ROTAMER DENSITY ESTIMATORS ARE UNSUPERVISED LEARNERS OF THE EFFECT OF MUTATIONS ON PROTEIN-PROTEIN INTERACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein-protein interactions play a fundamental role in a broad range of biological processes. Predicting the effect of amino acid mutations on binding is crucial to protein engineering. Traditional biophysical and statistical methods have dominated the area for years, but they depend heavily on expert prior and face the trade-off between efficiency and accuracy. Recent success in deep learning for proteins has made data-driven approaches more appealing than ever. Nevertheless, the major challenge is the scarcity of experimental mutational data annotated with the change in binding affinity. In this work, we demonstrate that mutational effects on binding can be predicted by the change in conformational flexibility of the protein-protein interface. We propose a flow-based generative model to estimate the probability distribution of conformation (named Rotamer Density Estimator, RDE) and use entropy as the measure of flexibility. The model is trained solely with protein structures and does not require the supervision of the experimental values of changes in binding affinities. Further, the unsupervised representations extracted by the model can be used for prediction even more accurately using simple downstream neural networks. The proposed method outperforms empirical energy functions and other machine learning-based approaches.

1 INTRODUCTION

Proteins rarely act alone but usually interact with other proteins to orchestrate biological processes (Alberts & Miake-Lye, 1992; Kastritis & Bonvin, 2013). For example, antibodies, a kind of immune system protein, recognize pathogens by binding to proteins on their surface and elicit immune responses by interacting with the receptor protein of immune cells (Lu et al., 2018). Since protein-protein interactions determine a large number of biological functions, developing methods to modulate protein-protein interactions is critical. A typical way to modulate protein-protein interactions is to mutate amino acids on the interface — some mutations improve the strength of binding while others weaken and even disrupt the interaction (Gram et al., 1992; Barderas et al., 2008). Biologists choose either to increase or decrease the binding strength depending on specific goals. For example, if one would like to enhance the effect of a neutralizing antibody against a virus, it is usually necessary to increase the binding strength between the antibody and the viral protein. However, as the combinatorial space of amino acid mutations is large, it is not always feasible or affordable to conduct wet-lab assays to test all the viable mutations. Therefore, computational approaches are needed to guide the identification of desirable mutations via predicting the mutational effect on binding strength measured by *the change in binding free energy* ($\Delta\Delta G$).

Traditional computational approaches are mainly based on biophysics and statistics (Schymkowitz et al., 2005; Park et al., 2016; Alford et al., 2017). Though having dominated the area for years, their limitations are non-negligible. In general, biophysics-based methods face the trade-off between efficiency and accuracy as they rely on sampling from energy functions. Statistical methods are efficient but their capacity is limited by the descriptors considered in the model. Both biophysics and statistics-based methods rely heavily on human knowledge. Thus, they can hardly benefit from the fast-growing of available protein structures. These limitations mark that predicting the effect of mutation on protein binding remains an open problem.

Recently, deep learning has gained tremendous success in modeling proteins, making data-driven methods more appealing than ever (Rives et al., 2019; Jumper et al., 2021). Nevertheless, challenges exist for developing deep learning-based models to predict mutational effects on protein-protein binding. The major challenge is the scarcity of experimental data — only a few thousands of protein mutations annotated with the change in binding affinity are publicly available (Geng et al., 2019b). This hinders supervised learning as the insufficiency of training data tends to cause over-fitting. Another difficulty is the absence of the structure of mutated protein-protein complexes. Mutating amino acids on a protein complex leads to changes on sidechain conformations (rotamers) (Najmanovich et al., 2000; Gaudreault et al., 2012). They account for the change in binding free energy but we do not have the knowledge of how exactly the conformation changes upon mutation.

In this work, we take inspiration from the observation that finds its root in thermodynamics: Protein-protein binding generally leads to entropy loss on the interface and more entropy loss implies higher binding affinity [1] (Brady & Sharp, 1997; Cole & Warwicker, 2002; Kastritis & Bonvin, 2013). Intuitively speaking, amino acids on the interface become less flexible (lower entropy) after they get contact with other proteins due to the geometrical and physical restraints imposed by the binding partner (Figure 1). Higher binding affinity implies better complementarity between the two parts, thus higher rigidity.

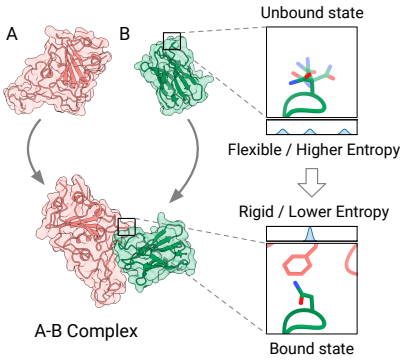

Figure 1: The conformational flexibility of the interface generally decreases upon binding.

Therefore, *we seek to predict the change in binding free energy via estimating the change in conformational flexibility*. Specifically, our method consists of three main parts. The first part is a conditional generative model built upon normalizing flows on torus for estimating the density of amino acid sidechain conformations (rotamers) given the structural environment of the amino acid, named Rotamer Density Estimator (RDE). We choose normalizing flows because it allows exact likelihood computation, which is indispensable for the second part of the method — entropy estimation. We estimate the entropy of the rotamer distribution predicted by the model and use it as the measurement of conformational flexibility. Finally, we use the entropy of protein-protein interfaces at different states to predict the change in binding free energy ($\Delta\Delta G$). In addition to directly using the entropy to predict $\Delta\Delta G$, we also use simple neural networks to extract $\Delta\Delta G$ from the unsupervised representations produced by the RDE.

Our method provides a solution to the aforementioned challenges. Training the rotamer density estimator requires only protein structures. Thus, it is an unsupervised learner of the effect of mutations on binding and it alleviates the difficulty rising from the scarcity of annotated mutation data. In addition, our method does not require the structure of the mutated protein as input. Rather, it treats mutated structures as latent variable and the rotamer density estimator is an approximator of the latent distribution. *Our method outperforms empirical energy functions and machine learning models for $\Delta\Delta G$ prediction*. In addition, as a generative model for rotamers, *the RDE predicts sidechain conformations accurately*.

2 Related Work

2.1 Mutational Effect Prediction for Protein-Protein Binding

Traditional approaches to predicting the effect of mutation on protein binding can be roughly divided into two classes: biophysical methods and statistical methods. Biophysical methods focus on modeling inter-atomic interactions, e.g. hydrogen bonding, electrostatic forces, etc., with mechanical and statistical energy functions. These methods use their underlying energy function to sample conformations of the mutated protein complex and predict the change in binding free energy upon mutations (Schymkowitz et al., 2005; Park et al., 2016; Alford et al., 2017; Steinbrecher et al., 2017). Statistical methods are based on feature engineering. Descriptors that summarize the geometric, physical, evolutionary, and motif properties of proteins are used to fit the statistical model that

[1] The free energy of binding can be decomposed into two terms: $\Delta G_{\mathrm{a}} = \Delta G_{\mathrm{bond}} + \Delta G_{\mathrm{entropy}}$, where the first term denotes bonded energy changes (for most protein-protein interactions this term has little effect as no bonds are formed or broken) and the second term denotes entropy changes (Kastritis & Bonvin, 2013).

predicts the effect of mutations on binding free energy (Geng et al., 2019a; Zhang et al., 2020). Traditional methods generally face the trade-off between speed and accuracy due to the time-consuming sampling process. Sophisticated energy functions and feature engineering underlying these methods depend heavily on human knowledge, thus limiting their pace to improve with the fast-growing of available protein structures.

Recently, deep learning-based approaches to predicting mutational effects on protein binding have emerged. We group them into three categories: end-to-end models, pre-training-based models, and unsupervised models. End-to-end models take both mutant and wild-type (not mutated) protein structures along with other features as input and directly predict the difference in binding free energy. End-to-end models achieve satisfactory correlation between prediction and ground truth on the whole SKEMPI benchmark consisting of many different structures (Shan et al., 2022), but the per-structure correlation, which is more relevant to practical use, is still low. In attempt to settle the issue of data scarcity, another line of work proposes to pre-train a feature extraction network and then use regression models to predict the effect of mutations based on the learned features (Liu et al., 2021; Yang et al., 2022). Nevertheless, these pre-training strategies do not capture well the foundation of protein-protein interactions. More recently, the success of protein language models in has drawn interests in adopting mask-predict (BERT) paradigm to protein 3D structures (Wang et al., 2018; Shroff et al., 2020; Yang et al., 2022; Zhang et al., 2022). These methods partially mask amino acid types on a given protein backbone and use neural networks to recover the masked information. It has been reported that the difference in the probability of amino acid types before and after mutation show correlation to the change in binding free energy (Yang et al., 2022). Hence, they can serve as unsupervised predictors of the mutational effects on binding.

2.2 Mutational Effect Prediction for Single Proteins

Methods for predicting mutational effects for single proteins are either structure-based or sequence-based (evolution-based). Structure-based approaches can also be divided into biophysical methods, statistical methods, and deep learning-based methods. They aim at predicting thermal stability or fitness of the protein rather than binding free energy between proteins (Schymkowitz et al., 2005; Park et al., 2016; Alford et al., 2017). The mutational effects of single proteins can also be predicted using only sequences via mining its evolutionary history. Traditionally, this is done by performing statistics on multiple sequence alignments (MSAs), which are constructed by searching from large-scale sequence databases (Hopf et al., 2017; Riesselman et al., 2018; Rao et al., 2021; Luo et al., 2021). Recent studies show that proteins language models (PLMs) trained on large sequence databases are capable of evaluating mutations without MSAs (Meier et al., 2021).

We would like to highlight that, *sequence-based (evolution-based) methods for single proteins are not applicable to protein-protein interactions* due to the lacking of evolutionary information in most cases — protein-protein interactions involve two or more chains. The chains might belong to different species (e.g. host and virus). Sometimes inter-chain co-evolution simply does not happen (e.g. immune response where effective antibodies are produced to induce rapid clearance of pathogens, leaving no time for the pathogen to evolve). These make it infeasible to predict mutational effects via mining sequence databases with existing powerful tools such as multiple sequence alignments, protein language models, etc. Hence, currently, effective ways for predicting mutational effects on binding are based on structures rather than sequences alone.

3 Method

3.1 Overview and Preliminaries

Overview Our method consists of three parts. The core component is the rotamer density estimator (RDE), a conditional normalizing flow that models the probability density of sidechain conformations (rotamers) given the amino acid type and environments (Section 3.2). Next is the algorithm for estimating the entropy of the distribution parameterized by the normalizing flow (Section 3.3). Finally, we show how to use the entropy of the mutated and wild-type protein-protein interfaces at both bound and unbound states to estimate *the change in binding free energy* ($\Delta\Delta G$) upon mutation, and how to use neural networks to predict $\Delta\Delta G$ more accurately using the unsupervised representations from the RDE (Section 3.4).

Definitions and Notations A *protein-protein complex* is a multi-chain protein structure that can be divided into two groups. Each group contains at least one protein chain and each chain consists of multiple amino acids. For a protein complex containing n amino acids, we number them from 1 to n.

The two groups of the complex can be represented by two disjoint sets of indices $A, B \subset \{1 \dots n\}$ An amino acid is characterised by its type, position, orientation, and sidechain conformation. We denote the type, position, and orientation of the i-th ($i \in \{1 \dots n\}$) amino acid as $a_i \in \{1 \dots 20\}$, $\boldsymbol{p}_i \in \mathbb{R}^3$, and $\boldsymbol{O}_i \in SO(3)$ respectively. The sidechain conformation of the amino acid is called *rotamer*. As the conformational degree of freedom of the sidechain is defined by rotatable bonds, a rotamer can be sufficiently parameterized by torsional angles w.r.t. the rotatable bonds. The number of torsional angles vary between 0 to 4 depending on the amino acid type. For an amino acid with d torsional angles, we denote the k-th ($k \in \{1 \dots 4\}$) torsional angle by $\chi_i^{(k)} \in [0, 2\pi)$. Collectively, all the torsional angles are denoted by a vector $\boldsymbol{\chi}_i = (\chi_i^{(k)})_{k=1}^d$. Using the language of geometry, an angle can represented by a point on the unit circle $\mathbb{S}^1$. A vector consisting of d angular values reside on the product of d unit circle, known as the d-dimensional torus $\mathbb{T}^D = (\mathbb{S}^1)^D$.

In this work, the first goal is to model the conditional probability density of rotamers, given the type, position, orientation, and prior rotamer of itself and other amino acids: $p(\boldsymbol{\chi}_i \mid \{a_j, \boldsymbol{p}_j, \boldsymbol{O}_j, \tilde{\boldsymbol{\chi}}_j\}_{j=1}^n)$. The prior rotamers $\tilde{\boldsymbol{\chi}}_j$ are inaccurate or unknown in many cases. For example, if we mutate some amino acids in the protein complex, the rotamers of the mutated amino acids are unknown, and the rotamers of amino acids nearby the mutated ones are inaccurate because they are affected by the mutation. The probability density is defined over the d-dimensional torus $\mathbb{T}^D = (\mathbb{S}^1)^D$, and we show below our proposed flow-based architecture to model the density.

3.2 Rotamer Density Estimator

The rotamer density estimator (RDE) is designed to model the conditional distribution of the rotamer of the i-th amino acid given the information of itself and other amino acids: $p(\boldsymbol{\chi}_i \mid \{a_j, \boldsymbol{p}_j, \boldsymbol{O}_j, \tilde{\boldsymbol{\chi}}_j\}_{j=1}^n)$. In this section, we first introduce the encoder network that produces hidden representations for each amino acid by taking into account its environment $\{a_i, \boldsymbol{p}_i, \boldsymbol{O}_i, \tilde{\boldsymbol{\chi}}_i\}_{i=1}^n$. Next, we formulate the conditional normalizing flow on $\mathbb{S}^1$ for rotamers with only 1 torsional angle. Based on the flow on $\mathbb{S}^1$, we further formulate the flows on d-dimensional torus $\mathbb{T}^D$ ($D > 1$) for rotamers with more than 1 torsional angle.

Encoder Network The encoder network starts with two multi-layer perceptrons (MLPs) that generate embeddings for each single amino acid and each pair of amino acids respectively. The MLP for single amino acid encodes the amino acid type, backbone dihedral angles, and local atom coordinates of each amino acid into a vector $\boldsymbol{e}_i$ ($i = 1 \dots n$). The other MLP for amino acid pairs mainly encodes the distance and relative position between two amino acids and we denote a pair embedding vector as $\boldsymbol{z}_{ij}$ ($i, j = 1 \dots n$). Next, we use a self-attention-based network invariant to rotation and translation (Tubiana et al., 2022; Jumper et al., 2021) to transform the single embeddings $\{\boldsymbol{e}_i\}$ and pair embeddings $\{\boldsymbol{z}_{ij}\}$ into hidden representations $\{\boldsymbol{h}_i\}$. The hidden representation $\boldsymbol{h}_i$ is aimed at capturing the information of both the i-th amino acid itself and its structural environments. It serves as an encoding of the condition $\{a_j, \boldsymbol{p}_j, \boldsymbol{O}_j, \tilde{\boldsymbol{\chi}}_j\}_{j=1}^n$ for the probability density w.r.t. $\boldsymbol{\chi}_i$.

Conditional Flow on $\mathbb{S}^1$ We model the distribution of rotamer with 1 torsional angle with conditional normalizing flows on $\mathbb{S}^1$ (Rezende et al., 2020). A normalizing flow is basically a bijective function. We choose to parameterize $\mathbb{S}^1$ by an angle $\theta \in [0, 2\pi]$ and construct a bijective function on $[0, 2\pi]$ which is equivalent to a bijective on $\mathbb{S}^1$. A common method for constructing a bijective function is ensuring the strict monotonicity. We adopt this method and construct a strictly monotonically increasing function on $[0, 2\pi]$, denoted by $f : [0, 2\pi] \to [0, 2\pi]$. Due to the periodicity of angular values, 0 and 2π are the same point. In order to preserve the continuity at the two ends, we have to additionally ensure that i. $f(0) = 0$, ii. $f(2\pi) = 2\pi$, and iii. $f'(0) = f'(2\pi)$. We resort to the rational quadratic spline flow (Durkan et al., 2019; Rezende et al., 2020) to construct a function that satisfies all the requirements. Specifically, the spline-based bijective function is a piece-wise function that contains K pieces delimited by $K + 1$ knots, and each piece takes the following form:

$$f_k(x|x_{k,k+1}, y_{k,k+1}, \delta_{k,k+1}) = y_k + \frac{(y_{k+1} - y_k)\left[s_k \xi_k^2(x) + \delta_k\left(1 - \xi_k(x)\right)\xi_k(x)\right]}{s_k + \left[\delta_{k+1} + \delta_k - 2s_k\right]\left(1 - \xi_k(x)\right)\xi_k(x)}, \tag{1}$$

$$\text{where } s_k = \frac{y_{k+1} - y_k}{x_{k+1} - x_k}, \text{ and } \xi(x) = \frac{x - x_k}{x_{k+1} - x_k} \quad (x \in [x_k, x_{k+1}]). \tag{2}$$

The spline is parameterized by the coordinates and derivatives of the $K + 1$ knots denoted by x_k, y_k and δ_k ($k = 1 \dots K + 1$). To fulfill the requirement of monotonicity, continuity, and periodicity, we impose the following constraints on the knots: i. $0 = x_1 < x_2 < \dots < x_{K+1} = 2\pi$, ii.

$0 = y_1 < y_2 < \ldots < y_{K+1} = 2\pi$, iii. $\delta_k > 0 (k = 1 \ldots K+1)$, and iv. $\delta_1 = \delta_{K+1}$. The parameters are produced by transforming the hidden representation $\boldsymbol{h}_i$ of the amino acid using neural networks, such that the probability distribution defined by the bijective is conditional on the amino acid and its environment. Therefore, we may also denote the spline by $f(x|\boldsymbol{h}_i)$.

We choose the uniform distribution on $[0, 2\pi]$ as the base distribution, and denote it by $p_z(z) = \frac{1}{2\pi}$ ($z \in [0, 2\pi]$). We let f be the mapping from the target distribution to the base distribution. According to the change-of-variable formula, the target rotamer density of the i-th amino acid is:

$$\log p(x|\boldsymbol{h}_i) = \log p_z\left(f(x)\right) + \log |f'(x|\boldsymbol{h}_i)| \tag{3}$$

$$= -\log 2\pi + \log |f'(x|\boldsymbol{h}_i)| \,. \tag{4}$$

The derivative f' can computed analytically according to Eq.1. The model is illustrated in Figure 2A.

In practice, we stack multiple bijectives to enable more complex transform. The derivative of the composite can be computed efficiently by applying the chain rule. At inference time, the inverse mapping $f^{-1}(y)$ can be computed efficiently (Rezende et al., 2020). To find the solution of $f^{-1}(y)$, the first step is to locate the unique bin that contains y. Assuming y belongs to the k-th bin, finding its corresponding x amounts to finding the root of the quadratic equation $f_k(x|x_{k,k+1}, y_{k,k+1}, \delta_{k,k+1}) = y$ in $[x_k, x_{k+1}]$, whose solution is closed-form.

Conditional Flow on $\mathbb{T}^D$ Rotamers with d torsional angles can be viewed as points on the d-dimensional torus. The D-dimensional torus is the product of D circles $\mathbb{S}^1$, i.e. $\mathbb{T}^D = \mathbb{S}^1 \times \cdots \times \mathbb{S}^1$. Therefore, we can model the distribution on $\mathbb{T}^D$ by modeling the joint distribution of D variables on $\mathbb{S}^1$. In specific, we adopt the coupling layer technique to model the joint distribution (Dinh et al., 2016). On each coupling layer, we update one dimension using the bijective for $\mathbb{S}^1$, keeping the other $D-1$ dimensions fixed and using them along with the hidden representation of the amino acid as the condition to parameterize the bijective (Figure 2B):

$$g_d(\boldsymbol{x}|\boldsymbol{h}_i)[j] = \begin{cases} f(x_j|\boldsymbol{x}_{\setminus j}, \boldsymbol{h}_i), & j = d \mod D \\ x_j, & j \neq d \mod D \end{cases}, \tag{5}$$

where d is the dimension to update. The coupling layer g_d preserves invertibility and has closed-form inversion. The determinant of its Jacobian is equal to the derivative of $f(x_j|\boldsymbol{x}_{\setminus j}, \boldsymbol{h}_i)$. We stack multiple spline-based coupling layers to ensure that all the dimensions are updated at least once: $g = g_1 \circ g_2 \circ \cdots \circ g_L$ ($L \geq D$).

We choose the uniform distribution as the base distribution, denoted by $p_{\boldsymbol{z}}(\boldsymbol{z}) = (\frac{1}{2\pi})^D$ and let g map the target distribution to the base distribution. The log-likelihood is computed by the change-of-variable rule:

$$\log p(\boldsymbol{x}|\boldsymbol{h}_i) = \log p_{\boldsymbol{z}}(g(\boldsymbol{x})) + \log |\det \nabla_{\boldsymbol{x}} g(\boldsymbol{x})| \tag{6}$$

$$= -D \log 2\pi + \sum_{l=1}^{L} \log \left| f_l'(x_{d(l)}|\boldsymbol{x}_{\setminus d(l)}, \boldsymbol{h}_i) \right| . \tag{7}$$

Training Objective To train the rotamer density estimator, we minimize the negative log-likelihood of native (ground truth) rotamers:

$$\min \mathbb{E}_{(\boldsymbol{\chi}_i, \mathcal{S}) \sim p_{\text{data}}} \left[-\log p\left(\boldsymbol{\chi}_i | \boldsymbol{h}_i(\mathcal{S})\right) \right], \tag{8}$$

where $\mathcal{S} = \{a_j, \boldsymbol{p}_j, \boldsymbol{O}_j, \tilde{\boldsymbol{\chi}}_j\}_{j=1}^N$ is a protein structure sampled from the dataset. Since the model is designed for mutation analysis, the prior rotamers should emulate the rotamers after some amino acids are mutated. Thus, we mask the rotamers of a random portion of amino acids and add noise to the rotamers of the neighbors of the masked ones. The negative log-likelihood training objective function is evaluated on the masked amino acids and a random part of the perturbed amino acids.

3.3 Rotamer Entropy Estimation

The rotamer density estimator models the distribution of possible rotamers of an amino acid. Therefore, the entropy of the rotamer distribution is a natural measurement of conformational flexibility (Brady & Sharp, 1997):

$$H(\mathcal{S}, i) = \mathbb{E}_{\boldsymbol{\chi} \sim p} \left[-\log p(\boldsymbol{\chi} | \boldsymbol{h}_i(\mathcal{S})) \right] . \tag{9}$$

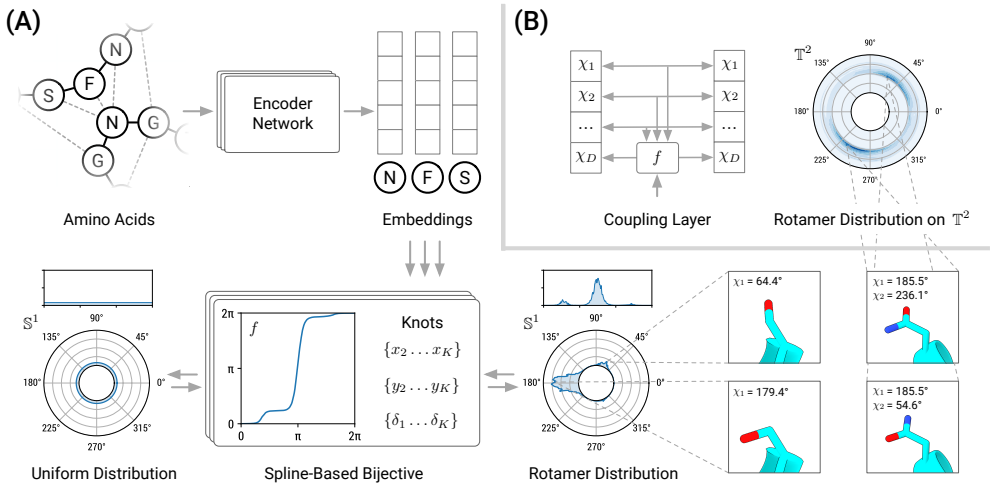

Figure 2: **(A)** The overall architecture of the rotamer density estimator (RDE) for estimating distributions of rotamers with 1 torsional angle. **(B)** Invertible coupling layers alternating among different dimensions enable modeling distributions of rotamers with multiple torsional angles.

We use the stochastic method to estimate the entropy. First, we sample a set of rotamers from the distribution using the inversion of the flows (Eq.1, 5). Then, we compute the negative log-probability of the samples and take the average as an estimation of the entropy. These two steps can be done efficiently thanks to the capability of computing exact likelihood of normalizing flows.

For amino acids without rotatable sidechains (alanine, glycine, and proline), we consider their conformational entropy constant. However, we can still estimate the entropy of amino acids nearby them to evaluate their impact on conformational flexibility.

3.4 Mutational Effect ($\Delta\Delta G$) Prediction

Assume that we have a protein-protein complex $\mathcal{W}_{LR} = \{a_i, \boldsymbol{p}_i, \boldsymbol{O}_i, \tilde{\boldsymbol{\chi}}_i\}$ consisting of N amino acids. M of them belong to the first group $L = \{1, \dots, M\}$, and the remaining $N-M$ belong to the second group $R = \{M+1, \dots, N\}$. In the unbound state, we denote the two separated structures by $\mathcal{W}_L$ and $\mathcal{W}_R$ respectively. We call group L *ligand* as it contains mutations and call group R *receptor*. Consider that we mutate m amino acids in the group L, numbered by $\{1, \dots, m\} \subset L$. The mutated structure in the bound state is denoted by $\mathcal{M}_{LR} = \{\tilde{a}_i, \boldsymbol{p}_i, \boldsymbol{O}_i, \varnothing\}_{i=1}^{m} \cup \{a_i, \boldsymbol{p}_i, \boldsymbol{O}_i, \tilde{\boldsymbol{\chi}}_i\}_{\text{unmutated}}$, where $\tilde{a}_i$ is the mutated amino acid type and $\varnothing$ indicates that the rotamers of the mutated amino acids are unknown. The two groups in the unbound state are denoted by $\mathcal{M}_L$ and $\mathcal{M}_R$ respectively. Note that $\mathcal{M}_R = \mathcal{W}_R$ if no mutations are in the receptor group.

Linear Predictor The *entropy loss upon binding* of a protein complex $\mathcal{S}$ ($\mathcal{S} = \mathcal{W}, \mathcal{M}$) is defined by the difference of entropy between the bound state and the unbound state (Kastritis & Bonvin, 2013). We approximate the entropy loss $\Delta H_{\mathcal{S}}$ by linear model using our estimated entropy:

$$\Delta H_{\mathcal{S}} = \Big[w_{\mathcal{S}L}^{\text{bound}} \underbrace{\sum_{i \in L} (H(\mathcal{S}_{LR}, i) + E(a_i))}_{H_{\mathcal{S}L}^{\text{bound}} \text{ bound ligand}} + w_{\mathcal{S}R}^{\text{bound}} \underbrace{\sum_{i \in R} (H(\mathcal{S}_{LR}, i) + E(a_i))}_{H_{\mathcal{S}R}^{\text{bound}} \text{ bound receptor}} \Big] - \Big[w_{\mathcal{S}L}^{\text{unbnd}} \underbrace{\sum_{i \in L} (H(\mathcal{S}_{L}, i) + E(a_i))}_{H_{\mathcal{S}L}^{\text{unbnd}} \text{ unbound ligand}} + w_{\mathcal{S}R}^{\text{unbnd}} \underbrace{\sum_{i \in R} (H(\mathcal{S}_{R}, i) + E(a_i))}_{H_{\mathcal{S}R}^{\text{unbnd}} \text{ unbound receptor}} \Big], (\mathcal{S} = \mathcal{W}, \mathcal{M}). \quad (10)$$

where $w_{\mathcal{S}J} > 0$ ($\mathcal{S} = \mathcal{W}, \mathcal{M}; J = L, R$) is the coefficient that controls the contribution of entropy to the binding free energy, and $E(\cdot)$ is the entropy bias for 20 different amino acid types. According to the thermodynamic definition, the core component in binding $\Delta\Delta G$ is the difference in entropy loss between the wild-type and mutated structure. Therefore, we estimate $\Delta\Delta G$ by:

$$\Delta\Delta G_{\text{pred}} = \Delta H_{\mathcal{M}} - \Delta H_{\mathcal{W}} + b \quad (11)$$

$$= (w_{\mathcal{M}L}^{\text{bound}} H_{\mathcal{M}L}^{\text{bound}} + w_{\mathcal{M}R}^{\text{bound}} H_{\mathcal{M}R}^{\text{bound}}) - (w_{\mathcal{M}L}^{\text{unbnd}} H_{\mathcal{M}L}^{\text{unbnd}} + w_{\mathcal{M}R}^{\text{unbnd}} H_{\mathcal{M}R}^{\text{unbnd}}) - (w_{\mathcal{W}L}^{\text{bound}} H_{\mathcal{W}L}^{\text{bound}} + w_{\mathcal{W}R}^{\text{bound}} H_{\mathcal{W}R}^{\text{bound}}) + (w_{\mathcal{W}L}^{\text{unbnd}} H_{\mathcal{W}L}^{\text{unbnd}} + w_{\mathcal{W}R}^{\text{unbnd}} H_{\mathcal{W}R}^{\text{unbnd}}) + b. \quad (12)$$

Table 1: Evaluation of $\Delta\Delta G$ predictors on the SKEMPI2 dataset. RDE-Network outperforms baseline methods. Most notably, RDE-Network significantly improves per-structure correlations, which are more relevant to practical applications.

Method	Overall Pearson	Overall Spearman	RMSE (kcal/mol)	MAE (kcal/mol)	AUROC	Per-Struct. Pearson	Per-Struct. Spearman
Rosetta	0.3113	0.3468	1.6173	1.1311	0.6562	0.3284	0.2988
FoldX	0.3120	0.4071	1.9080	1.3089	0.6582	0.3789	0.3693
ESM-1v	0.1921	0.1572	1.9609	1.3683	0.5414	0.0073	-0.0118
DDGPred	**0.6580**	0.4687	1.4998	1.0821	0.6992	0.3750	0.3407
End-to-End	0.6373	0.4882	1.6198	1.1761	0.7172	0.3873	0.3587
B-factor	0.2390	0.2625	2.0411	1.4402	0.6044	0.2042	0.1686
MIF-Δlogit	0.2918	0.2192	1.9092	1.3301	0.5749	0.1585	0.1166
MIF-Net.	0.6523	0.5134	1.5932	1.1469	0.7329	0.3965	0.3509
RDE-Linear	0.4185	0.3514	1.7832	1.2159	0.6059	0.2903	0.2632
RDE-Net.	0.6540	**0.5482**	**1.5901**	**1.1174**	**0.7342**	**0.4582**	**0.4210**

Note that when there are no mutations in the receptor group ($\mathcal{M}_R = \mathcal{W}_R$), $H^{\text{unbnd}}_{\mathcal{M}R}$ and $H^{\text{unbnd}}_{\mathcal{W}R}$ cancel out each other and should have no contribution to $\Delta\Delta G$. On some occasions where mutations are on both groups, we consider the set of only mutated amino acids as the ligand group and all the remaining as the receptor group. To calibrate the parameters in Eq.11, we use the block coordinate descent method, which alternates between optimizing the coefficients ($w_{\mathcal{S}J}$, b) and the entropy biases (E) using the mean squared error (MSE) loss.

Neural Network Predictor The hidden representation $\boldsymbol{h}_i$ of each amino acid for parameterizing the normalizing flows contains sufficient information about the rotamer probability density. To extract binding information from the representations in a more flexible way, we also try neural networks. Specifically, we use MLPs to transform the representation $\boldsymbol{h}_i$ and apply max-pooling to obtain a global representation of the structure. Then, we subtract the representation of the wild-type structure from the representation of the mutant and feed it to another MLP to predict $\Delta\Delta G$. To enforce anti-symmetry, we swap the wild-type and the mutant to predict $-\Delta\Delta G$ and take $(\Delta\Delta G - (-\Delta\Delta G))/2$ as the final prediction. The network is trained with the MSE loss. During training, *gradients are not back-propagated through* $\boldsymbol{h}_i$ so the rotamer density estimator is frozen, as our goal is to exploit the unsupervised representations learned by the RDE.

3.5 Model Training

The datasets for training the rotamer density estimator are derived from PDB-REDO (Joosten et al., 2014), a database containing refined X-ray structures in PDB. Structures with resolution worse than 3.5Å are removed. All the protein chains are clustered by 50% sequence identity. This process leads to 38,413 chain clusters, and they are randomly divided into the training set, the validation set, and the test set by 95%/0.5%/4.5%. At training time, the data-loader randomly selects a cluster and then randomly chooses a chain from the cluster. During training, the structure is cropped into a patch containing 128 amino acids. The patch is constructed by first choosing a seed amino acids, and then finding its 127 nearest neighbors according to C-beta distances. To emulate mutations, the rotamers of 10% of amino acids in the patch are masked. Noise is added to the rotamers of amino acids whose C-beta distance to the closest masked amino acid is less than 8Å.

The SKEMPI2 database (Jankauskaitė et al., 2019) is used to train the models for $\Delta\Delta G$ prediction described in Section 3.4. The dataset is split into 3 folds by structure, each containing unique protein complexes not appearing in other folds. Two folds are used for training and validation, and the rest one is used for testing. This yields 3 different sets of parameters and ensures that every data points in SKEMPI2 are tested on once.

4 Results

4.1 Prediction of the Effect of Mutations on Binding

Baselines We compare our two $\Delta\Delta G$ predictors **RDE-Linear** and **RDE-Network** with various categories of baseline methods. The first category is traditional empirical energy functions, including **Rosetta** Cartesian ddG (Park et al., 2016) and **FoldX** (Delgado et al., 2019). The second class is sequence-based methods represented by **ESM-1v** (Meier et al., 2021). We will show that sequence-based or evolution-based methods are incapable of predicting the effect of mutations for

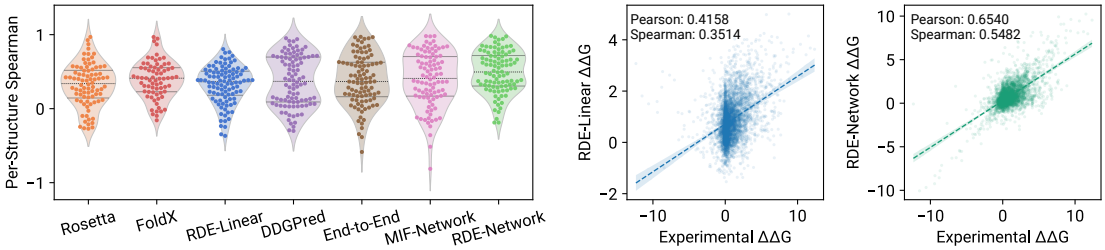

Figure 3: **Left**: The distribution of per-structure Spearman correlation coefficients. **Middle**: Correlation between experimental $\Delta\Delta G$s and $\Delta\Delta G$s predicted by RDE-Linear. **Right**: Correlation between experimental $\Delta\Delta G$s and $\Delta\Delta G$s predicted by RDE-Network.

protein-protein interactions. The third category is end-to-end learning models, including **DDGPred** (Shan et al., 2022) and a model that shares the same encoder architecture with RDE but uses an MLP to directly predict $\Delta\Delta G$ (**End-to-End**). The fourth class is unsupervised representation learning-based methods represented by Masked Inverse Folding (MIF) (Yang et al., 2022). Similar to our RDE-Network, this class of methods pre-train a network on other tasks and use the pretrained representations to predict $\Delta\Delta G$s. The baseline MIF network also uses the same encoder architecture as RDE for fair comparison. There are two variants for $\Delta\Delta G$ prediction: **MIF-Δlogit** uses the difference in log-probability of amino acid types to predict $\Delta\Delta G$, and **MIF-Network** predicts $\Delta\Delta G$ from the learned representations using the network identical to the prediction network of RDE-Network. Finally, as our method is based on conformational flexibility, we train a network to predict the **B-factor** of amino acids and use predicted B-factors in place of entropy to predict $\Delta\Delta G$.

Metrics Five metrics are used to evaluate the overall accuracy of $\Delta\Delta G$ prediction: **Pearson** correlation coefficient, **Spearman**'s rank correlation coefficient, **RMSE** (root mean squared error), **MAE** (mean absolute error), and **AUROC** (area under the receiver operating characteristic). To calculate AUROC, we classify mutations according to the sign of $\Delta\Delta G$. Mutations with $\Delta\Delta G \geq 0$ are labeled as destabilizing and those with $\Delta\Delta G < 0$ are labeled as stabilizing. In practical applications, we often care about the correlation for a specific protein complex. Therefore, we group mutations by structure, discard groups with less than 10 mutations, and calculate correlations for each structure separately. This leads to two other metrics: average **per-structure Pearson** correlation coefficient and average **per-structure Spearman** correlation coefficient.

Results According to Table 1, our RDE-Network outperforms all the baselines. Notably, RDE-Network improves per-structure correlations by a large margin, which implies that it is significantly more reliable for practical applications. The advantage of RDE-Network over MIF-Network shows that representations obtained by fitting rotamer densities are more effective than those from masked inverse folding, because protein binding is driven by atomic interactions which RDE captures well by modeling the conformation of sidechain atoms.

RDE-Linear achieves performance comparable to Rosetta and outperforms some unsupervised representation learning baselines. Though it does not outperform most of the baselines over the whole SKEMPI2 dataset, we find that its performance is much better when considering only single-point mutations (Table 5 in the appendix). The reason might be that simple linear models cannot capture well the non-linear relationship dominating multi-point mutations. Anyway, RDE-Linear shows that using simple statistics of the estimated rotamer density alone can predict $\Delta\Delta G$, lying the foundation for the more accurate RDE-Network.

Sequence-based models fail to predict $\Delta\Delta G$ for protein-protein binding in accordance to the discussion in Section 2.2. Figure 3 plots the distribution of per-complex correlation coefficients. More results are put to Section B of the appendix.

4.2 Optimization of Human Antibodies Against SARS-CoV-2

Shan et al. (2022) identifies 5 single-point mutations on a human antibody against SARS-CoV-2 that enhance neutralization (effectiveness). There are 494 possible single-point mutations on the heavy chain CDR region of the antibody in total. We use the most competitive methods benchmarked in Section 4.1 to predict $\Delta\Delta G$s for all the mutations and rank them in ascending order (lowest $\Delta\Delta G$ in the top). A predictor is considered more effective if it ranks more favorable mutations in the top place. Table 2 shows that RDE-Network and DDGPred successfully identify three mutations (Ranking $\leq 10\%$), and RDE-Network ranks them higher.

Table 2: Rankings of the five favorable mutations on the human antibody against SARS-CoV-2 by various competitive methods. RDE-Network ranks 3 of the 5 mutations in the top place (<10%).

Method	TH31W	AH53F	NH57L	RH103M	LH104F
Rosetta	10.73%	76.72%	93.93%	11.34%	27.94%
FoldX	13.56%	6.88%	5.67%	16.60%	66.19%
DDGPred	68.22%	2.63%	12.35%	8.30%	8.50%
End-to-End	29.96%	2.02%	14.17%	52.43%	17.21%
MIF-Net.	24.49%	4.05%	6.48%	80.36%	36.23%
RDE-Net.	**1.01%**	**2.23%**	25.10%	64.68%	**5.47%**

Table 3: Linear regression shows that the entropies estimated by the RDE correlate significantly to $\Delta\Delta G$.

Var.	Sign	Coef.	P-value	Signif.
$H^{\text{bound}}_{\mathcal{ML}}$	+	0.5122	<0.001	***
$H^{\text{bound}}_{\mathcal{MR}}$	+	0.1808	0.005	**
$H^{\text{unbnd}}_{\mathcal{ML}}$	-	0.2849	<0.001	***
$H^{\text{bound}}_{\mathcal{WL}}$	-	0.4939	<0.001	***
$H^{\text{bound}}_{\mathcal{WR}}$	-	0.2515	<0.001	***
$H^{\text{unbnd}}_{\mathcal{WL}}$	+	0.2471	<0.001	***
H^{unbnd}_{R}	/	0.0325	0.230	-
Bias	/	0.1888	<0.001	***

4.3 Analysis of the Entropy Estimated by the Rotamer Density Estimator

Statistical Significance To show that there is a statistically significant relationship between the entropy estimated by the RDE and the experimental $\Delta\Delta G$ values, we conduct linear regression analysis using the RDE-Linear model defined in Eq.11. The linear model contains 7 coefficients and 1 bias: $w^{\text{bound}}_{\mathcal{WL}}$, $w^{\text{bound}}_{\mathcal{WR}}$, $w^{\text{unbnd}}_{\mathcal{WL}}$, $w^{\text{bound}}_{\mathcal{ML}}$, $w^{\text{bound}}_{\mathcal{MR}}$, $w^{\text{unbnd}}_{\mathcal{ML}}$, $(w^{\text{unbnd}}_{\mathcal{MR}} - w^{\text{unbnd}}_{\mathcal{WR}})$, and b. Here, $w^{\text{unbnd}}_{\mathcal{MR}}$ and $w^{\text{unbnd}}_{\mathcal{WR}}$ are merged because the receptor is not mutated, so the entropy estimated by the model are identical. For short, we denote the merged coefficient by w^{unbnd}_{R}. We perform linear regression on the SKEMPI2 dataset and present the regression coefficients, bias, and P-values in Table 3. According to the statistics, all the entropy terms except the entropy of unbound receptor H^{unbnd}_{R} (coefficient w^{unbnd}_{R}) have significant relationship with experimental $\Delta\Delta G$s. The coefficients of the significant terms are all positive and roughly close to each other. H^{unbnd}_{R} has no contribution because the receptor alone does not involve in the mutation. These agree with the thermodynamic definition of the change in binding free energy: $\Delta\Delta G = \Delta G^{\mathcal{M}} - \Delta G^{\mathcal{W}} = (G^{\mathcal{M}}_{LR} - G^{\mathcal{M}}_{L} - G^{\mathcal{M}}_{R}) - (G^{\mathcal{W}}_{LR} - G^{\mathcal{W}}_{L} - G^{\mathcal{W}}_{R})$, where $G^{\mathcal{M}}_{R}$ and $G^{\mathcal{W}}_{R}$ cancel each other as the receptor is unmutated. Therefore, it is evidenced that our model captures well the thermodynamics underlying protein-protein interactions.

Correlation Between Estimated Entropies and B-factors B-factor is an experimental measurement of conformational flexibility. We collect the average b-factor of sidechain atoms of amino acids in the test split of the PDB-REDO dataset. Simultaneously, we use the RDE to estimate the conformational entropy for each amino acid in the testset. The average Pearson correlation coefficient between these two quantities is 0.4637 and the average Spearman coefficient is 0.4282. Detailed results are presented in Table 7 in the appendix. In summary, this supports that the entropy estimated by the RDE is correlated to experimentally determined conformational flexibility.

4.4 Prediction of Sidechain Conformations

The rotamer density estimator (RDE) is a generative model for sidechain structures. It can be used to predict sidechain conformations by sampling from the estimated distribution. We use the RDE to sample sidechain torsional angles (rotamers) for structures with sidechains removed in our test split of PDB-REDO. For each amino acid, 10 rotamers are sampled independently and the one with highest probability is selected as the final prediction. We compare RDE with two baseline methods Rosetta (fixbb) (Alford et al., 2017) and SCWRL4 (Krivov et al., 2009). Table 4 shows that RDE outperforms the baselines on all the four torsional angles in terms of angular errors. Detailed per-amino-acid accuracy is presented in Table 8 in the appendix.

Table 4: Mean absolute error of the predicted sidechain torsional angles.

	χ_1	χ_2	χ_3	χ_4
SCWRL4	24.33°	32.84°	47.42°	56.15°
Rosetta	23.98°	32.14°	48.49°	58.78°
RDE	**16.02°**	**28.75°**	**45.38°**	**53.26°**

5 Conclusions and Limitations

In this work, we propose the rotamer density estimator (RDE) that estimates the distribution of rotamers. We find the entropy of the estimated distributions and the unsupervised representations produced by the RDE enable accurate prediction of binding $\Delta\Delta G$. One of the major limitations of this work is that it cannot directly model backbone flexibility. Neither do all the considered baselines. Therefore, an important non-trivial future direction is extending the proposed model to backbone-flexible cases.

REFERENCES

Bruce Alberts and Ryn Miake-Lye. Unscrambling the puzzle of biological machines: the importance of the details. *Cell*, 68(3):415–420, 1992.

Rebecca F Alford, Andrew Leaver-Fay, Jeliazko R Jeliazkov, Matthew J O'Meara, Frank P DiMaio, Hahnbeom Park, Maxim V Shapovalov, P Douglas Renfrew, Vikram K Mulligan, Kalli 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.

Rodrigo Barderas, Johan Desmet, Peter Timmerman, Rob Meloen, and J Ignacio Casal. Affinity maturation of antibodies assisted by in silico modeling. *Proceedings of the National Academy of Sciences*, 105(26):9029–9034, 2008.

G Patrick Brady and Kim A Sharp. Entropy in protein folding and in protein—protein interactions. *Current opinion in structural biology*, 7(2):215–221, 1997.

Christian Cole and Jim Warwicker. Side-chain conformational entropy at protein–protein interfaces. *Protein Science*, 11(12):2860–2870, 2002.

Javier Delgado, Leandro G Radusky, Damiano Cianferoni, and Luis Serrano. Foldx 5.0: working with rna, small molecules and a new graphical interface. *Bioinformatics*, 35(20):4168–4169, 2019.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Francis Gaudreault, Matthieu Chartier, and Rafael Najmanovich. Side-chain rotamer changes upon ligand binding: common, crucial, correlate with entropy and rearrange hydrogen bonding. *Bioinformatics*, 28(18):i423–i430, 2012.

Cunliang Geng, Anna Vangone, Gert E Folkers, Li C Xue, and Alexandre MJJ Bonvin. isee: Interface structure, evolution, and energy-based machine learning predictor of binding affinity changes upon mutations. *Proteins: Structure, Function, and Bioinformatics*, 87(2):110–119, 2019a.

Cunliang Geng, Li C Xue, Jorge Roel-Touris, and Alexandre MJJ Bonvin. Finding the $\delta\delta$g spot: Are predictors of binding affinity changes upon mutations in protein–protein interactions ready for it? *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 9(5):e1410, 2019b.

Hermann Gram, Lori-Anne Marconi, CF Barbas 3rd, Thomas A Collet, Richard A Lerner, and Angray S Kang. In vitro selection and affinity maturation of antibodies from a naive combinatorial immunoglobulin library. *Proceedings of the National Academy of Sciences*, 89(8):3576–3580, 1992.

Thomas A Hopf, John B Ingraham, Frank J Poelwijk, Charlotta PI Schärfe, Michael Springer, Chris Sander, and Debora S Marks. Mutation effects predicted from sequence co-variation. *Nature biotechnology*, 35(2):128–135, 2017.

Justina Jankauskaitė, Brian Jiménez-García, Justas Dapkūnas, Juan Fernández-Recio, and Iain H Moal. Skempi 2.0: an updated benchmark of changes in protein–protein binding energy, kinetics and thermodynamics upon mutation. *Bioinformatics*, 35(3):462–469, 2019.

Robbie P Joosten, Fei Long, Garib N Murshudov, and Anastassis Perrakis. The pdb_redo server for macromolecular structure model optimization. *IUCrJ*, 1(4):213–220, 2014.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Panagiotis L Kastritis and Alexandre MJJ Bonvin. On the binding affinity of macromolecular interactions: daring to ask why proteins interact. *Journal of The Royal Society Interface*, 10(79): 20120835, 2013.

Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack Jr. Improved prediction of protein side-chain conformations with scwrl4. *Proteins: Structure, Function, and Bioinformatics*, 77(4): 778–795, 2009.

Xianggen Liu, Yunan Luo, Pengyong Li, Sen Song, and Jian Peng. Deep geometric representations for modeling effects of mutations on protein-protein binding affinity. *PLoS computational biology*, 17(8):e1009284, 2021.

Lenette L Lu, Todd J Suscovich, Sarah M Fortune, and Galit Alter. Beyond binding: antibody effector functions in infectious diseases. *Nature Reviews Immunology*, 18(1):46–61, 2018.

Yunan Luo, Guangde Jiang, Tianhao Yu, Yang Liu, Lam Vo, Hantian Ding, Yufeng Su, Wesley Wei Qian, Huimin Zhao, and Jian Peng. Ecnet is an evolutionary context-integrated deep learning framework for protein engineering. *Nature communications*, 12(1):1–14, 2021.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 34:29287–29303, 2021.

Rafael Najmanovich, Josef Kuttner, Vladimir Sobolev, and Marvin Edelman. Side-chain flexibility in proteins upon ligand binding. *Proteins: Structure, Function, and Bioinformatics*, 39(3):261–268, 2000.

Hahnbeom Park, Philip Bradley, Per Greisen Jr, Yuan Liu, Vikram Khipple Mulligan, David E Kim, David Baker, and Frank DiMaio. Simultaneous optimization of biomolecular energy functions on features from small molecules and macromolecules. *Journal of chemical theory and computation*, 12(12):6201–6212, 2016.

Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In *International Conference on Machine Learning*, pp. 8844–8856. PMLR, 2021.

Danilo Jimenez Rezende, George Papamakarios, Sébastien Racaniere, Michael Albergo, Gurtej Kanwar, Phiala Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. In *International Conference on Machine Learning*, pp. 8083–8092. PMLR, 2020.

Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, 2018.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, 2019. doi: 10.1101/622803. URL https://www.biorxiv.org/content/10.1101/622803v4.

Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The foldx web server: an online force field. *Nucleic acids research*, 33(suppl_2):W382–W388, 2005.

Sisi Shan, Shitong Luo, Ziqing Yang, Junxian Hong, Yufeng Su, Fan Ding, Lili Fu, Chenyu Li, Peng Chen, Jianzhu Ma, et al. Deep learning guided optimization of human antibody against sars-cov-2 variants with broad neutralization. *Proceedings of the National Academy of Sciences*, 119(11): e2122954119, 2022.

Raghav Shroff, Austin W Cole, Daniel J Diaz, Barrett R Morrow, Isaac Donnell, Ankur Annapareddy, Jimmy Gollihar, Andrew D Ellington, and Ross Thyer. Discovery of novel gain-of-function mutations guided by structure-based deep learning. *ACS synthetic biology*, 9(11):2927–2935, 2020.

Thomas Steinbrecher, Robert Abel, Anthony Clark, and Richard Friesner. Free energy perturbation calculations of the thermodynamics of protein side-chain mutations. *Journal of Molecular Biology*, 429(7):923–929, 2017.

Jérôme Tubiana, Dina Schneidman-Duhovny, and Haim J Wolfson. Scannet: An interpretable geometric deep learning model for structure-based protein binding site prediction. *Nature Methods*, pp. 1–10, 2022.

Jingxue Wang, Huali Cao, John ZH Zhang, and Yifei Qi. Computational protein design with deep learning neural networks. *Scientific reports*, 8(1):1–9, 2018.

Kevin K Yang, Niccolò Zanichelli, and Hugh Yeh. Masked inverse folding with sequence transfer for protein representation learning. *bioRxiv*, 2022.

Ning Zhang, Yuting Chen, Haoyu Lu, Feiyang Zhao, Roberto Vera Alvarez, Alexander Goncearenco, Anna R Panchenko, and Minghui Li. Mutabind2: predicting the impacts of single and multiple mutations on protein-protein interactions. *Iscience*, 23(3):100939, 2020.

Zuobai Zhang, Minghao Xu, Arian Jamasb, Vijil Chenthamarakshan, Aurelie Lozano, Payel Das, and Jian Tang. Protein representation learning by geometric structure pretraining. *arXiv preprint arXiv:2203.06125*, 2022.

A IMPLEMENTATION DETAILS

A.1 NETWORK ARCHITECTURE

The encoder of the rotamer density estimator is a stack of attention layers invariant to the rotation and translation (Jumper et al., 2021; Tubiana et al., 2022) of the input protein structure. Let $\boldsymbol{h}_i^\ell$ denote the embedding of the i-th amino acid output by the previous attention layer. The logit of attention weights is between residue i and residue j is defined as:

$$a_{ij} = \langle Q(\boldsymbol{h}_i^\ell), K(\boldsymbol{h}_i^\ell) \rangle + G(\boldsymbol{z}_{ij}) + H\left(\{\boldsymbol{O}_i^\intercal(\boldsymbol{x}_j^k - \boldsymbol{x}_i^k)\}_k\right), \tag{13}$$

where Q, K, G, and H are MLP networks that transform the embeddings into queries, keys, pairwise bias, and distance bias, respectively. The attention weight is computed by taking the softmax on the j dimension: $w_{ij} = \mathrm{softmax}_j(a_{ij})$. In practice, we use multiple attention heads. Each attention head has different attention weights. The vector to update the representation of residue i from residue j is computed by:

$$\boldsymbol{v}_{ij} = V\left(\boldsymbol{h}_j^\ell, \boldsymbol{z}_{ij}, \{\boldsymbol{O}_i^\intercal(\boldsymbol{x}_j^k - \boldsymbol{x}_i^k)\}_k\right). \tag{14}$$

Finally, the sum of $\{\boldsymbol{v}_{ij}\}_j$ weighted by $\{w_{ij}\}_j$ is used to update the representation of residue j using residual connection and layer normalization, similar to the standard transformer architecture.

A.2 TRAINING

The rotamer density estimator is trained using the Adam optimizer for 200K iterations. The learning rate is 0.0001. The learning rate decays by 0.8 if the validation loss does not decrease in the last 5 validation steps (the model is validated every 1000 iteration), until the learning rate reaches 0.000001. The batch size is 64. It takes 8h56m in total on a single A100 GPU.

To emulate mutations, the rotamers of 10% of amino acids are masked. Noise is added to the rotamers of amino acids whose C-beta distance to the closest masked amino acid is less than 8.0Å. The noise added to χ angles consists of two components. The first component is Gaussian noise centered at 0 wrapped into $[-\pi, \pi]$. It standard deviation is dependent on the C-beta distance: $\sigma_{ij} = -\frac{1}{16}\beta_{ij} + 1$, where i is the index of the nearest masked amino of the j-th amino acid that the noise is added to, and β_{ij} is the C-beta distance between them. The second component is random flipping (adding π to the angle). Every χ angle in the 8Å neighborhood has 25% chance of being flipped. Our noise model is totally empirical. There are other ways to perturb rotamers, for example, using using rotamer libraries. We leave the problem of finding an optimal noise model that emulate mutations for future work.

A.3 BASELINES

Baselines that require training and calibration using the SKEMPI2 dataset (DDGPred, End-to-End, B-factor, MIF-Δlogit, MIF-Network) are trained independently using the 3 different splits of the dataset as described in Section 3.5. This is to ensure that every data point in the SKEMPI2 dataset is tested on once. Below are descriptions of the implementation of baseline methods.

Rosetta The version we used is 2021.16, and the scoring function is `ref2015_cart`. Every protein structures in the SKEMPI2 dataset are first pre-processed using the `relax` application. The mutant structure is built by `cartesian_ddg`. The binding free energies of both wild-type and mutant structures are predicted by `interface_energy` (dG_separated/dSASAx100). Finally, the binding $\Delta\Delta G$ is calculated by substracting the binding energy of the wild-type structure from the binding energy of the mutant.

FoldX Structures are first relaxed by the `RepairPDB` command. Mutant structures are built with the `BuildModel` command based on the repaired structure. The binding free energy is estimated using the complex analyzer and $\Delta\Delta G$ is calculated with the binding energy of the wild-type and the mutant.

ESM-1v Protein language models can only predict the effect of mutations for single protein sequences. Therefore, we ignore cases where mutations are on multiple sequences. We extract the sequence of mutated protein chain from the `SEQRES` entry of the PDB file. We use

`masked-marginals` mode to score both wild-type and mutant sequences and use their difference as the estimation of $\Delta\Delta G$.

DDGPred We use the implementation accompanying the paper by Shan et al. (2022). Since this model requires predicted sidechain structures of the mutant, we use mutant structures packed during our evaluation of Rosetta to train the model and run prediction.

End-to-End The end-to-end model shares the same encoder architecture with the rotamer density estimator. The difference is that in the RDE normalizing flows follow the encoder to model rotamer distributions, but in the end-to-end model, the embeddings are directly fed to an MLP to predict $\Delta\Delta G$.

B-factor This model predicts per-atom b-factors for proteins. It has the same encoder architecture as the RDE. Following the encoder is an MLP that predicts a vector for each amino acid where each dimension is the predicted b-factor of different atoms in the amino acid. The amino acid-level b-factor is calculated by averaging the atom-level b-factors. The predicted b-factors are used as a measurement of conformational flexibility. They are used to further predict $\Delta\Delta G$ using the linear model same as RDE-Linear defined in Eq.11.

MIF Architecture The masked inverse folding (MIF) network uses the same encoder architecture as RDE. Following the encoder is a per-amino-acid 20-category classifier that predicts the type of masked amino acids. We use the same PDB-REDO train-test split to train the model. At training time, we randomly crop a patch consisting of 128 residues, and randomly mask 10% amino acids. The model learns to recover the type of the masked amino acids with the standard cross entropy loss.

MIF-Δlogit To score mutations, we first mask the type of mutated amino acids. Then, we use the log-probability of the amino acid type as the score. Analogously, we have the score of the wild-type bound ligand, wild-type bound receptor, wild-type unbound ligand, unbound receptor, mutated bound ligand, mutated bound receptor, and mutated unbound ligand. Therefore, we use the linear model identical to RDE-Linear (Eq.11) to predict $\Delta\Delta G$ from the scores.

MIF-Network This is similar to RDE-Network. The difference is that we use the pre-trained encoder of MIF rather than the encoder of RDE. We also freeze the MIF encoder as we aim to utilize the unsupervised representations.

A.4 Source Code

The code and data will be made available once this paper is made public.

B ADDITIONAL RESULTS

Table 5: Evaluation of $\Delta\Delta G$ predictors on the single-mutation subset of SKEMPI2.

Method	Overall Pearson	Overall Spearman	RMSE (kcal/mol)	MAE (kcal/mol)	AUROC	Per-Struct. Pearson	Per-Struct. Spearman
Rosetta	0.3113	0.3468	1.6173	1.1311	0.6562	0.3284	0.2988
FoldX	0.3560	0.3511	1.5576	1.0713	0.6478	0.3908	0.3640
ESM-1v	0.1914	0.1572	1.7226	1.1917	0.5492	0.0422	0.0273
DDGPred	0.6515	0.4390	1.3285	0.9618	0.6858	0.3711	0.3427
End-to-End	0.6605	0.4594	1.3148	0.9569	0.7019	0.3818	0.3426
B-factor	0.1748	0.2054	1.7242	1.1889	0.6100	0.1884	0.1661
MIF-Δlogit	0.2548	0.1927	1.6928	1.1671	0.5630	0.1616	0.1231
MIF-Net.	**0.6667**	0.4802	**1.3052**	0.9411	0.7175	0.3952	0.3479
RDE-Linear	0.4003	0.3754	1.5845	1.0688	0.6284	0.3588	0.3271
RDE-Net.	0.6627	**0.5121**	1.3114	**0.9159**	**0.7218**	**0.4952**	**0.4512**

Table 6: Evaluation of $\Delta\Delta G$ predictors on the Multi-mutation subset of SKEMPI2.

Method	Overall Pearson	Overall Spearman	RMSE (kcal/mol)	MAE (kcal/mol)	AUROC	Per-Struct. Pearson	Per-Struct. Spearman
Rosetta	0.1991	0.2303	2.6581	2.0246	0.6207	0.1915	0.0836
FoldX	0.2347	0.4137	2.5290	1.8639	0.6828	0.2801	0.2771
ESM-1v	0.1923	0.1749	2.7586	2.1193	0.5415	-0.0599	-0.1284
DDGPred	0.5938	0.5150	2.1813	1.6699	0.7590	0.3912	0.3896
End-to-End	0.5858	0.4942	2.1971	1.7087	0.7532	0.4178	0.4034
B-factor	0.2009	0.2445	2.6557	2.0186	0.5876	0.2078	0.1850
MIF-Δlogit	0.3358	0.2886	2.5361	1.8967	0.6066	0.1053	0.0783
MIF-Net.	0.6139	0.5370	2.1399	1.6422	**0.7735**	0.3968	0.3789
RDE-Linear	0.4636	0.4290	2.4383	1.8134	0.6560	0.1764	0.1910
RDE-Net.	**0.6240**	**0.5779**	**2.1185**	**1.6003**	0.7677	**0.4263**	**0.4206**

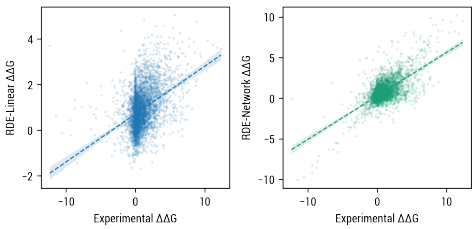

Figure 4: Correlation between experimental $\Delta\Delta G$s and $\Delta\Delta G$s predicted by RDE-Linear and RDE-Network on SKEMPI2 single-mutation subset.

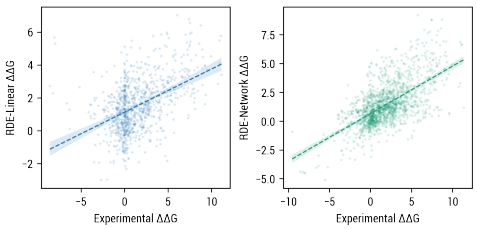

Figure 5: Correlation between experimental $\Delta\Delta G$s and $\Delta\Delta G$s predicted by RDE-Linear and RDE-Network on SKEMPI2 multi-mutation subset.

Table 7: Correlation between average sidechain B-factors and entropies estimated by RDE. Scatter plots of each amino acid type are in Figure 6.

Type	Pearson	Spearman
ARG	0.4683	0.4904
ASN	0.4329	0.3826
ASP	0.4680	0.4408
CYS	0.4099	0.4099
GLN	0.4691	0.4463
GLU	0.4683	0.4861
HIS	0.3743	0.3261
ILE	0.5288	0.4677
LEU	0.5281	0.4638
LYS	0.5479	0.5777
MET	0.5026	0.4758
PHE	0.3615	0.2667
SER	0.5368	0.5247
THR	0.5945	0.5263
TRP	0.3263	0.2910
TYR	0.3588	0.2493
VAL	0.5068	0.4536
Average	0.4637	0.4282

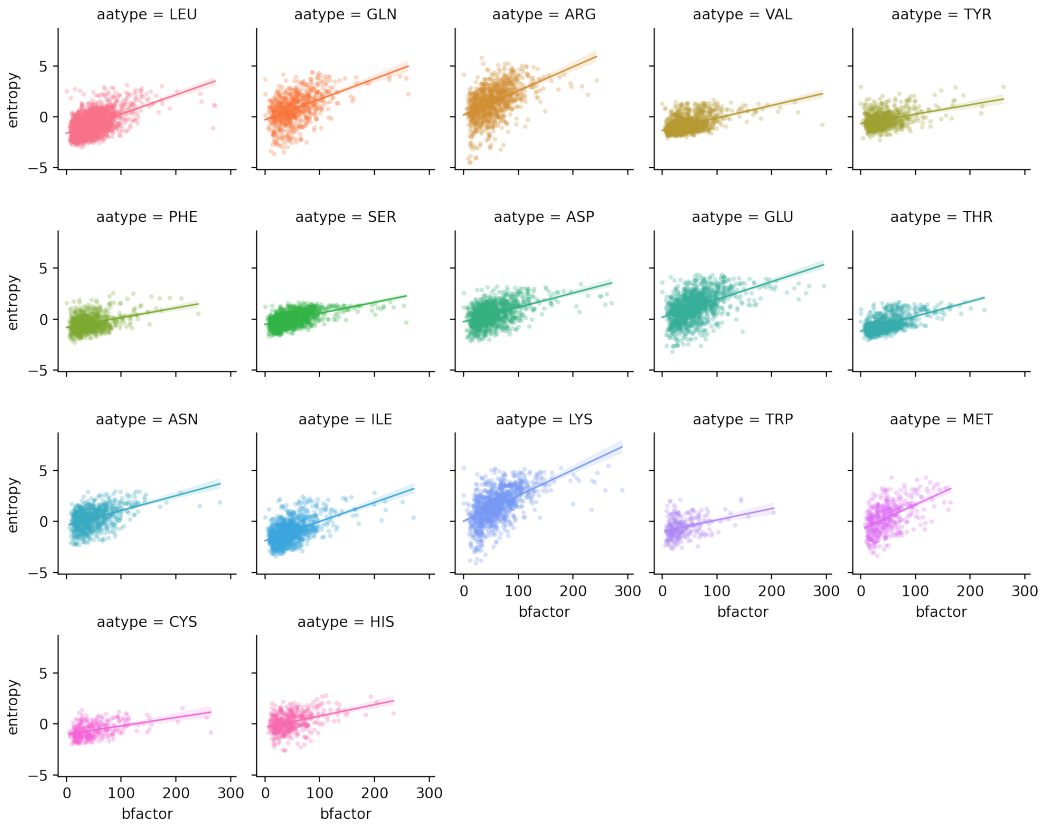

Figure 6: Scatter plots of average sidechain B-factors and estimates entropies.

Table 8: Mean absolute error of the predicted sidechain torsional angles.

Type	χ	SCWRL4	Rosetta	RDE
ARG	1	29.30	30.50	**20.90**
	2	**28.68**	36.12	28.72
	3	57.89	60.73	**56.83**
	4	60.35	63.62	**56.76**
ASN	1	21.70	19.39	**17.26**
	2	44.00	43.41	**41.36**
ASP	1	25.75	22.62	**17.54**
	2	23.90	21.26	**21.15**
CYS	1	24.83	25.90	**12.74**
GLN	1	33.16	31.53	**22.78**
	2	46.33	**33.96**	35.16
	3	53.72	56.52	**52.80**
GLU	1	35.45	34.69	**26.14**
	2	38.23	38.43	**34.60**
	3	31.50	30.85	**29.35**
HIS	1	23.15	19.12	**17.88**
	2	70.62	**61.97**	69.69
ILE	1	13.92	14.65	**8.86**
	2	26.43	27.54	**21.54**
LEU	1	13.97	14.15	**9.41**
	2	23.76	27.61	**19.32**
LYS	1	31.32	33.88	**21.64**
	2	**30.95**	33.15	32.90
	3	38.90	42.07	**37.70**
	4	51.94	53.94	**49.76**
MET	1	26.36	26.07	**16.87**
	2	38.52	36.09	**27.09**
	3	55.11	58.77	**50.21**
PHE	1	12.30	13.08	**9.22**
	2	12.40	12.35	**9.45**
SER	1	47.19	46.83	**25.66**
THR	1	28.05	22.67	**16.85**
TRP	1	14.52	18.64	**8.50**
	2	31.74	31.44	**24.42**
TYR	1	11.39	14.56	**9.25**
	2	11.37	14.45	**8.35**
VAL	1	21.31	19.41	**10.82**