Antibody-Antigen Docking and Design via Hierarchical Structure Refinement

Wengong Jin [1] **Regina Barzilay** [2] **Tommi Jaakkola** [2]

Abstract

Computational antibody design seeks to automatically create an antibody that binds to an antigen. The binding affinity is governed by the 3D binding interface where antibody residues (paratope) closely interact with antigen residues (epitope). Thus, the key question of antibody design is how to predict the 3D paratope-epitope complex (i.e., docking) for paratope generation. In this paper, we propose a new model called Hierarchical Structure Refinement Network (HSRN) for paratope docking and design. During docking, HSRN employs a hierarchical message passing network to predict atomic forces and use them to refine a binding complex in an iterative, equivariant manner. During generation, its autoregressive decoder progressively docks generated paratopes and builds a geometric representation of the binding interface to guide the next residue choice. Our results show that HSRN significantly outperforms prior state-of-the-art on paratope docking and design benchmarks.

1. Introduction

Recently, generative models (Saka et al., 2021; Jin et al., 2021) have demonstrated the feasibility of *de novo* antibody design to combat a pathogen of interest (i.e., antigen). In this paper, we propose to further advance this generation capacity by tailoring antibody design models for binding a specific region of an antigen (an epitope). Figure 1 illustrates a binding interface consisting of an antigen epitope and an antibody paratope. While a typical antigen has multiple epitopes, some of them may constitute more desirable targets from a therapeutic perspective. For instance, binding to a more conserved epitope decreases the chance of viral escape and prolongs antibody efficacy (Yuan et al., 2020). Therefore, conditioning the generation process on a specific epitope gives us more control in antibody design.

[1]Eric and Wendy Schmidt Center, Broad Institute of MIT and Harvard [2]CSAIL, Massachusetts Institute of Technology. Correspondence to: Wengong Jin <wengong@csail.mit.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

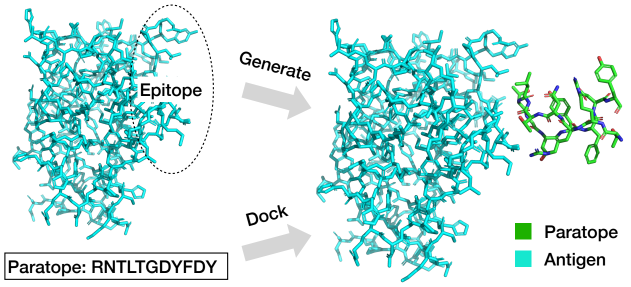

Figure 1. Illustration of paratope docking and design. The docking task assumes a paratope sequence is given and aims to predict the 3D coordinate of each atom. Paratope design assumes only the epitope is given and seeks to generate a paratope sequence and its 3D structure for epitope binding.

Antibody binding affinity relies on the quality of paratope-epitope match, similar to a lock and a key. Thus, predicting the paratope-epitope complex (docking) is the key to antibody engineering. There are three modeling challenges for paratope docking and design. The first is paratope flexibility. Existing rigid-body docking models (Yan et al., 2020; Ganea et al., 2021) assume the ligand 3D structure is given and fixed, so they cannot be integrated into a generative model. Since the paratope is being generated on the fly, we need to adjust the inferred paratope structure as more information arrives. The second is equivariance. Prior work on paratope design (Jin et al., 2021) cannot model a two-body system since it predicts the absolute coordinates while the epitope can be rotated arbitrarily. The third is efficiency. It is computationally expensive to build an all-atom structure from scratch because a binding interface usually involves hundreds of atoms. Previous methods (Ingraham et al., 2018) model protein backbone only and ignore the side-chain conformation, but side-chain atom contacts play an important role in binding.

In this paper, we propose a new architecture called Hierarchical Structure Refinement Network (HSRN) to address the above challenges.[1] As opposed to rigid-body docking, HSRN simultaneously folds and docks a paratope sequence and iteratively refines its 3D structure during generation.

[1]The code is available at github.com/wengong-jin/abdockgen

The main novelty of our approach is *hierarchical equivariance*. HSRN represents a binding interface as a dynamic, hierarchical graph. It consists of a residue-level graph with only C_α atoms and an atom-level graph that includes the side chains. This multi-scale representation allows us to factorize the paratope docking task into a global backbone update step and a local side-chain refinement step. To ensure equivariance of the docking procedure, HSRN predicts the pseudo force between atoms instead of their absolute coordinates. During generation, the HSRN decoder builds a series of docked paratope-epitope complexes and use their geometric representation to select the next amino acid.

We evaluate our method on standard paratope docking and design benchmarks (Adolf-Bryfogle et al., 2018). In terms of docking, we compare HSRN against HDOCK (Yan et al., 2020) and combined it with IgFold (Ruffolo & Gray, 2022) for end-to-end paratope folding and docking. In terms of generation, we compare HSRN against standard sequence-based generative models and a state-of-the-art structure-based methods (Jin et al., 2021). HSRN significantly outperformed all baselines in both settings, with over 50% absolute improvement in docking success rate.

2. Related Work

Protein docking. Our method is closely related to protein-protein docking (Kozakov et al., 2017; Yan et al., 2020; Ganea et al., 2021), which predicts the 3D structure of a protein-protein complex given the 3D structures of the two unbound proteins. These approaches assumes the paratope 3D structure is given and perform rigid-body docking. In contrast, HSRN predicts the 3D structure of a paratope-epitope complex by simultaneously folding and docking a paratope sequence to an epitope. In that regard, our work is more similar to AF2-multimer (Evans et al., 2021) that simultaneously folds two proteins. However, AF2-multimer relies on multi-sequence alignment (MSA) and sometimes template features to predict protein complex structures. Antibody paratopes are highly variable and generally lacks MSA data. Moreover, existing docking models are not directly applicable to a generative setting where they need to dock a partial sequence for paratope design. Our approach does not require MSA data and can be easily integrated into a generative design workflow.

Generative antibody/protein design. Recently, the antibody design community started to explore deep generative models due to their computational efficiency compared to traditional physics-based models (Lapidoth et al., 2015; Adolf-Bryfogle et al., 2018). For example, Liu et al. (2020); Shin et al. (2021); Saka et al. (2021); Akbar et al. (2021) trained recurrent neural networks to generate paratope sequences. The limitation of sequence-based models is that they do not utilize the paratope 3D structure. Therefore, Jin et al. (2021) proposed a graph-based generative model that co-designs a paratope sequence and its 3D structure. However, none of these models incorporated the antigen structure. Thus, existing methods will generate the paratope distribution for any epitope given a fixed training set.

For general protein design, most previous methods are not conditioned on a target protein structure. While Ingraham et al. (2019); Strokach et al. (2020); Karimi et al. (2020); Cao et al. (2021b) proposed generative models conditioned on a backbone structure or protein fold, they do not model the interaction between a generated protein and its target protein. This limitation also applies to energy-based protein design models based on TrRosetta (Tischer et al., 2020; Norn et al., 2021). While non-deep learning based methods (Adolf-Bryfogle et al., 2018; Cao et al., 2021a) use docking algorithms to design target-specific proteins, they are computationally very expensive. In contrast, we propose a new generative model that leverages both 3D structural information and epitope structure.

Protein structure encoder. Prior work has utilized graph neural networks (GNN) (Fout et al., 2017), 3D convolutional neural networks (Townshend et al., 2019), and equivariant neural networks (ENN) (Satorras et al., 2021; Eismann et al., 2021) to encode protein 3D structures. Similar to our work, Eismann et al. (2021) developed a hierarchical ENN that represents a protein in terms of backbone (C_α) and side chains. However, their method assumes the 3D structure of a full protein complex is given. Therefore, it can only be used to re-rank docked structures generated by another docking algorithm. In contrast, our model predicts the 3D structure of the backbone and side chains from scratch. Somnath et al. (2021) also developed a hierarchical GNN that represents a protein in terms of triangulated surfaces and residue C_α atoms. Their method is not applicable to our setting because it does not model the side chain atoms.

3. Paratope Docking and Design with HSRN

When an antibody binds to an antigen, they form a joint structure called an antibody-antigen complex. Its binding interface consists of a *paratope* and an *epitope*. A paratope is a sequence of residues $\boldsymbol{a} = \boldsymbol{a}_1 \cdots \boldsymbol{a}_n$ in the complementarity determining regions (CDRs) of an antibody. An epitope $\boldsymbol{b} = \boldsymbol{b}_1 \boldsymbol{b}_2 \cdots \boldsymbol{b}_m$ is composed of m residues that are closest to a paratope. It is a subsequence of an antigen $\boldsymbol{c} = \boldsymbol{c}_1 \boldsymbol{c}_2 \cdots \boldsymbol{c}_M$, where $\boldsymbol{b}_i = \boldsymbol{c}_{e_i}$ and e_i is the index of epitope residue $\boldsymbol{b}_i$ in the antigen.

The epitope 3D structure $\mathcal{G}_b$ is described as a point cloud of atoms $\{\boldsymbol{b}_{i,j}\}_{1 \le i \le m, 1 \le j \le n_i}$, where n_i is the number of atoms in residue $\boldsymbol{b}_i$. Likewise, the 3D structure of a paratope and a paratope-epitope binding interface is denoted as $\mathcal{G}_a$ and $\mathcal{G}_{a,b}$, respectively. The first four atoms in any residue correspond to its backbone atoms (N, C_α, C, O) and the rest are its side chain atoms. The 3D coordinate of an atom $\boldsymbol{b}_{i,k}$ is denoted as $\boldsymbol{x}(\boldsymbol{b}_{i,k}) \in \mathbb{R}^3$.

In this paper, we assume the 3D structure of the antigen and its epitope are given as inputs to our model. Hence, both paratope docking and generation can be formulated as a 3D point cloud completion task. Given a training set of antibody-antigen complexes, HSRN learns to append an epitope structure $\mathcal{G}_b$ with paratope atoms to form a binding interface $\mathcal{G}_{a,b}$ (Figure 1). Overall, HSRN consists of three modules to model 3D paratope-epitope interaction for docking and generation.

- Its *encoder module* learns to encode a docked paratope-epitope complex into a set of atom and residue vectors.
- Its *docking module* predicts the 3D structure of a paratope-epitope complex given any paratope sequence. In contrast to rigid body docking, our model simultaneously folds and docks a paratope sequence since the paratope 3D structure is unknown.
- Its *decoder module* generates paratope amino acids autoregressively and runs the docking module to update the paratope-epitope complex in each generation step.[2]

3.1. Hierarchical Encoder

During docking and generation, the input to HSRN is a point cloud with atoms from an antigen, its epitope, and its binding paratope. A point cloud typically contains over 10^3 atoms so we need to encode it at different resolutions for computational efficiency. Specifically, we propose a hierarchical message passing network (MPN) composed of an antigen encoder, an atom-level and a residue-level interface encoder. The antigen encoder models the interaction between epitope residues and other interior antigen residues. The atom-level interface encoder captures paratope-epitope interaction at the finest resolution, including atomic contacts in the side-chains. The residue-level interface encoder only considers backbone C_α atoms to focus on long-range residue interactions.

3.1.1. Antigen Encoder

We represent an antigen as a point cloud $\mathcal{G}_c$ with only its C_α atoms. Each residue $\boldsymbol{c}_i$ is represented by a feature vector $\boldsymbol{f}(\boldsymbol{c}_i)$ including its dihedral angles, polarity, hydropathy, volume, charge, and whether it is a hydrogen bond donor or acceptor. In the forward pass, we first construct a K nearest neighbor graph of $\mathcal{G}_c$, build a local coordinate frame for each residue, and compute edge features $\boldsymbol{f}(\boldsymbol{c}_i, \boldsymbol{c}_j)$ describing the distance, direction, and orientation between two residues (Ingraham et al., 2019). Next, we encode the antigen graph $\mathcal{G}_c$ using a message passing network (MPN) whose inputs include node features $\{\boldsymbol{f}(\boldsymbol{c}_i)\}$ and edge features $\{\boldsymbol{f}(\boldsymbol{c}_i, \boldsymbol{c}_j)\}$ (details in the appendix). The output of this MPN is a vector representation $\boldsymbol{h}(\boldsymbol{c}_i)$ for each antigen residue, which will be used for initial distance prediction in the docking module. Crucially, the whole encoding process is rotation and translation invariant.

[2]The decoder module is not used during paratope docking.

3.1.2. Atom-level Interface Encoder

The atom-level encoder keeps all the atoms in the point cloud $\mathcal{G}_{a,b}$ and constructs a K nearest neighbor graph $\mathcal{G}^A_{a,b}$. Each node feature is a one-hot encoding of its atom name (e.g., N, C_α, C_β, O). Each edge feature between two atoms $(\boldsymbol{a}_{i,k}, \boldsymbol{b}_{j,l})$ is represented as

$$\boldsymbol{f}(\boldsymbol{a}_{i,k}, \boldsymbol{b}_{j,l}) = \mathrm{RBF}(\|\boldsymbol{x}(\boldsymbol{a}_{i,k}) - \boldsymbol{x}(\boldsymbol{b}_{j,l})\|) \qquad (1)$$

where $\mathrm{RBF}(\cdot)$ encodes the distance between two atoms in a radial basis. We encode $\mathcal{G}^A_{a,b}$ by another MPN which learns a vector representation $\boldsymbol{h}(\boldsymbol{a}_{i,k}), \boldsymbol{h}(\boldsymbol{b}_{j,l})$ for each paratope atom $\boldsymbol{a}_{i,k}$ and epitope atom $\boldsymbol{b}_{j,l}$.

3.1.3. Residue-level Interface Encoder

The residue-level encoder only keeps the C_α atoms and constructs a K nearest neighbor graph $\mathcal{G}^R_{a,b}$ at the residue level. It is encoded by another MPN that operates on top of the antigen MPN and atom MPN. The initial representation for epitope and paratope residues are defined as

$$\tilde{\boldsymbol{f}}(\boldsymbol{a}_i) = \boldsymbol{f}(\boldsymbol{a}_i) \oplus \sum\nolimits_k \boldsymbol{h}(\boldsymbol{a}_{i,k}) \qquad (2)$$

$$\tilde{\boldsymbol{f}}(\boldsymbol{b}_j) = \boldsymbol{f}(\boldsymbol{b}_j) \oplus \sum\nolimits_l \boldsymbol{h}(\boldsymbol{b}_{j,l}) \qquad (3)$$

where $\oplus$ means vector concatenation and e_j is the index of the epitope residue $\boldsymbol{b}_j$ in the antigen $\boldsymbol{c}$. The initial representation is the concatenation of its amino acid features $\boldsymbol{f}(\boldsymbol{a}_i)$ and the sum of atom-level encodings $\boldsymbol{h}(\boldsymbol{a}_{i,k})$ within that residue. The edge features are the same as the antigen MPN. The encoder then takes the node and edge features $\tilde{\boldsymbol{f}}$ into another MPN to compute residue-level representations $\{\boldsymbol{h}(\boldsymbol{a}_i)\}, \{\boldsymbol{h}(\boldsymbol{b}_j)\}$ for the paratope and epitope.

In summary, the output of our hierarchical encoder is a set of residue representations $\{\boldsymbol{h}(\boldsymbol{a}_i), \boldsymbol{h}(\boldsymbol{b}_j), \boldsymbol{h}(\boldsymbol{c}_j)\}$ and atom representations $\{\boldsymbol{h}(\boldsymbol{a}_{i,k}), \boldsymbol{h}(\boldsymbol{b}_{j,l})\}$.

3.2. Docking Module

Given a paratope sequence $\boldsymbol{a}$ and epitope 3D structure $\mathcal{G}_b$, our docking module predicts the 3D coordinate of each atom to form a correct paratope-epitope complex $\mathcal{G}_{a,b}$. The docking task is challenging as the model needs to simultaneously fold and dock the paratope. It is hard to predict the 3D structure in one shot due to the complex dependency between different variables $\boldsymbol{x}(\boldsymbol{a}_{i,k}), \boldsymbol{x}(\boldsymbol{b}_{j,l})$ and various physical constraints. Thus, we propose an iterative refinement procedure to fold and dock a paratope onto a epitope.

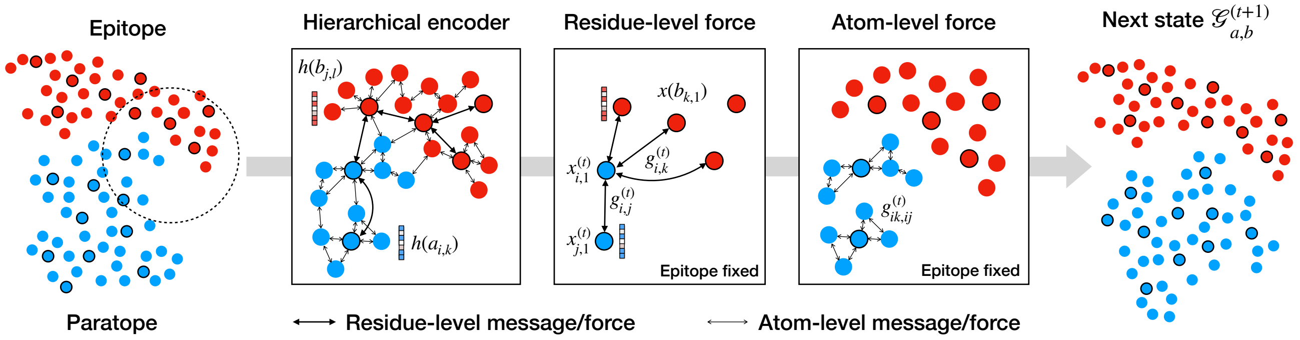

Figure 2. Illustration of the docking process (one iteration). In each refinement step, the docking module first encodes the residues and atoms into vector representations. It then computes the residue-level force between the C_α atoms and the local force between the side-chain atoms of the same residue. Finally, HSRN update the paratope-epitope structure based on the predicted forces.

3.2.1. Initialization

At the outset of structure refinement, we need to properly initialize the coordinates of all paratope atoms. Here we explore two initialization strategies.

Random initialization. A simple strategy is to initialize all coordinates by adding a small Gaussian noise around the center of the epitope

$$\boldsymbol{x}^{(0)}(\boldsymbol{a}_{i,k}) = \frac{1}{m}\sum_i \boldsymbol{x}(\boldsymbol{b}_{i,1}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0,1), \tag{4}$$

where the epitope center is defined as the mean of epitope C_α atoms. From now on, we will abbreviate paratope coordinates $\boldsymbol{x}^{(0)}(\boldsymbol{a}_{i,k})$ as $\boldsymbol{x}_{i,k}^{(0)}$ to simplify notation.

Distance-based initialization. Random initialization can be far from the final solution and difficult for HSRN to optimize. Another strategy is to directly predict the pairwise distance $\boldsymbol{D} \in \mathbb{R}^{(n+m)\times(n+m)}$ between paratope and epitope atoms. This scheme can be beneficial since coordinates inferred from $\boldsymbol{D}$ are closer to the final solution. Specifically, let $\boldsymbol{h}_{\text{seq}}(\boldsymbol{a}_i)$ be the paratope representation learned by a recurrent neural network. HSRN predicts the pairwise distance as the following:

$$\boldsymbol{D}_{i,j} = \begin{cases} \|\boldsymbol{h}_{\text{seq}}(\boldsymbol{a}_i) - \boldsymbol{h}_{\text{seq}}(\boldsymbol{a}_j)\| & i,j \le n \\ \|\boldsymbol{h}_{\text{seq}}(\boldsymbol{a}_i) - \boldsymbol{h}(\boldsymbol{c}_{e_j})\| & i \le n, j > n \\ \|\boldsymbol{x}(\boldsymbol{b}_i) - \boldsymbol{x}(\boldsymbol{b}_j)\| & i,j > n \end{cases}$$

Intuitively, if i belongs to the paratope ($i \le n$) and j to the epitope ($j > n$), $\boldsymbol{D}_{i,j}$ is the Euclidean distance between $\boldsymbol{h}_{\text{seq}}(\boldsymbol{a}_i)$ and the antigen encoding $\boldsymbol{h}(\boldsymbol{c}_{e_j})$. Similarly, the distance between two paratope residues is modeled as the Euclidean distance between their sequence representations $\boldsymbol{h}_{\text{seq}}$. The distance between two epitope residues are directly calculated from their given coordinates $\boldsymbol{x}(\boldsymbol{b}_i)$.

Given this pairwise distance matrix $\boldsymbol{D}$, we can obtain the 3D coordinates of each residue via eigenvalue decomposition of the following Gram matrix (Crippen & Havel, 1978)

$$\tilde{\boldsymbol{D}}_{i,j} = 0.5(\boldsymbol{D}_{i,1}^2 + \boldsymbol{D}_{1,j}^2 - \boldsymbol{D}_{i,j}^2), \quad \tilde{\boldsymbol{D}} = \boldsymbol{USU}^\top \tag{5}$$

where $\boldsymbol{S}$ is a diagonal matrix with eigenvalues in descending order. The coordinate of each residue $\boldsymbol{a}_i$ is calculated as

$$\tilde{\boldsymbol{x}}_i^{(0)} = [\boldsymbol{X}_{i,1}, \boldsymbol{X}_{i,2}, \boldsymbol{X}_{i,3}], \quad \boldsymbol{X} = \boldsymbol{U}\sqrt{\boldsymbol{S}}. \tag{6}$$

Note that the predicted coordinates $\{\tilde{\boldsymbol{x}}_i^{(0)}\}$ retain the original distance $\boldsymbol{D}$, but they are located in a different coordinate frame from the given epitope. Therefore, we apply the Kabsch algorithm (Kabsch, 1976) to find a rigid body transformation $(\boldsymbol{R}, \boldsymbol{t})$ that aligns the predicted epitope coordinates $\{\tilde{\boldsymbol{x}}_{n+1}^{(0)}, \cdots, \tilde{\boldsymbol{x}}_{n+m}^{(0)}\}$ with the given epitope $\{\boldsymbol{x}(\boldsymbol{b}_{1,1}), \cdots, \boldsymbol{x}(\boldsymbol{b}_{n,1})\}$. Lastly, the C_α coordinates of each paratope residue is predicted as

$$\boldsymbol{x}^{(0)}(\boldsymbol{a}_{i,1}) = \boldsymbol{x}_{i,1}^{(0)} = \boldsymbol{R}\tilde{\boldsymbol{x}}_i^{(0)} + \boldsymbol{t} \tag{7}$$

To predict the initial coordinates $\boldsymbol{x}_{j,k}^{(0)}$ for other atom types k, we apply the same alignment procedure but with different reference points $\{\boldsymbol{x}(\boldsymbol{b}_{1,k}), \cdots, \boldsymbol{x}(\boldsymbol{b}_{n,k})\}$ from the epitope.

3.2.2. Equivariant Structure Refinement

The initial coordinates are often inaccurate since the pairwise distances $\boldsymbol{D}_{i,j}$ are predicted independently for each residue pair and many higher-order interaction terms between different residues are ignored. For example, if two epitope residues i and j are far from each other, it is impossible for a paratope residue k to be near both i and j. To this end, we propose an iterative structure refinement procedure to incorporate these higher-order interactions. The refinement procedure must be equivariant because epitope coordinates are given and they can be rotated and translated arbitrarily. Inspired by force fields in physics, we propose to model the *force* between atoms rather than predicting absolute coordinates (Jin et al., 2021) to achieve equivariance.

Let $\mathcal{G}_{a,b}^{(t)}$ be the predicted paratope-epitope complex in the t-th iteration, with predicted paratope coordinates $\{\boldsymbol{x}_{i,k}^{(t)}\}$. In each iteration, we first encode $\mathcal{G}_{a,b}^{(t)}$ using our hierarchical encoder to compute residue-level and atom-level representations $\boldsymbol{h}^{(t)}(\cdot)$ for each residue. The force between two C$_\alpha$ atoms ($k=1$) is calculated based on their residue-level representation

$$\boldsymbol{g}_{i,j}^{(t)} = \boldsymbol{g}\big(\boldsymbol{h}^{(t)}(\boldsymbol{a}_i), \boldsymbol{h}^{(t)}(\boldsymbol{a}_j)\big) \cdot \big(\boldsymbol{x}_{i,1}^{(t-1)} - \boldsymbol{x}_{j,1}^{(t-1)}\big) \quad (8)$$

$$\boldsymbol{g}_{i,k}^{(t)} = \boldsymbol{g}\big(\boldsymbol{h}^{(t)}(\boldsymbol{a}_i), \boldsymbol{h}^{(t)}(\boldsymbol{b}_k)\big) \cdot \big(\boldsymbol{x}_{i,1}^{(t-1)} - \boldsymbol{x}(\boldsymbol{b}_{k,1})\big) \quad (9)$$

where $\boldsymbol{g}$ is a feed-forward neural network with a scalar output. In practice, we truncate the output of $\boldsymbol{g}(\cdot)$ so that two atoms do not clash after the update. We then update the C$_\alpha$ atom coordinates of each paratope residue by averaging the pairwise forces

$$\boldsymbol{x}_{i,1}^{(t)} = \boldsymbol{x}_{i,1}^{(t-1)} + \frac{1}{n}\sum_{j\neq i}\boldsymbol{g}_{i,j}^{(t)} + \frac{1}{m}\sum_{k}\boldsymbol{g}_{i,k}^{(t)} \quad (10)$$

For other atoms types ($k \neq 1$), we only model the force between atoms in the same residue. It is computationally expensive to calculate forces between all atom pairs due to the large number of atoms in the binding interface. Specifically, the force from atom $\boldsymbol{a}_{i,j}$ to $\boldsymbol{a}_{i,k}$ is calculated from their atom-level representations

$$\boldsymbol{g}_{ik,ij}^{(t)} = \boldsymbol{g}\big(\boldsymbol{h}^{(t)}(\boldsymbol{a}_{i,j}), \boldsymbol{h}^{(t)}(\boldsymbol{a}_{i,k})\big)\big(\boldsymbol{x}_{i,k}^{(t-1)} - \boldsymbol{x}_{i,j}^{(t-1)}\big) \quad (11)$$

Similarly, we truncate the force term if the atom distance is less than the Van der Waals radius. We update the atom coordinates by applying all the pairwise forces

$$\boldsymbol{x}_{i,k}^{(t)} = \boldsymbol{x}_{i,k}^{(t-1)} + \frac{1}{n_i}\sum_{j}\boldsymbol{g}_{ik,ij}^{(t)} \quad (12)$$

In summary, the C$_\alpha$ update step in Eq.(10) focuses on the global structure refinement and the atom update step in Eq.(12) focuses on the local side-chain arrangement. Following the proof from Satorras et al. (2021), it is easy to show that this coordinate update procedure is equivariant under epitope rotation and translation.

3.3. Decoder Module

Our decoder works together with the docking module to generate a paratope sequence autoregressively. The major difference between HSRN and standard RNN decoders is the representation of intermediate states. In standard RNNs, an intermediate state is a partial paratope sequence. In contrast, the intermediate state in HSRN is a paratope-epitope point cloud $\mathcal{G}_{a,b}^{(t)}$ predicted by our docking module, which provides more structural information than a partial sequence. The overall generation process is illustrated in Figure 3 and detailed below.

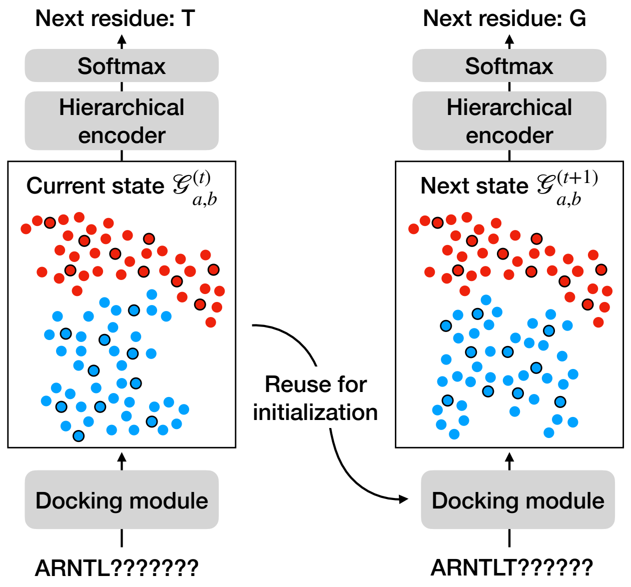

Figure 3. Illustration of the decoding process. In each generation step, HSRN first use the docking module to predict the paratope-epitope complex given the current paratope. It then encodes the predicted complex $\mathcal{G}_{a,b}^{(t)}$ with a hierarchical encoder and predict the amino acid type of the next residue. To avoid waste of computation, the docking module reuses the previous state to initialize the structure refinement process.

3.3.1. Initial State

To construct the initial state $\mathcal{G}_{a,b}^{(0)}$, we need to first guess the amino acid type for every paratope residue because the docking module takes an entire sequence as input. Therefore, we guess the amino acid type of each paratope residue

$$\boldsymbol{p}_i^{(0)} = \mathrm{softmax}\big(\boldsymbol{W}_0^\top \mathrm{FFN}(E_{\mathrm{pos}}(i))\big) \quad (13)$$

where $\mathrm{FFN}(\cdot)$ is a feed-forward network and $E_{\mathrm{pos}}(i)$ is the positional encoding (Vaswani et al., 2017) of paratope residue $\boldsymbol{a}_i$. $\boldsymbol{p}_i^{(0)}[k]$ is the probability of residue $\boldsymbol{a}_i$ being amino acid type k. Notably, the input to our docking module is a sequence $(\boldsymbol{p}_1^{(0)}, \cdots, \boldsymbol{p}_n^{(0)})$ with "soft" amino acid assignments. Based on this soft sequence, the docking module predicts the binding interface $\mathcal{G}_{a,b}^{(0)}$ as the initial state.

Due to the soft assignment, there are two slight modifications in the docking module during paratope generation. First, the node feature in the hierarchical encoder is defined as an expectation $\boldsymbol{f}(\boldsymbol{a}_i) = \sum_k \boldsymbol{p}_i^{(0)}[k]\boldsymbol{f}(k)$. Second, the docking module only predicts the four backbone atoms (N, C$_\alpha$, C, O) for the paratope since different amino acids have distinct side-chain atom configurations. Adding all side-chain atoms would require a hard assignment of all residue types before generation starts. Nonetheless, our hierarchical encoder still models all side-chain atoms in the antigen and epitope since they does not need to generated.

3.3.2. Intermediate States

In each generation step t, HSRN first use the docking module to predict the paratope-epitope complex given the current paratope sequence. It then encodes the predicted complex $\mathcal{G}_{a,b}^{(t)}$ with another hierarchical encoder to learn a hidden representation for each residue $\boldsymbol{h}^{(t)}(\boldsymbol{a}_i)$. Finally, we predict the amino acid type of the next residue $\boldsymbol{a}_t$ by

$$\boldsymbol{p}_t = \mathrm{softmax}\big(\boldsymbol{W}_s \boldsymbol{h}^{(t)}(\boldsymbol{a}_t)\big) \tag{14}$$

During training, we apply teacher forcing and set $\boldsymbol{a}_t$ to its ground truth. During testing, we sample an amino acid type $k \sim \boldsymbol{p}_t$ and set $\boldsymbol{a}_t = \mathrm{one\_hot}(k)$. The future residues remain as their initial guesses, i.e. the new paratope sequence becomes $(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_t, \boldsymbol{p}_{t+1}^{(0)}, \cdots, \boldsymbol{p}_n^{(0)})$.

Reusing docked structures. As the docking module applies L refinement iterations to finalize the paratope coordinates, a naive implementation costs in total nL refinement steps for generation. In fact, we can speed up the entire process by reusing previously docked structures instead of predicting the pairwise distance from scratch every time. In each step t, the docking module starts with paratope coordinates from $\mathcal{G}_{a,b}^{(t-1)}$ and perform only one refinement step to predict the next state $\mathcal{G}_{a,b}^{(t)}$ (Figure 3). By interleaving structure refinement with sequence generation, it only takes n docking steps to generate a sequence.

3.4. Training Loss

Docking loss. The training loss of our docking module contains two terms. The first term comes from initial distance prediction, which is calculated as the Huber loss between the predicted distance $\boldsymbol{D}_{i,j}$ and its ground truth. The second term comes from the iterative refinement procedure. We compute the pairwise distance $\|\boldsymbol{x}_{i,k}^{(t)} - \boldsymbol{x}_{j,l}^{(t)}\|$ between all atoms in the binding interface and calculate the Huber loss against its ground truth in every refinement step. For training stability, we do not propagate the gradient across multiple refinement steps during back-propagation.

Generation loss. For paratope generation, our training loss includes another cross entropy term between the predicted probabilities $\boldsymbol{p}_i^{(t)}$ and their ground truth amino acid types. During training, the docking module and decoder module are optimized simultaneously in one backward pass.

4. Experiments

We evaluate HSRN on two challenging tasks: paratope docking and generation. For simplicity, we only consider the CDR-H3 paratope of a heavy chain, which is the most flexible and important part of an antibody. The corresponding epitope is constructed by the following procedure. For each antigen residue $\boldsymbol{c}_i$, we first compute its shortest distance $d_i = \min_{j,k,l} \|\boldsymbol{x}(\boldsymbol{c}_{i,k}) - \boldsymbol{x}(\boldsymbol{a}_{j,l})\|$ to the paratope $\boldsymbol{a}$. We then rank the antigen residues based on d_i and take the top m residues as our epitope. The final epitope sequence $\boldsymbol{b}_1 \cdots \boldsymbol{b}_m = \boldsymbol{c}_{e_1} \cdots \boldsymbol{c}_{e_m}$ are sorted in the ascending order $e_1 < \cdots < e_m$, to ensure it's a subsequence of the antigen.

Table 1. Paratope docking results with epitope size $m = 20$.

Method	DockQ	Success
HDOCK	0.237	48.3%
HSRN (Ours)	**0.438**	**100%**

Table 2. HSRN docking performance with different epitope sizes.

	$m = 20$	$m = 40$	$m = 80$
DockQ	0.438	0.413	0.375
Success	100%	96.6%	89.7%

Data. Our training data comes from the Structural Antibody Database (SAbDab) (Dunbar et al., 2014), which contains around 3K antibody-antigen complexes after filtering structures without antigens and removing duplicates. The test set for this task comes from a paratope design benchmark compiled by Adolf-Bryfogle et al. (2018). It contains 60 antibody-antigen complexes with diverse antigen types. To measure the ability to generalize to novel paratopes, we remove any paratope sequences from the training set if it is similar to one of the test set paratopes (sequence identity above 70%). The training set contains 2777 complexes and the same dataset is used for docking and generation.

4.1. Paratope Docking

In this task, the input to our model is an antigen 3D structure with a specified epitope $\boldsymbol{b}_1 \cdots \boldsymbol{b}_m$ and a paratope sequence $\boldsymbol{a}_1 \cdots \boldsymbol{a}_n$. The goal is to predict the 3D coordinates of all paratope atoms, namely the 3D paratope-epitope complex. This task is also called *local docking*, as opposed to global docking which does not assume the epitope is given.

Baselines. We compare HSRN with HDOCK (Yan et al., 2020), a docking algorithm with state-of-the-art performance on protein docking benchmarks (Ganea et al., 2021). We choose HDOCK because it allows us to specify the receptor binding site and perform local docking. Since HDOCK is a rigid-body docking software that requires a paratope unbound structure as input, we use the IgFold algorithm (Ruffolo & Gray, 2022) to fold antibody heavy chains to provide paratope unbound structures. IgFold is a state-of-the-art antibody folding algorithm with comparable performance to AlphaFold (Jumper et al., 2021). Thus, we believe the combination of HDOCK and IgFold is a strong baseline for comparison.

Metrics. For all methods, we report the DockQ score (Basu

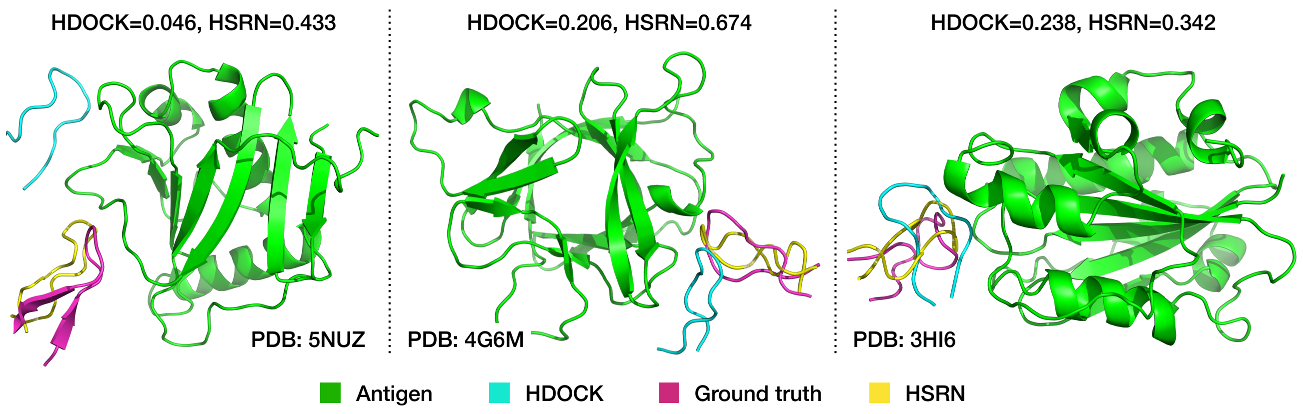

Figure 4. Comparison between the paratope structures predicted by HDOCK, HSRN, and their ground truth. Their DockQ scores are reported at the top of the figure. Best viewed in color.

Table 3. Ablation study of the docking module with $m = 20$. "Rand" means random initialization and "dist" means distance-based initialization.

Init	Refine	Hierarchical	DockQ	Success
rand	✓	✓	**0.438**	**100%**
rand	✓	×	0.357	89.7%
dist	✓	✓	0.377	87.9%
dist	✓	×	0.375	86.2%
dist	×	✓	0.303	65.5%

& Wallner, 2016), a standard evaluation metric for docking. DockQ is a weighted average of three terms: contact accuracy, interface RMSD, and ligand RMSD. A docked structure is considered acceptable if its DockQ score is above 0.23. We define the success rate of a model as the percentage of acceptable docked structures.

Hyperparameters. Each MPN in our hierarchical encoder contains four message passing layers with a hidden dimension of 256. The docking module performs eight iterations of structure refinement. All models are trained by an Adam optimizer for 20 epochs with 10% dropout.

Results. We first compare all methods with epitope size $m = 20$. As shown in Table 1, HSRN significantly outperforms HDOCK baseline in terms of average DockQ (0.237 vs 0.438) and success rate (48.3% vs 100%). In terms of speed, the average runtime for HSRN is less than 1 second per instance, while HDOCK takes more than a minute. The reason is that HDOCK enumerates and ranks all possible docking poses (over 1000 candidates) in a brute-force manner. HSRN instead treats docking as an optimization problem that requires only few iterations. As illustrated in Figure 4, our predicted paratope structure mostly agrees with the ground truth, while HDOCK sometimes place the paratope far from the correct location.

4.1.1. Ablation Study

Now we perform a series of ablation studies to investigate the importance of different model components.

Epitope size. First, we evaluate our method with increasing epitope sizes $m \in \{20, 40, 80\}$. Our results are summarized in Table 2. Indeed, the average DockQ gradually decreases as the docking task becomes increasingly difficult, but the overall success rate remains high.

Hierarchical encoder. Second, we find that removing the atom-level interface encoder hurts the performance in both initialization schemes (Table 3), with substantial performance decrease under the random coordinate initialization (DockQ: 0.438 vs 0.357). These results demonstrate the necessity of utilizing all-atom information for docking.

Structure refinement. Third, we remove the iterative structure refinement procedure and run the model under distance-based initialization.As shown in Table 3, the model performance degrades significantly without structure refinement (DockQ: 0.377 vs 0.303), validating our hypothesis that initial distance prediction is often inaccurate.

Initialization schemes. Lastly, we report our model performance under two coordinate initialization schemes (Table 3). Interestingly, HSRN performs much better with random initialization rather than distance-based initialization. However, as we will show in the next section, random initialization performs quite poorly in paratope design. This mixed result shows we need different initialization schemes for paratope docking and generation.

4.2. Paratope Generation

In this task, our model input contains only an antigen 3D structure with a specified epitope ***b***. The goal is to generate a CDR-H3 paratope that binds to a given epitope. There are two notable differences between our setup and previous work on paratope design (Jin et al., 2021). First, our model does not assume the framework region is given since our ultimate goal is *de novo* antibody design. Second, prior work does not consider the antigen structure.

Baselines. In this task, we consider the following state-of-the-art baselines for comparison.

- *RosettaAntibodyDesign* (RAbD) (Adolf-Bryfogle et al., 2018): We apply their *de novo* design protocol to design CDR-H3 paratopes. Starting from a random sequence, it performs 250 iterations of sequence design, docking, and energy minimization to find the best paratope with minimal interface energy. We cannot afford to run more iterations because it takes over 10 hours per instance.

- *Sequence decoder*: This model consists of an encoder and a decoder. The encoder is the same as our antigen MPN, whose input is a 3D antigen structure with a specified epitope. Its decoder is a RNN (Lei, 2021) that generates a paratope sequence autoregressively. The decoder employs an attention layer to extract information from encoded epitope representations so that the generation process is condition on the 3D epitope.

- *RefineGNN* (Jin et al., 2021) is a state-of-the-art generative model for antibody paratopes. Similar to HSRN, its decoder operates on a 3D paratope structure. Since the original RefineGNN is conditioned on the framework region rather than the antigen, we replace its encoder by our antigen MPN encoder and use the attention layer to extract information from the epitope representation. We call this modified model *CondRefineGNN* to make a distinction from its original architecture.

Metrics. Following Adolf-Bryfogle et al. (2018), we use amino acid recovery (AAR) as our metric. For a generated paratope $\tilde{\boldsymbol{a}}$, we define its AAR as $\sum_i \frac{1}{n}\mathbb{I}[\boldsymbol{a}_i = \tilde{\boldsymbol{a}}_i]$, the percentage of residues matching the corresponding residue in the ground truth $\boldsymbol{a}$. Specifically, for each epitope, we generate 1000 paratope sequences and choose the one with the best log-likelihood to calculate AAR score.

Given the critical role of contact residues in binding, we report another metric called contact AAR (CAAR). It is defined as $\sum_{i\in\mathcal{C}} \frac{1}{\mathcal{C}}\mathbb{I}[\boldsymbol{a}_i = \tilde{\boldsymbol{a}}_i]$, where $\mathcal{C}$ is a set of residues making contact with the epitope. A residue $\boldsymbol{a}_i$ makes contact with the epitope if $\min_{j,k,l} \|\boldsymbol{x}(\boldsymbol{a}_{i,k}) - \boldsymbol{x}(\boldsymbol{b}_{j,l})\| < 4.0$.

Hyperparameters. Each MPN in our hierarchical encoder contains three message passing layers with a hidden dimension of 256. For the docking module, we use distance-based initialization as default and explore random initialization in ablation studies. All models are trained by an Adam optimizer for 10 epochs with 10% dropout.

Results. As shown in Table 4, HSRN achieves the best AAR and CAAR score on this benchmark. Indeed, CondRefineGNN yields sub-optimal performance because it does not model the 3D paratope-epitope complex. The paratope 3D structure alone is not sufficient for reconstructing the true paratope. The sequence model yields even lower AAR because it does not model the paratope 3D structure at all. The output distribution of generated paratopes is illustrated in Figure 5.

Table 4. Paratope generation results with $m = 20$.

Method	AAR	CAAR
RAbD	28.6%	14.9%
Sequence decoder	32.2%	17.6%
CondRefineGNN	33.2%	19.7%
HSRN (Ours)	**34.1%**	**20.8%**
HSRN (random init)	31.0%	15.6%
HSRN (w/o atom MPN)	30.4%	19.0%

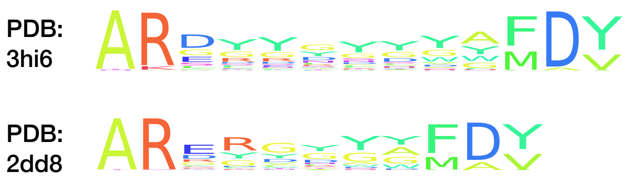

Figure 5. Sequence profile of generated paratopes for two different antigens: integrin LFA-1 (PDB: 3hi6) and SARS-CoV Spike receptor-binding domain (PDB: 2dd8).

4.2.1. Ablation study

We further perform two ablation studies to study the importance of initialization and hierarchical encoder in the context of paratope generation.

Hierarchical encoder. When removing the atom-level interface encoder, the generation performance degrades substantially in both initialization schemes (Table 4), with AAR decreasing from 34.1% to 30.4% under distance-based initialization. Indeed, both docking and generation experiments support the importance of hierarchical encoding.

Initialization. In contrast to the docking experiment, HSRN performs very poorly under the random initialization scheme (34.1% vs 31.0%, Table 4). This result is expected because the paratope structure looks almost random in the beginning and does not provide any useful information for predicting the first few amino acids.

5. Conclusion

In this paper, we have developed a hierarchical, equivariant architecture for paratope docking and design. Our model significantly outperforms existing docking methods with a wide margin while running orders of magnitudes faster. Indeed, there are various directions to improve our model. First, our model does not predict the binding energy of a docked/generated paratope. An interesting direction is to replace our autoregressive decoder with an energy-based model and use the learned energy function to quantify binding affinity. Second, our model assumes the epitope is given as input. We envision that HSRN can be combined with epitope prediction approaches (Del Vecchio et al., 2021) for end-to-end antibody engineering. Lastly, the atomic force learned by HSRN is estimated based on training data alone. We believe that incorporating domain knowledge from physics (e.g., Lennard-Jones potential) can further improve the generalization of our approach.

Acknowledgement

We would like to thank Rachel Wu and Jeremy Wohlwend for their valuable feedback on the manuscript. This work is supported by a grant from Sanofi on next generation molecular design, DTRA Discovery of Medical Countermeasures Against New and Emerging (DOMANE) threats program, DARPA Accelerated Molecular Discovery program, Eric and Wendy Schmidt Center at the Broad Institute, and Abdul Latif Jameel Clinic for Machine Learning in Health.

References

Adolf-Bryfogle, J., Kalyuzhniy, O., Kubitz, M., Weitzner, B. D., Hu, X., Adachi, Y., Schief, W. R., and Dunbrack Jr, R. L. RosettaAntibodyDesign (RAbD): A general framework for computational antibody design. *PLoS computational biology*, 14(4):e1006112, 2018.

Akbar, R., Robert, P. A., Weber, C. R., Widrich, M., Frank, R., Pavlović, M., Scheffer, L., Chernigovskaya, M., Snapkov, I., Slabodkin, A., et al. In silico proof of principle of machine learning-based antibody design at unconstrained scale. *BioRxiv*, 2021.

Basu, S. and Wallner, B. Dockq: a quality measure for protein-protein docking models. *PloS one*, 11(8): e0161879, 2016.

Cao, L., Coventry, B., Goreshnik, I., Huang, B., Park, J. S., Jude, K. M., Marković, I., Kadam, R. U., Verschueren, K. H., Verstraete, K., et al. Robust de novo design of protein binding proteins from target structural information alone. *bioRxiv*, 2021a.

Cao, Y., Das, P., Chenthamarakshan, V., Chen, P.-Y., Melnyk, I., and Shen, Y. Fold2seq: A joint sequence (1d)-fold (3d) embedding-based generative model for protein design. In *International Conference on Machine Learning*, pp. 1261–1271. PMLR, 2021b.

Crippen, G. M. and Havel, T. F. Stable calculation of coordinates from distance information. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 34(2):282–284, 1978.

Del Vecchio, A., Deac, A., Liò, P., and Veličković, P. Neural message passing for joint paratope-epitope prediction. *arXiv preprint arXiv:2106.00757*, 2021.

Dunbar, J., Krawczyk, K., Leem, J., Baker, T., Fuchs, A., Georges, G., Shi, J., and Deane, C. M. SAbDab: the structural antibody database. *Nucleic acids research*, 42 (D1):D1140–D1146, 2014.

Eismann, S., Townshend, R. J., Thomas, N., Jagota, M., Jing, B., and Dror, R. O. Hierarchical, rotation-equivariant neural networks to select structural models of protein complexes. *Proteins: Structure, Function, and Bioinformatics*, 89(5):493–501, 2021.

Evans, R., O'Neill, M., Pritzel, A., Antropova, N., Senior, A. W., Green, T., Žídek, A., Bates, R., Blackwell, S., Yim, J., et al. Protein complex prediction with alphafold-multimer. *Biorxiv*, 2021.

Fout, A., Byrd, J., Shariat, B., and Ben-Hur, A. Protein interface prediction using graph convolutional networks. *Neural Information Processing Systems*, 2017.

Ganea, O.-E., Huang, X., Bunne, C., Bian, Y., Barzilay, R., Jaakkola, T., and Krause, A. Independent SE(3)-equivariant models for end-to-end rigid protein docking. *arXiv preprint arXiv:2111.07786*, 2021.

Ingraham, J., Riesselman, A., Sander, C., and Marks, D. Learning protein structure with a differentiable simulator. In *International Conference on Learning Representations*, 2018.

Ingraham, J., Garg, V. K., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. *Neural Information Processing Systems*, 2019.

Jin, W., Wohlwend, J., Barzilay, R., and Jaakkola, T. Iterative refinement graph neural network for antibody sequence-structure co-design. *arXiv preprint arXiv:2110.04624*, 2021.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

Karimi, M., Zhu, S., Cao, Y., and Shen, Y. De novo protein design for novel folds using guided conditional wasserstein generative adversarial networks. *Journal of Chemical Information and Modeling*, 60(12):5667–5681, 2020.

Kozakov, D., Hall, D. R., Xia, B., Porter, K. A., Padhorny, D., Yueh, C., Beglov, D., and Vajda, S. The cluspro web server for protein–protein docking. *Nature protocols*, 12 (2):255–278, 2017.

Lapidoth, G. D., Baran, D., Pszolla, G. M., Norn, C., Alon, A., Tyka, M. D., and Fleishman, S. J. AbDesign: A n algorithm for combinatorial backbone design guided by natural conformations and sequences. *Proteins: Structure, Function, and Bioinformatics*, 83(8):1385–1406, 2015.

Lei, T. When attention meets fast recurrence: Training language models with reduced compute. *arXiv preprint arXiv:2102.12459*, 2021.

Liu, G., Zeng, H., Mueller, J., Carter, B., Wang, Z., Schilz, J., Horny, G., Birnbaum, M. E., Ewert, S., and Gifford, D. K. Antibody complementarity determining region design using high-capacity machine learning. *Bioinformatics*, 36(7):2126–2133, 2020.

Norn, C., Wicky, B. I., Juergens, D., Liu, S., Kim, D., Tischer, D., Koepnick, B., Anishchenko, I., Baker, D., and Ovchinnikov, S. Protein sequence design by conformational landscape optimization. *Proceedings of the National Academy of Sciences*, 118(11), 2021.

Ruffolo, J. A. and Gray, J. J. Fast, accurate antibody structure prediction from deep learning on massive set of natural antibodies. *Biophysical Journal*, 121(3):155a–156a, 2022.

Saka, K., Kakuzaki, T., Metsugi, S., Kashiwagi, D., Yoshida, K., Wada, M., Tsunoda, H., and Teramoto, R. Antibody design using lstm based deep generative model from phage display library for affinity maturation. *Scientific reports*, 11(1):1–13, 2021.

Satorras, V. G., Hoogeboom, E., and Welling, M. E(n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021.

Shin, J.-E., Riesselman, A. J., Kollasch, A. W., McMahon, C., Simon, E., Sander, C., Manglik, A., Kruse, A. C., and Marks, D. S. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):1–11, 2021.

Somnath, V. R., Bunne, C., and Krause, A. Multi-scale representation learning on proteins. *Advances in Neural Information Processing Systems*, 34, 2021.

Strokach, A., Becerra, D., Corbi-Verge, C., Perez-Riba, A., and Kim, P. M. Fast and flexible design of novel proteins using graph neural networks. *BioRxiv*, pp. 868935, 2020.

Tischer, D., Lisanza, S., Wang, J., Dong, R., Anishchenko, I., Milles, L. F., Ovchinnikov, S., and Baker, D. Design of proteins presenting discontinuous functional sites using deep learning. *bioRxiv*, 2020.

Townshend, R., Bedi, R., Suriana, P., and Dror, R. End-to-end learning on 3d protein structure for interface prediction. *Advances in Neural Information Processing Systems*, 32:15642–15651, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Yan, Y., Tao, H., He, J., and Huang, S.-Y. The hdock server for integrated protein–protein docking. *Nature protocols*, 15(5):1829–1852, 2020.

Yuan, M., Wu, N. C., Zhu, X., Lee, C.-C. D., So, R. T., Lv, H., Mok, C. K., and Wilson, I. A. A highly conserved cryptic epitope in the receptor binding domains of sars-cov-2 and sars-cov. *Science*, 368(6491):630–633, 2020.

A. Model Architecture Details

Amino acid features. Each amino acid is represented by six features: polarity $f_p \in \{0, 1\}$, hydropathy $f_h \in [-4.5, 4.5]$, volume $f_v \in [60.1, 227.8]$, charge $f_c \in \{-1, 0, 1\}$, and whether it is a hydrogen bond donor $f_d \in \{0, 1\}$ or acceptor $f_a \in \{0, 1\}$. For hydropathy and volume features, we expand it into radial basis with interval size 0.1 and 10, respectively. As a result, the amino acid feature dimension is 112.

Residue-level edge features. For each residue $\boldsymbol{a}_i$, we define its local coordinate frame $\boldsymbol{O}_i = [\boldsymbol{c}_i, \boldsymbol{n}_i, \boldsymbol{c}_i \times \boldsymbol{n}_i]$ as

$$\boldsymbol{u}_i = \frac{\boldsymbol{x}_i - \boldsymbol{x}_{i-1}}{\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|}, \quad \boldsymbol{c}_i = \frac{\boldsymbol{u}_i - \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i - \boldsymbol{u}_{i+1}\|}, \quad \boldsymbol{n}_i = \frac{\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|} \tag{15}$$

Based on the local frame, we compute the following edge features

$$\boldsymbol{f}(\boldsymbol{a}_i, \boldsymbol{a}_j) = \Big(E_{\text{pos}}(i-j), \quad \text{RBF}(\|\boldsymbol{x}_{i,1} - \boldsymbol{x}_{j,1}\|), \quad \boldsymbol{O}_i^\top \frac{\boldsymbol{x}_{j,1} - \boldsymbol{x}_{i,1}}{\|\boldsymbol{x}_{i,1} - \boldsymbol{x}_{j,1}\|}, \quad \boldsymbol{q}(\boldsymbol{O}_i^\top \boldsymbol{O}_j) \Big). \tag{16}$$

The edge feature $\boldsymbol{f}_{ij}$ contains four parts. The positional encoding $E_{\text{pos}}(i-j)$ encodes the relative distance between two residues in an antibody sequence. The second term $\text{RBF}(\cdot)$ is a *distance* encoding lifted into radial basis. The third term in $\boldsymbol{f}_{ij}$ is a *direction* encoding that corresponds to the relative direction of $\boldsymbol{x}_j$ in the local frame of residue i. The last term $\boldsymbol{q}(\boldsymbol{O}_i^\top \boldsymbol{O}_j)$ is the *orientation* encoding of the quaternion representation $\boldsymbol{q}(\cdot)$ of the spatial rotation matrix $\boldsymbol{O}_i^\top \boldsymbol{O}_j$.

MPN architecture. Our MPN contains L message passing layers. Let $\mathcal{N}_i$ be the set of neighbor nodes for residue $\boldsymbol{a}_i$. Each MPN layer consists of a standard message passing step followed by an aggregation step with residual connection, where $\boldsymbol{h}_0(\boldsymbol{a}_i) = \boldsymbol{z}(\boldsymbol{a}_i)$.

$$\boldsymbol{h}_{l+1}(\boldsymbol{a}_i) = \boldsymbol{h}_l(\boldsymbol{a}_i) + \sum_{j \in \mathcal{N}_i} \text{FFN}\big(\boldsymbol{h}_l(\boldsymbol{a}_i), \boldsymbol{h}_l(\boldsymbol{a}_j), \boldsymbol{f}(\boldsymbol{a}_j), \boldsymbol{f}(\boldsymbol{a}_i, \boldsymbol{a}_j)\big) \ \ (0 \le l < L) \tag{17}$$