$$V_{decoder}^{AA(0)} = S_{interface}(V_{sequence}) + S_{interface}(h_v^{complex}) + h_v^{encoder} \in \mathbb{R}^{hidden\_dim}$$

$$P_{decoder}^{C\alpha} = P_{encoder}^{C\alpha}$$

$$V_{decoder}^{atom} = MLP_v^{decoder}(h_{encoder}^{atom}) \in \mathbb{R}^{hidden\_dim}$$

$$P_{decoder}^{atoms} = P_{encoder}^{atoms}$$

Where $MLP_v^{decoder}$ expands the the encoder's atomic embeddings from $\mathbb{R}^{hidden\_dim/2}$ to $\mathbb{R}^{hidden\_dim}$.

$V_{decoder}^{atom}$, and $P_{decoder}^{atoms}$, from the encoder, are used as input to perform 4 layers of atomic message passing similar to the encoder's 5.2.2, defining:

$$V_{decoder}^{AA} = \left(h_{decoder}^{AA},\ S_{C\alpha}(h_{decoder}^{atom}) \right) \in \mathbb{R}^{hidden\_dim \times 2} \tag{16}$$

$$P_{decoder}^{AA} = S_{C\alpha}(P_{decoder}^{atom}) \tag{17}$$

Finally, we perform 4 layers of amino-acid message-passing geometrically similar to the encoder's 5.2.1. Fig. 3 summarizes the information flow within the encoder and decoder modules.

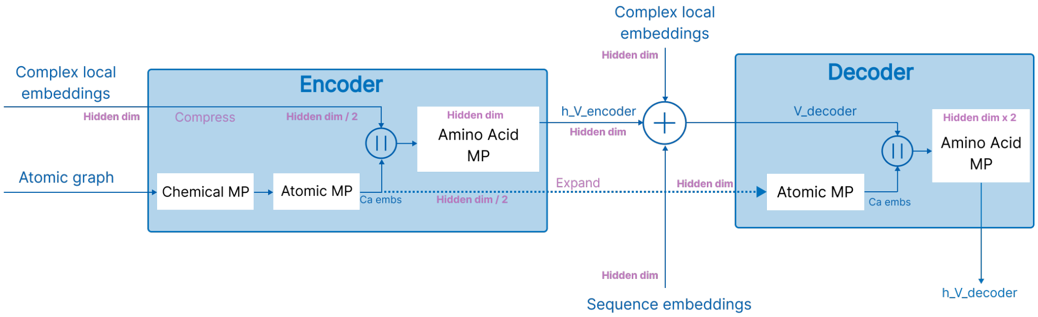

Figure 3: HICON's Encoder and decoder information flow. Expand and Compress are MLPs that map the node embeddings to a new embedding dimension.

5.2.4 Predictor

The predictor module maps the decoder's hidden node embeddings $h_v^{decoder}$ to the probability distribution of the amino acid type:

$$p(aa\_type \,|\, h_v^{decoder}) = Softmax(MLP_{out}(h_v^{decoder})),\ MLP_{out} : \mathbb{R}^{hidden\_dim \times 2} \longrightarrow \mathbb{R}^{20} \quad (18)$$

The predicted C_α coordinates correspond to the decoder's node coordinates after the amino-acid message-passing and the rest of the backbone coordinates correspond to the respective atom coordinates after the decoder's atomic message-passing.

5.3 Datasets

5.3.1 Curation and cleaning

We use the Protein Data Bank (PDB) [4] to access the protein and ligand atomic coordinates using PDB files and extract the aligned sequence from the entity's Fasta files. We remove entries with sequences longer than 2.000 residues or shorter than 30. We also set a resolution threshold of 3.5 Angstroms. Additionally, we remove chains with more than 10% missing residues in the middle and more than 30% missing residues at the ends of the protein chain. Finally, we remove redundant protein chains (sharing more than 90% sequence identity).

5.3.2 Protein Complexes Preprocessing Pipeline

To evaluate HICON on Protein-Protein interface co-design, we remove single-chain PDBs and generate pairs of interacting chains from the rest of the PDBs. Namely, we consider protein pairs with at least 3 contact points (C_α's with a distance ≤ 10Å) i.e:

$$is\_valid\_pair((chain_1, chain_2)) = |\{aa_1 \in chain_1, \exists aa_2 \in chain_2, \\ \|P_{C_\alpha}^{aa_1} - P_{C_\alpha}^{aa_2}\| \leq 10\}| \geq 3 \quad (19)$$

5.3.3 Enzyme Preprocessing Pipeline

We retrieve a list of enzyme PDBs and their corresponding EC classes from BRENDA [16]. Then, we extracted the ligands from every PDB. We do not consider ions, or heterogeneous molecules having covalent bonds to the protein. We also compare the atomic graph we get from the PDB with the natural SMILES of each molecule to get the correct canonical indexing of the atoms 3.

5.4 Splitting

We consider a (90%, 5%, 5%) split for training, validation, and testing. For the Protein-Protein dataset, we split the dataset according to the sequence similarity between single chains. Namely, we consider Algorithm 1 using MMseqs[15] clustering.

Algorithm 1: Sequence similarity splitting algorithm

```
Input :Chains = {(chain$_i$, $pdb\_id_i$), $i \in [1...N]$}, $valid\_ratio$, $test\_ratio$, $train\_ratio$
1 Clusters ← MMseqs_clustering(Chains) using 30% sequence similarity threshold.
2 Build a graph $G$(Clusters, E) where E are edges connecting clusters with chains sharing
    the same pdb_id.
3 Retrieve connected components C from graph G.
4 Assign sets of connected components $C_{valid} \subset C$, $C_{test} \subset C$, $C_{train} \subset C$, such that the total
    number of chains in each set of connected components is respectively equal to
    $valid\_ratio$, $test\_ratio$, $train\_ratio$ up to 20% margin.
5 if not possible, cut edges from the graph and repeat 4.
```

For the enzyme dataset, we consider 3 different splits. First, the sequence similarity splitting algorithm 1. Second, we consider splitting using ligand similarity. We achieve this by using a different clustering algorithm in step 1, using the Tanimoto similarity measure and Rdkit's Butina clustering algorithm with a similarity threshold of 30%. Third, we split according to EC classes, we select members from all major EC classes for training but prevent certain subclasses (ex: 2.1) to appear in both training, and validation and testing set, using algorithm 2 to partition the dataset.

Algorithm 2: EC classes splitting algorithm

```
Input :Enzymes = {(pdb_id$_i$, ec_number$_i$), $i \in$ [1...$N$], ec_number$_i \in$ X.Y.Z.T},
       valid_ratio, test_ratio, train_ratio
Cl$_{valid}$, $Cl_{test}$, $Cl_{train}$ = {}, {}, {}
for $x \leftarrow 1$ to $X$ do
    Clusters ← {(pdb_id$_i$, $ec\_number_i$), ec_number$_i = x.y.*.*$, $y \in$Y}
    Assign sets of clusters Cl$^x_{valid} \subset$ Clusters, Cl$^x_{test} \subset$ Clusters, Cl$^x_{train} \subset$ Clusters, using
      the Cutting stock algorithm with sizes valid_ratio × N, test_ratio × N, train_ratio
      × N
    Cl$_{valid} \leftarrow$ Cl$_{valid} \cup$ Cl$^x_{valid}$
    Cl$_{test} \leftarrow$ Cl$_{test} \cup$ Cl$^x_{test}$
    Cl$_{train} \leftarrow$ Cl$_{train} \cup$ Cl$^x_{train}$
end
```

When sampling from these datasets for training or inference, we iterate over the individual clusters and randomly pick a chain within that cluster. For the EC dataset, we sample from the smallest subclass i.e: x.y.z.* (example 2.1.1.*) instead of the clusters defined for the splitting. Meaning that we loop over the x.y.z.* subclasses, then we select a random data point from that subcluster (example: 2.1.1.2) as the next training/inference point.

Protein-Protein interface Preprocessing pipeline

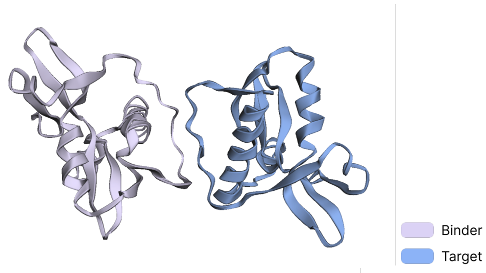

Figure 4: We consider a pair of proteins as a complex if they have at least 3 contact points (defined using $radius_{contact} = 10$).

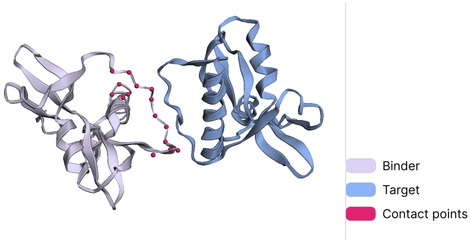

Figure 5: We calculate all contact points in the binder using $radius_{contact}$

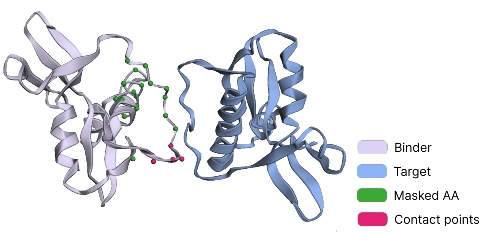

Figure 6: We pick a random contact point and consider the $l - 1$ closest neighbors in the binder as masked amino acids ($l = 20$).

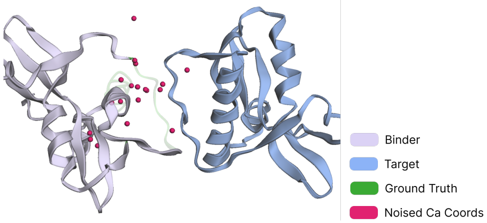

Figure 7: We noise the backbone coordinates and remove sidechain atoms of the masked amino acids. See Sec. **??** for more detail.

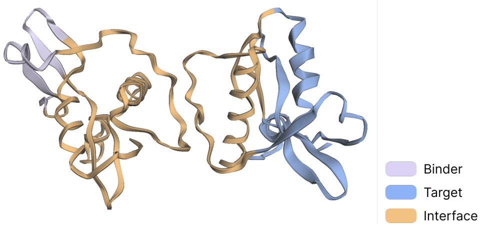

Figure 8: We define interface amino acids using $radius_{interface} = 15$ around each masked amino acids

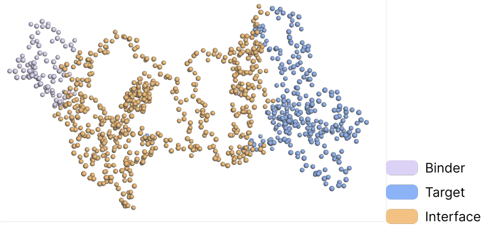

Figure 9: We construct 3 types of graphs using: $radius_{complex\_gnn}$=20 for the complex AA graph. $radius_{AA\_gnn}$=10 for the encoder and decoder AA graphs. $radius_{atom\_gnn}$=10 for atomic graphs.

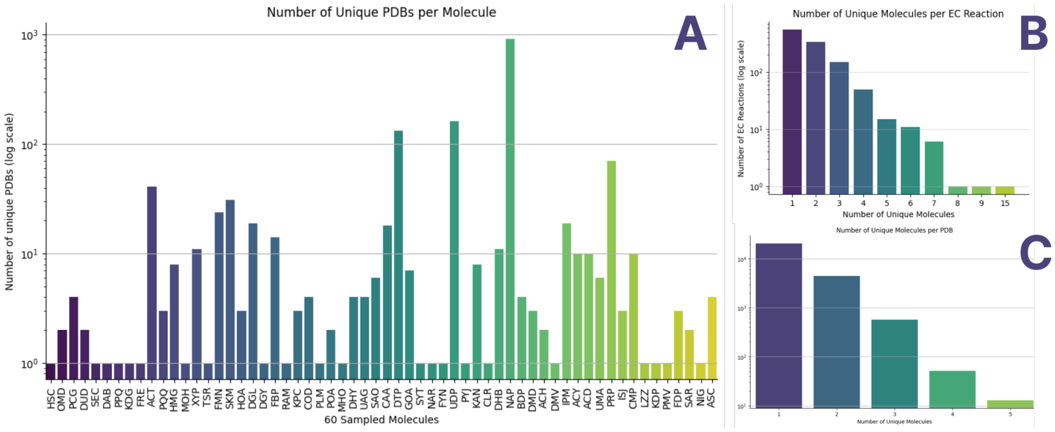

Figure 10: **A:** Number of unique pdb ids where specific molecules appear. 60 random molecules were shown. Some molecules are more common than others: ex. NAP, DTP, ACT... **B:** Number of unique active compounds involved in all instances of individual EC reactions. Data was generated based on NLDB: a subset of BRENDA. 83.3% of EC reactions have 4 or fewer active compounds involved in the reaction. **C:** Number of unique active small molecules per pdb in NLDB. 97.5% of PDBs have 2 or fewer active compounds.

Table 3: Enzyme dataset sources [3]

Dataset	Nb. PDB ids	Nb. unique compounds
BRENDA	80.173	-*
NLDB	9.284	466

Algorithm 3: Ligand extraction algorithm

```
Input   : BRENDA PDB ids
Output: Ligand smiles and atom coordinates
For each chain in PDBParser main chains: extract amino acid sequence as
  main_component, and consider the rest as molecules (filter for ions and amino acids)
Merge all main_components
For all other chains: Construct a connectivity matrix of length: len(ligands)+1 (index0
  is the main component)
Fix CONECT lines in the pdb
Get all covalent connections from the connect statements
Iteratively aggregate connected components together
Independent components are considered ligands
Extract smiles from pdb block
Compare smiles to the natural canonical smiles to reorder atoms
```

N.B.: Algorithm 3 reaches 70% accuracy when comparing the extracted ligands to the active compounds stated in NLDB.

[3] *We used BRENDA, which doesn't contain ligand information, for constructing our training dataset, and NLDB to extract statistics and validate our ligand extraction algorithm.

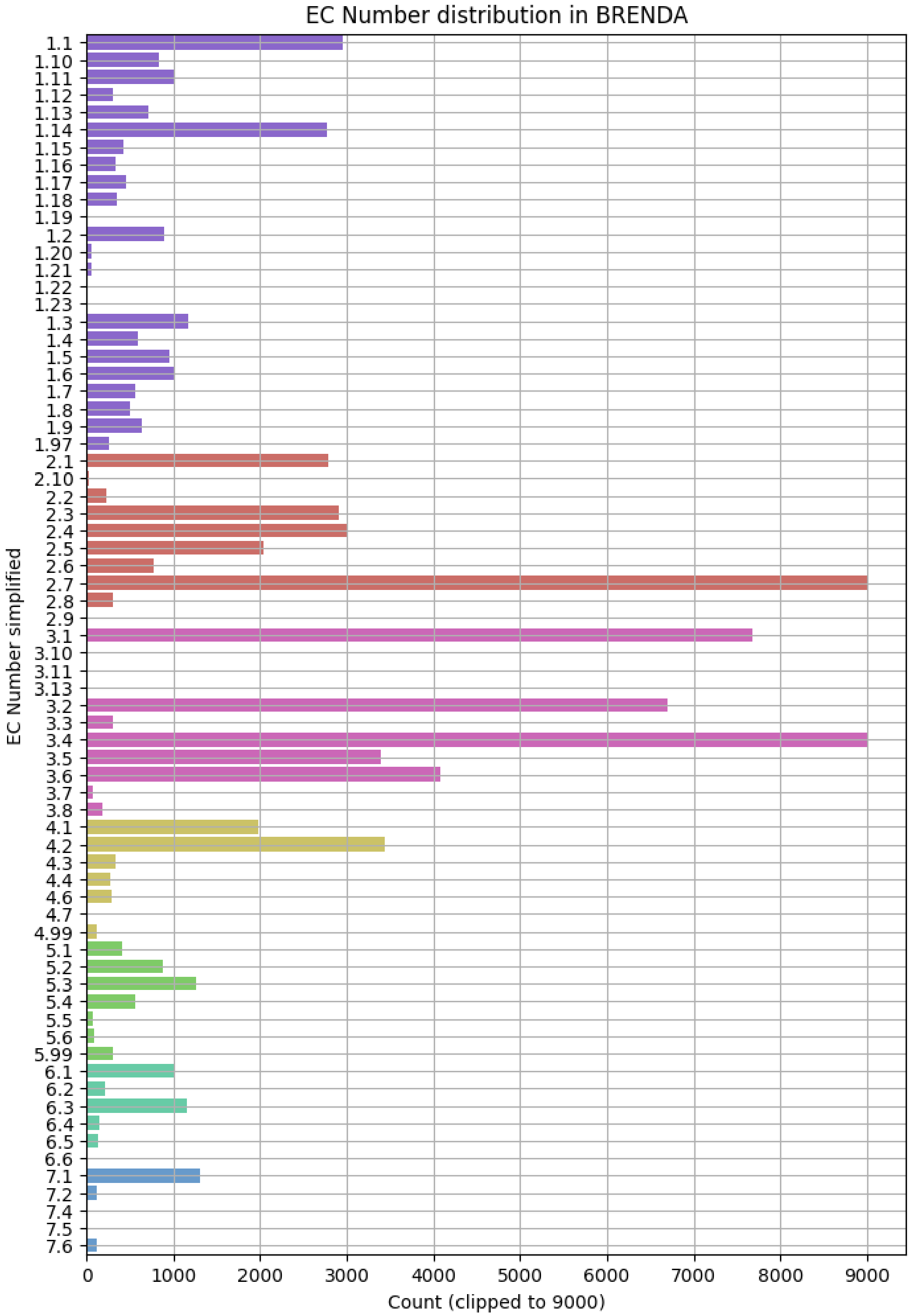

Figure 11: Distribution of simplified EC numbers (x.y.*.* example: 1.2) in BRENDA. Counts were clipped to 9000 for a simplified visualization.

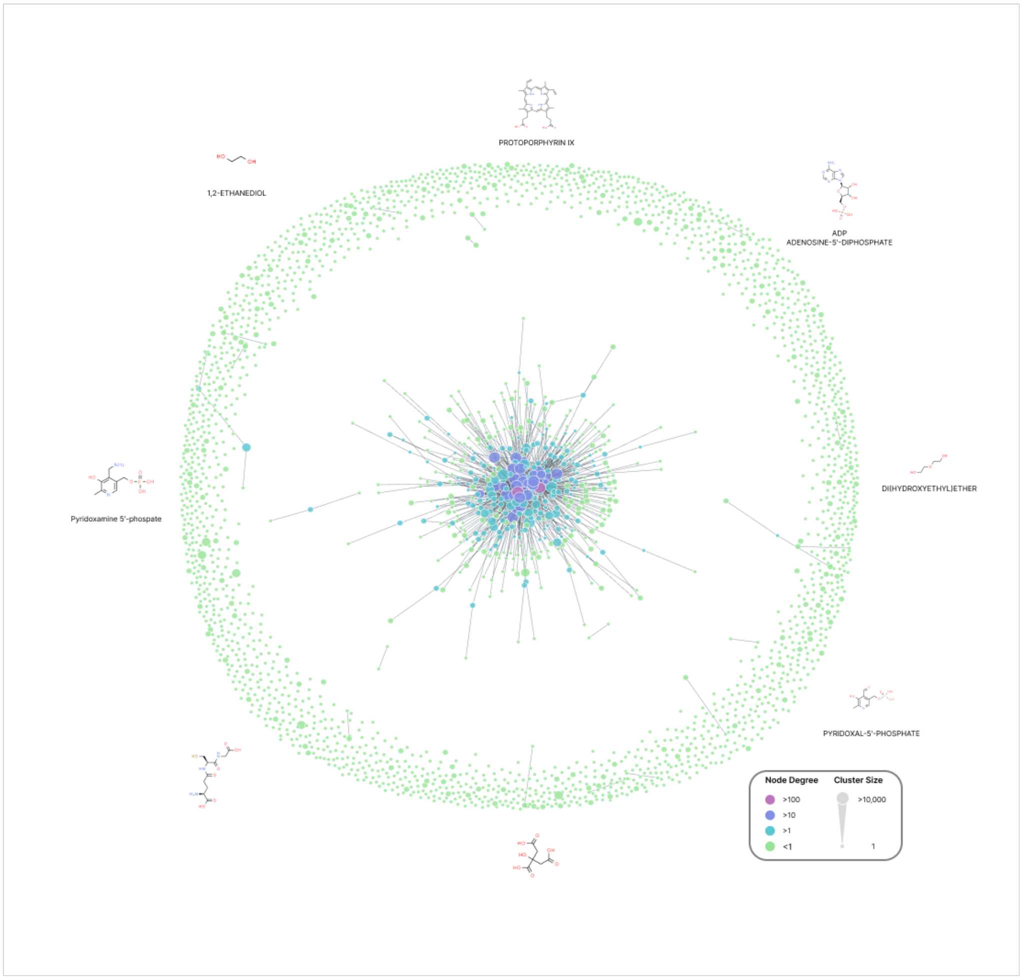

Figure 12: Visualization of the Tanimoto similarity-based clustering (30% threshold) of the BRENDA dataset. The biggest component is shown in the center, and edges denote pdb ids connections (chains in clusters appearing in the same pdb)

Table 4: Splits train/val/test partitions

Dataset	Tain	Validation	Test
Protein-Protein MMseqs	133.984	6.799	6.063
Enzyme MMseqs	47.531	2.355	1.415
Enzyme Tanimoto	46.735	2.060	2.506
Enzyme EC nb.	43.835	3.093	4.373
HSRN SAbDab	2.820	188	79

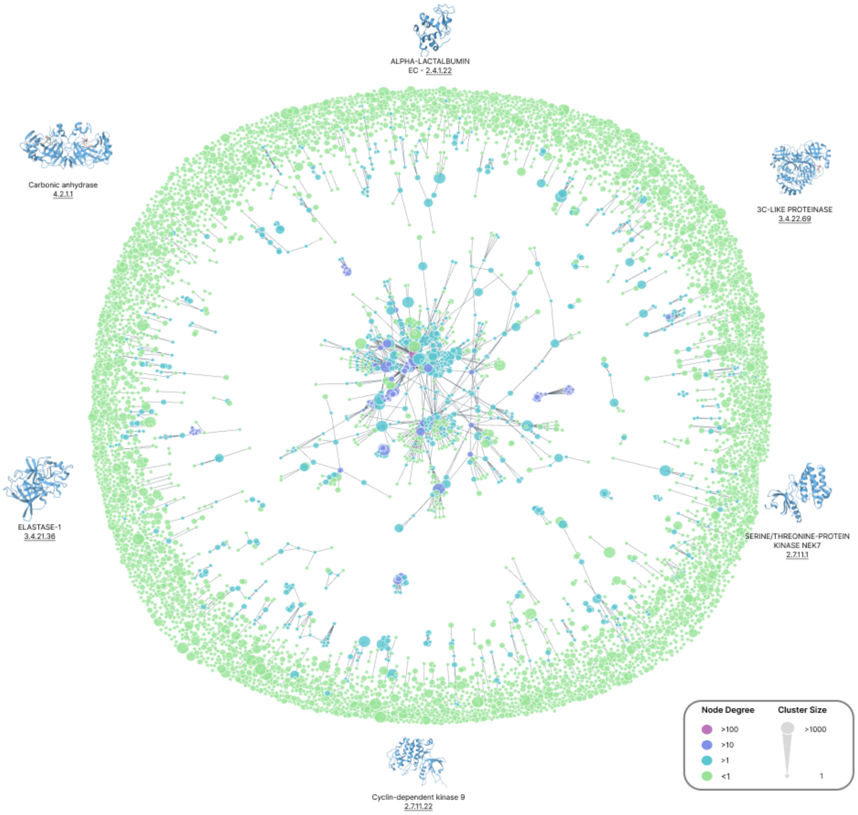

Figure 13: Visualization of the MMseqs similarity-based clustering (30% threshold) of the BRENDA dataset. The biggest component is shown in the center, and edges denote pdb ids connections (chains in clusters appearing in the same pdb)

References

[1] Krawczyk K. et al. Dunbar, J. Sabdab: the structural antibody database. https://doi.org/10.1093/nar/gkt1043, 2014.

[2] J. Dauparas et al. Robust deep learning–based protein sequence design using ProteinMPNN. https://doi.org/10.1126/science.add2187, 2022.

[3] Zeming Lin et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. https://doi.org/10.1126/science.ade2574, 2023.

[4] Z. Feng G. Gilliland T.N. Bhat H. Weissig I.N. Shindyalov P.E. Bourne H.M. Berman, J. Westbrook. The protein data bank. https://doi.org/10.1093/nar/28.1.235, 2000.

[5] Tingyang Xu Yu Rong Jiaqi Han, Wenbing Huang. Equivariant Graph Hierarchy-Based Neural Networks. https://doi.org/10.48550/arXiv.2202.10643, 2023.

[6] Max Baranov John Ingraham. Illuminating protein space with a programmable generative model. https://doi.org/10.1101/2022.12.01.518682, 2022.

[7] Regina Barzilay Tommi Jaakkola. John Ingraham, Vikas Garg. Generative Models for Graph-Based Protein Design. https://papers.nips.cc/paper_files/paper/2019/file/f3a4ff4839c56a5f460c88cce3666a2b-Paper.pdf, 2019.

[8] Patrick F. Riley Oriol Vinyals George E. Dahl Justin Gilmer, Samuel S. Schoenholz. Neural message passing for quantum chemistry. https://doi.org/10.48550/arXiv.1704.01212, 2017.

[9] Elizaveta Kozlova, Arthur Valentin, Aous Khadhraoui, and Daniel Nakhaee-Zadeh Gutierrez. Proteinflow: a python library to pre-process protein structure data for deep learning applications, 2023.

[10] Yi Liu Jerry Kurtin Shuiwang Ji Limei Wang, Haoran Liu. Learning Hierarchical Protein Representations via Complete 3D Graph Networks. https://doi.org/10.48550/arXiv.2207.12600, 2022.

[11] Matěj Lang Gloria Fackelmann Pere Pau Vázquez Barbora Kozlíková Michael Krone Tobias Ritschel Timo Ropinski Pedro Hermosilla, Marco Schäfer. Intrinsic-Extrinsic Convolution and Pooling for Learning on 3D Protein Structures. https://doi.org/10.48550/arXiv.2007.06252, 2021.

[12] RDKit:. Open-source cheminformatics. https://www.rdkit.org.

[13] Yufeng Su Shitong Luo. Antigen-Specific Antibody Design and Optimization with Diffusion-Based Generative Models for Protein Structures. https://doi.org/10.1101/2022.07.10.499510, 2022.

[14] Sam Tipps Lucas Arnoldt Samuel Hendel Jeremiah Nelson Sims Xinting Li David Baker Sidney Lyayuga Lisanza, Jake Merle Gershon. Joint generation of protein sequence and structure with rosettafold sequence space diffusion. https://doi.org/10.1101/2023.05.08.539766, 2023.

[15] Söding J. Steinegger, M. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. https://doi.org/10.1038/nbt.3988, 2017.

[16] Max Welling. Victor Garcia Satorras, Emiel Hoogeboom. E(n) Equivariant Graph Neural Networks. https://doi.org/10.48550/arXiv.2102.09844, 2021.

[17] Bennett N.R. et al. Watson J.L., Juergens D. De novo design of protein structure and function with RFdiffusion. https://doi.org/10.1038/s41586-023-06415-8, 2023.

[18] Regina Barzilay Tommi Jaakkola Wengong Jin, Jeremy Wohlwend. Iterative refinement graph neural network for antibody sequence-structure co-design. https://doi.org/10.48550/arXiv.2110.04624, 2021.

[19] Tommi Jaakkola Wengong Jin, Dr.Regina Barzilay. Antibody-Antigen Docking and Design via Hierarchical Structure Refinement. https://doi.org/10.48550/arXiv.2207.06616, 2022.

[20] Yang Liu Xiangzhe Kong, Wenbing Huang. Conditional antibody design as 3d equivariant graph translation. https://doi.org/10.48550/arXiv.2208.06073, 2022.

[21] Mohammed AlQuraishi Yeqing Lin. Generating Novel, Designable, and Diverse Protein Structures by Equivariantly Diffusing Oriented Residue Clouds. https://doi.org/10.48550/arXiv.2301.12485, 2023.

[22] Cheng Tan Zhangyang Gao. PiFold: Toward effective and efficient protein inverse folding. https://doi.org/10.48550/arXiv.2209.12643, 2022.