Supporting Information for

Learning the shape of protein micro-environments with a holographic convolutional neural network

M. N. Pun, A. Ivanov, Q. Bellamy, Z. Montague, C. LaMont, P. Bradley, J. Otwinowski, A. Nourmohammad

Armita Nourmohammad.
E-mail: armita@uw.com

This PDF file includes:

Supporting Information Text

1. Data pre-processing and code availability

Code and data availability. All codes and references to data are available on github through:
https://github.com/StatPhysBio/protein_holography.

Processing of the protein structure data. We minimally process the protein structures from the Protein Data Bank (PDB) (1). We use PyRosetta (2) to infer coordinates of hydrogen atoms in the protein structure. We also use PyRosetta to assign solvent accessible surface area (SASA) and partial charge to all atoms.

Constructing the training, the validation, and the test sets for protein neighborhood classifications. We classified amino acid neighborhoods extracted from the tertiary protein structures from the PDB. We used ProteinNet's splittings for training and validation sets of the CASP12 competition to avoid redundancy due to similarities in homologous protein domains (3). Since PDB identifiers in ProteinNet were only provided for the training and the validation sets, we used ProteinNet's training set as the base of both our training and validation and ProteinNet's validation set as our testing set. Specifically, to define our training and validation sets, we make a [80%, 20%] split in the ProteinNet's training data of proteins with X-ray crystallography structures of 2.5Å resolution or better. Furthermore, in anticipation of validating our model on experimental stabilities of T4 Lysozyme and SARS-CoV2 receptor-binding domain (RBD), we removed all structures that shared the same UniProtKB accession as the domains from each of these proteins. We used all of ProteinNet's validation structures as our testing set.

ProteinNet has divided the testing set into subsets depending on sequence similarity to the entire set of protein sequences that were used for the different thinnings of training sets. We evaluated the model's accuracy on each of these subsets to examine how sequence similarity affects the model's performance. For slicing with sequence identity $> 10\%$, H-CNN performance is insensitive to exact splitting of sequences in training and test sets (Fig. S5C), irrespective of the amplitude of the noise injected in the coordinated of the atoms in the training and test sets. We used ProteinNet's 30% splitting to have less chance of evolutionary redundancy in our training set. This splitting resulted in 10,957 training structures, 2,730 validation structures, and 212 testing structures, each of which contained 2,810,503 training neighborhoods, 682,689 validation neighborhoods, and 4,472 testing neighborhoods.

In addition to ProteinNet, we also used ProteinMPNN's CATH-based splitting of the PDB (4). Specifically, ProteinMPNN assigns PDB structures to clusters based on structural similarity. We used a 80/10/10 train/val/test split of these clusters and took one representative structure per cluster at random. This splitting resulted in 14,052 training structures, 1,754 validation structures, and 1,756 testing structures. All residues in each structure were used in each set resulting in 3,331,033 training neighborhoods, 421,578 validation neighborhoods, and 415,360 testing neighborhoods.

Noising of training and testing structures. Noised neighborhoods were generated by adding Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ where the standard deviation was fixed across a given noised ensemble. The channel information including SASA and charge were taken directly from the un-noised structure. We evaluated the accuracy of models trained on un-noised data in predicting amino acid identities in noised test sets. The range of noise levels for these test sets were chosen as $\sigma \in \{0.001, 0.01, 0.03, 0.05, 0.1, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$. We also trained models on noised structures and evaluated their accuracies on noised and un-noised test sets. The range of noise levels for these training sets were chosen as $\sigma \in \{0.02, 0.1, 0.3, 0.5\}$.

Data processing to assess the stability effect of mutations in the T4 Lysozyme protein. Predictions for the stability effect of mutations in the T4 Lysozyme protein were made using the PDB structure 2LZM (5) as the wild-type. The PDB identifiers for structures of all the T4 Lysozyme variants used in our analyses (shown in Figs. 4 and S10) along with the reported $\Delta\Delta G$ and the pH values of the stability measurements are reported in Table S2.

Experimental measurements of $\Delta\Delta G$ values in variants for which we do not have a matching protein structure (shown in Fig. S9) are taken from the dataset FireProtDB (6).

The AlphaFold prediction of the T4 Lysozyme protein structure, used in Fig. 4C and S9, was performed using the AlphaFold GoogleColab notebook (7).

Relaxation of mutant T4 Lysozyme structures for the *in silico* model in Figs. 4B,C and S10, was done using PyRosetta `FastRelax` with the score function `ref2015_cart` (2). Backbone flexible positions were restricted to the mutated residue and the neighboring residues in the sequence. Side-chain flexible positions were restricted to the neighbors with alpha carbons within 10 Å of the mutated residue's alpha carbon.

Data processing to assess the fitness effect of mutations in the RBD of SARS-CoV-2. Deep mutational scanning experiments from ref. (8) were used to obtain the SARS-CoV-2 RBD expression and binding to the ACE2 receptor. Predictions of the RBD-ACE2 binding strength, shown in Fig. 5B, were made using the co-crystallized protein structure with the PDB identifier 6M0J (9). The predictions for the RBD stability, shown in Fig. 5A, were made using the 6M0J structure with the ACE2 chain computationally removed. This computational removal was performed by selecting the RBD structure in PyMol (10) and exporting a PDB file from said selection.

Data processing to assess the effect of shearing in Protein G. To analyze the effect of shearing in protein G, shown in Figs. 3 and S8, we used the native structure of protein G with the PDB identifier 1PGA (11). Sheared structures were produced with PyRosetta by manually editing the backbone angles ϕ and ψ of each residue (Fig. 3A) via `Pose.set_phi` and `Pose.set_psi` (2).

2. Rotationally equivariant structure-to-function map for proteins

The tertiary protein structure is a result of a folding process whereby amino acid side-chains interact with each other and the backbone structure to form the three-dimensional (3D) shape of the protein. The favorability of a certain amino acid occurring at a given site in a protein is determined by numerous factors: the presence of this amino acid during translation, the ability of the polypeptide chain to fold with that amino acid present, the amino acid's interactions with the local structural surroundings of the protein, and the amino acid's interactions with the surrounding environment. Our work focuses on revealing the third factor's contribution. Specifically, we aim to learn how a micro-environment in a protein's tertiary structure (i.e., atoms in a given structural neighborhood) constrain the usage of different amino acids in the neighborhood.

We define a structural neighborhood surrounding a focal amino acid by the coordinates (and identities) of all the N atoms from the surrounding amino acids in the structure, within a given radius of the focal residue. Mathematically, we want to find a map $f: \mathbb{R}^{N\times 3} \to [\mathbb{R}^{+}_{\leq 1}]^{19}$ between the 3D coordinates of the N neighboring atoms and the vector of probabilities $p \in [\mathbb{R}^{+}_{\leq 1}]^{19}$ associated with preferences for different types of amino acids that can be placed at the center of the neighborhood; note that the amino acid probability vector has 19 independent entries due to normalization.

Assuming that the protein structure is a crystal (at least on average), the local interactions between the amino acid at the focal site of interest and the surrounding ones in the protein structure have rotational symmetry, in that they only depend on the pairwise distances of atoms and not on their absolute orientation in space. This symmetry implies that the interactions are unchanged under global rotations since a global rotation preserves pairwise distances and relative orientations. Therefore, any model of amino acid preferences based on local structural information should respect 3D rotational symmetry. Thus, our goal becomes learning a map f, as described above, that is also invariant to global rotations acting on the inputs. A rotation acting on any Cartesian vector can be expressed as an Euler angle matrix R_{euler} (Fig. S1). Since our inputs $\mathbf{x}$ for coordinates of the atoms in a structural neighborhood are Cartesian vectors $\mathbf{x} \in \mathcal{R}^{N\times 3}$, a global rotation R would act on these vectors via a Euler angle matrix, which we represent by $D_{\text{cart.}}(R)$. Therefore, the rotationally invariant map should have the property $f(D_{\text{cart.}}(R)\mathbf{x}) = f(\mathbf{x})$.

In the rest of this section, we describe the mathematical foundation to construct 3D rotationally equivariant models for protein structure data. In the following section (Section A), we specifically focus on our approach in encoding the structural data and training neural networks to learn 3D rotationally equivariant models for protein neighborhoods. The methodology discussed in Sections B, C is self-contained, and if a reader wishes they can choose to skip Section A on the mathematical foundation of our approach.

A. Mathematical foundation for 3D rotationally equivariant transformations. The simplest approach to train a neural network model that is invariant to global rotations of proteins is to use invariant descriptors of the tertiary structure as inputs to the neural network. For example, one can use dot products between all Cartesian vectors describing the coordinates of atoms within a structural neighborhood. However, recent work has shown that neural networks that are allowed to work with full geometric data as opposed to simple invariants have higher expressivity and perform better on learning tasks (12).

Networks that build more complex invariants rely on representation theory and the idea of *equivariance* to build more expressive models. Equivariance is a generalization of invariance. Rather than requiring no change when inputs are transformed, equivariance allows the outputs of a map to change but requires them to change in a well-defined way. Under a transformation of the inputs corresponding to a rotation R, an equivariant map f satisfies the relation

$$f(D_{\text{input}}(R)x) = D_{\text{out}}(R)f(x) \qquad \text{[S1]}$$

where $D_{\text{input}}(R)$ is the operator corresponding to the rotation in the input space, and $D_{\text{out}}(R)$ is the operator corresponding to the rotation in the output space. Note that if $D_{\text{out}}(R)$ is the identity matrix for all rotations, then the map is invariant.

The key to building an equivariant map is to know the operator D_{out} for the transformation of interest. The prevailing method in the field of equivariant machine learning is to modularly build a neural network out of equivariant units where each unit has an input and output space where these operators are known. The input and output space are often the same and tend to be the Fourier space defined by the symmetry that is respected.

Rotations in three dimensions are non-commutative which means that the output of two consecutive rotations can depend on the order by which they act on the input vector. As a result, the Fourier space for rotations is more complicated than that of a commutative transformation like translations. Here, we will first use translations as an illustrative example for equivariant operations in Fourier space and then generalize to rotations.

Translational equivariance. The Fourier transform of a signal on a real line $f(x)$ can be expressed as $\hat{F}(k) = \int_{-\infty}^{\infty} e^{-ikx} f(x)dx$. We define a translated signal $f_a(x) = f(x-a)$ which is the original signal translated by a distance a. The Fourier transform of this translated signal follows

$$\begin{aligned}
\hat{F}_a(k) &= \int_{-\infty}^{\infty} e^{-ikx} f_a(x)dx \\
&= \int_{-\infty}^{\infty} e^{-ikx} f(x-a)dx \\
&= \int_{-\infty}^{\infty} e^{-ik(x'+a)} f(x')dx' \\
&= e^{-ika}\hat{F}(k)
\end{aligned} \qquad \text{[S2]}$$

Thus, the Fourier transform of a translated signal transforms as $\hat{F}_a \xrightarrow{a} e^{-ika}\hat{F}(k)$, implying that the Fourier transform is equivariant under translation with an output operator that is simply a phase shift $D_{\text{out}}(a) = e^{-ika}$. These operators $D_{\text{out}}(a)$ form a representation of the group of 1D translations (13).

In three dimensions, translations act on vectors via $\mathbf{x} \to \mathbf{x} + \mathbf{a}$ and the representation is given by the operators operator $D_{\text{out}}(\mathbf{a}) = e^{-i\mathbf{k}\cdot\mathbf{a}}$.

Rotational equivariance. Similar to translation, we can use Fourier space to define an equivariant transformation for rotations. We consider rotations about a given reference point that defines an origin for a spherical coordinate system in which points in 3D can be expressed by the resulting spherical coordinates (r, θ, ϕ). Since the radius r (i.e., the distance of a point from to the reference) does not change under rotations about the origin, we will ignore the radial component for now and consider a signal over the sphere of radius r, $f(\theta, \phi) : \; S^2(r) \to \mathbb{R}$. The Fourier transform $\hat{F}$ of the signal on the sphere follows,

$$\hat{F}_{\ell m} = \int_0^{2\pi} \int_0^{\pi} f(\theta, \phi) Y_{\ell m}(\theta, \phi) \sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi \tag{S3}$$

where $Y_{\ell m}(\theta, \phi)$ is the spherical harmonic of degree ℓ and order m defined as

$$Y_{\ell m}(\theta, \phi) = \sqrt{\frac{2n+1}{4\pi} \frac{(n-m)!}{(n+m)!}} e^{im\phi} P_\ell^m(\cos\theta) \tag{S4}$$

where ℓ is a non-negative integer ($0 \leq \ell$), and m is an integer within the interval $-\ell \leq m \leq \ell$. $P_\ell^m(\cos\theta)$ is the Legendre polynomial of degree ℓ and order m, which, together with the complex exponential $e^{im\phi}$, define sinusoidal functions over the angles θ and ϕ. In quantum mechanics, spherical harmonics are used to represent the orbital angular momenta, e.g., for an electron in a hydrogen atom. In this context, the degree ℓ relates to the eigenvalue of the square of the angular momentum, and the order m is the eigenvalue of the angular momentum about the azimuthal axis.

The operators that describe how spherical harmonics transform under rotations are called the Wigner D-matrices, denoted by $D^\ell_{mm'}(R)$ (13); this operator is a matrix because the rotation group in 3D is not commutative. Under a rotation R, spherical harmonics transform as

$$Y_{\ell m}(\theta, \phi) \xrightarrow{R} \sum_{m'=-\ell}^{\ell} D^\ell_{m'm}(R) Y_{\ell m'}(\theta, \phi) \tag{S5}$$

Since spherical harmonics transform in this well-defined way, we can now examine how a rotation acts on the Fourier transform of a signal over the sphere. Suppose we have a signal $f(\mathbf{n})$ defined on the elements $\mathbf{n}$ of the spherical shell S^2 (i.e., the set of angular coordinates for points on the sphere). The Fourier coefficients associated with the rotated signal $f_R(\mathbf{n}) = f(R\mathbf{n})$ follow,

$$\begin{aligned}
\hat{F}^R_{\ell m} = \int Y_{\ell m}(\mathbf{n}) f_R(\mathbf{n}) \mathrm{d}\Omega \quad &= \int Y_{\ell m}(\mathbf{n}) f(R\mathbf{n}) \mathrm{d}\Omega \\
&= \int Y_{\ell m}(-R\mathbf{n}') f(\mathbf{n}') \mathrm{d}\Omega' \\
&= \int \sum_{m'=-\ell}^{\ell} D^\ell_{m'm}(-R) Y_{\ell m}(\mathbf{n}') f(\mathbf{n}') \mathrm{d}\Omega' \\
&= \sum_{m'=-\ell}^{\ell} D^\ell_{m'm}(-R) \int Y_{\ell m}(\mathbf{n}') f(\mathbf{n}') \mathrm{d}\Omega' \\
&= \sum_{m'=-\ell}^{\ell} D^\ell_{m'm}(-R) \hat{F}_{lm}
\end{aligned} \tag{S6}$$

where $\mathrm{d}\Omega = \sin^2\theta \, \mathrm{d}\theta \, \mathrm{d}\phi$ is the angular differential in the spherical coordinate system. Under a rotation R about the origin, the spherical Fourier coefficients of a real-space signal transform according to $\hat{F}_{\ell m} \xrightarrow{R} \sum_{m'=-\ell}^{\ell} D^\ell_{m'm}(-R) \hat{F}_{\ell m'}$, which is simply a matrix product.

It should be noted that the Wigner D-matrices form an irreducible representation of SO(3), which is the group of all rotations around a fixed origin in three-dimensional Euclidean space. Therefore, it provides a natural basis to expand functions in SO(3) while respecting the rotational symmetry.

Nonlinear transforms respecting rotational equivariance. One key feature of neural networks is applying nonlinear activations, which enable a network to approximately model complex and non-linear phenomena. Commonly used nonlinearities include reLU, tanh, and softmax functions. However, these conventional nonlinearities can break rotational equivariance in the Fourier space. To construct expressive rotationally equivariant neural networks we can use the Clebsch-Gordan tensor product, which is the natural nonlinear (or bilinear in the case of using two sets of Fourier coefficients) operation in the space of spherical harmonics (13).

Given two spherical tensors $\hat{F}_{\ell_1}$ and $\hat{G}_{\ell_2}$, we can take a product between all of their components $\hat{F}_{\ell_1 m_1}\hat{G}_{\ell_2 m_2}$, which would allow us to express nonlinearities. Although these products do not behave well under rotations, linear combinations of them do transform in well-defined ways. The Clebsch-Gordan coefficients are the coefficients that decompose these products back into the space of spherical harmonics, where Wigner D-matrices define the rotation operations. Specifically, the product of spherical tensors $\hat{F}_{\ell_1}$ and $\hat{G}_{\ell_2}$ will yield spherical tensors of degree L such that $|\ell_1 - \ell_2| < L < \ell_1 + \ell_2$ in the following way,

$$\hat{H}_{LM} = \sum_{m_1=-\ell_1}^{\ell_1} \sum_{m_2=-\ell_2}^{\ell_2} \langle LM|\ell_1 m_1; \ell_2 m_2\rangle \hat{F}_{\ell_1 m_1} \hat{G}_{\ell_2 m_2} \qquad [S7]$$

where $\langle LM|\ell_1 m_1; \ell_2 m_2\rangle$ is a Clebsch-Gordan coefficient and can be precomputed for all degrees of spherical tensors (13). Similar to spherical harmonics, Clebsch-Gordan products also appear in quantum mechanics, and they are used to express couplings between angular momenta. In following with recent work on group-equivariant machine learning (12, 14–16), we will use Clebsch-Gordan products to express nonlinearities in 3D rotationally equivariant neural networks for protein structures.

B. Rotationally equivariant encoding of protein micro-environments as spherical holograms. We define an amino acid's micro-environment as all the atoms associated with the surrounding amino acids within a 10 Å of the central residue's α-carbon. We use the position of the central residue's α-carbon as the reference point of the neighborhood $\mathcal{N}$. Each atom i within this neighborhood, located at position $\mathbf{r}_i$ with respect to the reference point, has three attributes associated with it: (i) the element type of the atom, $e_i \in \{C, N, O, S, H\}$, (ii) the partial charge of the atom Q_i, and (iii) the SASA A_i of the atom. With these characteristics we define a feature vector v_i^c where c indexes the input channels $\{C, N, O, S, H, \text{charge}, \text{SASA}\}$. This vector takes on values given by

$$v_i^c = \begin{cases} \delta_{e_i,c} & \text{for } c \in \{C, N, O, S, H\} \\ Q_i & \text{for } c = \text{charge} \\ A_i & \text{for } c = \text{SASA} \end{cases} \qquad [S8]$$

where δ_{ij} is a Kronecker delta function, and takes the value 1 if $i = j$ and 0 otherwise. Thus, for $c \in \{C, N, O, S, H\}$ the feature vector v_i^c takes value 1 if atom i is of type c and 0 otherwise. Q_i and A_i denote the partial charge and the SASA (in units of Å^2) of atom i, respectively. These quantities are computed by PyRosetta for each protein structure (2).

Each channel defines a point cloud with the attributed values stored at the coordinates of the corresponding atoms. We express the point cloud of each channel in a spherical coordinate system with the origin set at the alpha carbon of the central amino acid in the neighborhood. We then define an atomic density as a sum of Dirac delta distributions parameterized by each atomic coordinate in the neighborhood $\rho^c(\mathbf{r}) = \sum_{i \in \mathcal{N}} v_i^c \delta^{(3)}(\mathbf{r} - \mathbf{r}_i)$. Here $\delta^{(3)}(\mathbf{x})$ denotes a normalized Dirac delta probability density in 3D, which is zero everywhere except at the origin $\mathbf{x} = \mathbf{0}$, where it is infinite. We use 3D Zernike transforms to encode the point clouds associated with each channel in a 3D rotationally equivariant fashion in the spherical Fourier space:

$$\hat{Z}^c_{n\ell m} = \int \rho^c(\mathbf{r}) Y_{\ell m}(\theta, \phi) R_{n\ell}(\mathbf{r})\, \mathrm{d}\Omega. \qquad [S9]$$

where $\mathrm{d}\Omega$ is the angular differential in the spherical coordinate system, $Y_{\ell m}(\theta, \phi)$ is the spherical harmonics (eq. S4), and $R_{n\ell}(\mathbf{r})$ is the radial function of the 3D Zernike transform, which can be expressed as,

$$R_{n\ell}(\mathbf{r}) = (-1)^{\frac{n-\ell}{2}} \sqrt{2n+3} \binom{\frac{n+\ell+3}{2} - 1}{\frac{n-\ell}{2}} |\mathbf{r}|^{\ell}\, {}_2F_1\left(-\frac{n-1}{2}, \frac{n+\ell+3}{2}; \ell + \frac{3}{2}; |\mathbf{r}|^2\right) \qquad [S10]$$

where ${}_2F_1(\cdot)$ is the ordinary hypergeometric function. The index $n \geq 0$ is a non-negative integer, and the radial function $R_{n\ell}(\mathbf{r})$ is nonzero only for even values of $n - \ell \geq 0$. Zernike polynomials form a complete orthonormal basis in 3D and, therefore, can be used to expand and retrieve any 3D shape, if large enough ℓ and n coefficients are used. Approximations that restrict the series to finite n and ℓ are often sufficient for shape retrieval, and hence, desirable algorithmically. Importantly, expansion of an input signal by Zernike polynomials (eq. S9) is rotationally equivariant since a rotation R transforms the resulting coefficients through Wigner-D matrices as $\hat{Z}^c_{n\ell m} \xrightarrow{R} \sum_{m'=-\ell}^{\ell} D^{\ell}_{m'm}(-R) \hat{Z}^c_{n\ell m'}$ (eq. S5).

Zernike projections in the spherical Fourier space can be thought of as a superposition of spherical holograms of an input point cloud, and thus, we term this operation the *holographic encoding* of protein micro-environments.

C. Rotationally equivariant model of protein micro-environments with H-CNN. We use the coefficients of the Zernike expansion $\hat{Z}^c_{nlm}$ in eq. S9 (i.e., the holographic encoding of the data) as inputs to a rotationally equivariant convolutional neural network to classify amino acids based on their surrounding atomic micro-environments. We term this network holographic convolutional neural network (H-CNN).

The convolutional neural network that we use for our analysis is similar to the Clebsch-Gordan network (CGNet), described in ref. (14). This network is built out of three rotationally equivariant modular units: (i) linear transformation of spherical harmonics, (ii) Clebsch-Gordan nonlinearity, and (iii) spherical batch normalization.

Below we will describe the computations done in each of these modular equivariant units. Without a loss of generality, we denote the operations in each unit of the network in terms of a general equivariant signal $\hat{F}^k_{lm}$ where k is a catch-all index that represents all non-rotational indices (i.e., channel c and index n in eq. S9), and ℓ and m correspond to the angular indices (eq. S4). The output of each unit will be denoted by $\hat{G}^k_{lm}$.

Linear transformations in CG-nets. Linearities are ubiquitous in neural networks and contribute partially to their powerful expressiveness. Our linearity takes the form

$$\hat{G}^k_{\ell m} = \sum_{k'} W^{(\ell)}_{kk'} \hat{F}^{k'}_{\ell m} \tag{S11}$$

where $W^{(\ell)}_{kk'}$ denotes two dimensional matrices that mix different input channels k' (e.g., combinations of radial index n and channel c in the input or general learned representations in intermediate layers) producing different output channels k. To preserve rotational symmetry, we will impose the restriction of only combining information (i.e., forming linear combinations of inputs) that belongs to the same irreducible representations (irreps) of SO(3) in eq. S11. In other words, we mix only inputs associated with the same degree ℓ and order m of spherical harmonics. These inputs are mixed across channels associated with different chemical or radial information in the first layer, or composite channels in the following layers. Generally, different weights are learned for each irrep ℓ but shared across coefficients of different order m.

Clebsch-Gordan nonlinearity. The last equivariant component of the network is a Clebsch-Gordan nonlinearity. Nonlinearities are ubiquitous in neural networks. The Clebsch-Gordan nonlinearity is the simplest equivariance-respecting nonlinearity, and it is expressive enough for most learning tasks (14, 16). This Clebsch-Gordan nonlinearity is simply a product of spherical tensors that is decomposed back into the spherical harmonic basis via the Clebsch-Gordan coefficients,

$$\hat{H}^K_{LM} = \sum_{m_1=-\ell_1}^{\ell_1} \sum_{m_2=-\ell_2}^{\ell_2} \langle LM|\ell_1 m_1; \ell_2 m_2\rangle \hat{F}^{k_1}_{\ell_1 m_1} \hat{G}^{k_2}_{\ell_2 m_2} \tag{S12}$$

where $\langle LM|\ell_1 m_1; \ell_2 m_2\rangle$ is a Clebsch-Gordan coefficient and $K = (k_1, k_2)$ is the new channel index.

Generally we have a choice for which combinations of values k_1, k_2, ℓ_1, and ℓ_2 are used in this product. We focus on two choices for these indices. We define *fully-connected* networks, for which k_1, k_2, ℓ_1, and ℓ_2 are allowed to take on all possible values. We also define *simply-connected* networks, for which we impose the condition that $k_1 = k_2$ and $\ell_1 = \ell_2$.

The exact choice made in combining these indices affects the dimensionality of the network. The width of the Clebsch-Gordan nonlinearity output in a fully-connected network with a maximum spherical degree $L_{\max}$ and a width d_h (i.e., $k \in \{1, ..., d_h\}$) is given by

$$\Omega^{\text{full}}_{\ell, L_{\max}} = d_h^2 \left(\left[\frac{1}{4}(2 + \ell(5+\ell) - 2\left\lfloor \frac{1}{2}(\ell - 1) \right\rfloor \right] + (\ell+1)(L_{\max} - \ell) \right) + (\ell+1)(L_{\max} - \ell) \tag{S13}$$

Meanwhile, for a simply-connected network with the same $L_{\max}$ and d_h, the width of the Clebsch-Gordan nonlinearity is

$$\Omega^{\text{simple}}_{\ell, L_{\max}} = d_h(L_{\max} + 1 - \ell) \tag{S14}$$

Spherical batch normalization. Batch normalization is a common feature in neural networks that allows for smooth training of the network. Since the output of our nonlinear Clebsch-Gordan product is not bounded, batch normalization becomes extremely important to ensure that activations remain finite throughout training. We impose a batch normalization layer after each linear operation, producing normalized inputs to the nonlinear operation (Clebsch-Gordan product). We use the following batch normalization during training,

$$\hat{F}^k_{\ell m} = \frac{\hat{F}^k_{\ell m}}{\sqrt{\langle |\hat{F}^k_\ell|^2 \rangle_{\text{batch}} + \epsilon}} \tag{S15}$$

where $|\hat{F}^k_\ell|^2 = \sum_{m=-\ell}^{\ell} \hat{F}^k_{\ell m} \hat{F}^{*k}_{\ell m}/(2\ell+1)$, * denotes complex conjugation, $\langle \cdot \rangle_{\text{batch}}$ denotes averaging over a batch, and ϵ is a small number used to ensure numerical stability, which we set to $\epsilon = 10^{-3}$. During training the network stores a moving average of the norm for each ℓ in each channel as

$$N^{\mu,i}_\ell = \xi \langle |\hat{F}^k_\ell|^2 \rangle_{\text{batch}} + (1-\xi) N^{\mu,i-1}_\ell \tag{S16}$$

where ξ serves as the momentum of our moving average, set to $\xi = 0.99$. This moving average is then used as the norm during testing in the following way:

$$\hat{F}^\mu_{\ell m} = \frac{\hat{F}^\mu_{\ell m}}{\sqrt{N^{\mu,i}_\ell + \epsilon}} \tag{S17}$$

The different batch normalization used for training and testing prevents the batching of the validation data to affect the evaluation of the model accuracy during testing.

3. Network and training procedure for H-CNN

Network architecture. A complete H-CNN features an equivariant unit of Clebsch-Gordan layer followed by an invariant unit of dense layers as detailed in Fig. S2. A Clebsch-Gordan layer is comprised of an equivariant linearity, a spherical batch norm, a Clebsch-Gordan product, and a concatenation. Invariants from the inputs as well as all outputs of the each CG layer are collected for use in the invariant dense layers (Figs. 1, S2).

Training procedure. Networks were trained with all training data being read at run time using the keras `fit` function and a data generator made from a subclass of the `keras.utils.sequeunce` class (17). Generally 40 workers were used for reading the data and a `max_queue_size` of 900 was found to be optimal for read/evaluation balance. Approximately one tenth of the validation data (51,200 neighborhoods) was evaluated after an approximate one tenth of the training data (256,000 neighborhoods) was seen. This was implemented by using `tf.keras.Model.fit` arguments `steps_per_epoch=1000` and `validation_steps=100`. Data was shuffled at the end of each evaluation interval. These evaluation intervals also served as checkpoints for both early stopping and weight saving. The best network was trained for 8.47 epochs in 4.54 hours on one NVIDIA A40 GPU hosted at the Hyak supercomputer at the University of Washington.

We used stochastic gradient descent and Adam optimizer (18) for training our networks. Adam generally showed better performance, and thus, all networks were trained with Adam during hyperparameter optimization. The main component of the loss was defined as the categorical cross entropy between the one-hot amino acid encoding and the softmax output of the network

$$\mathcal{L} = -\sum_{i=1}^{N} \log p_{\alpha_i} \qquad [S18]$$

where p_{α_i} is the network-assigned probability of the true central amino acid α_i in neighborhood i.

Some hyperparameters were scheduled according to the behavior of the network's loss. The learning rate was put on a schedule of `ReduceLROnPLateau` to decrease by a factor of 0.1 whenever the validation loss did not improve over 10 epochs, with a minimum learning rate of 10^{-9}. Ultimately, the networks were trained with early stopping when the validation loss did not improve by a difference of 0.01 over the course of 20 epochs.

Hyperparameter optimization. Hyperparameter optimization was performed in two steps: First, we scanned a larger parameter regime and trained 500 networks to identify a reasonable subregion of the parameter space (Fig. S3A). This sampling revealed networks with parameter ranges listed in Table S1. These parameters were then sampled more finely to get the best network models. The resulted training curves are shown in Fig. S3B,C.

It should be noted that we optimized over two classes of networks: (i) fully-connected networks, and (ii) the simply-connected networks, as defined in Section C. In simply-connected networks, the nonlinear Clebsch-Gordan products are restricted to the square of Fourier coefficients with the same ℓ and channel index, in contrast to the fully-connected networks that use bilinear products between all sets of Fourier coefficients (see equation Eq. (S12)). These restricted nonlinearities allowed us to use linear layers that had roughly one order of magnitude larger output dimensions in simply vs. fully-connected networks (150 as opposed to 20); see Fig. S3E. We separately optimized the hyperparameters for these two network classes.

Model performance. For both the simply-connected and the fully-connected networks, we find the best hyperparameters that minimize the validation loss function in equation S18. The best simply-connected model had 62% classification accuracy and a minimal validation loss of 1.2 while the best fully-connected model had 68% classification accuracy and a minimal validation loss of 0.91. This difference in predictive power of the models appeared consistent across all training scenarios. Due to the superior predictive power of the best fully-connected network compared to the best simply-connected network, we used the best fully-connected network to evaluate the stability and binding prediction tasks in main text.

We compare the accuracy and the training efficiency of our model to the existing methods that classify amino acids based on all-atom representations of their local environments in Table 1. For this comparison, we used the metric of testing accuracy of the amino acid class since this was more commonly reported in the literature. Specifically, we compare H-CNN with two 3D CNN methods that used conventional convolutional neural networks on voxelized protein structures (19, 20). Notably, these 3D CNN methods require local alignment of neighborhoods to the central amino acid. Also, each of these methods uses a cube of side length 20 Å, while spheres of radius 10 Å are used as inputs to H-CNN. Our method thus sees approximately half ($\pi/6$) of the volume that these other models see. Overall, our model has a comparable accuracy to the best 3D CNN method (20), but it is much more efficient in training.

We also compare our model to other methods that prioritize rotational symmetry, i.e., the Spherical CNN introduced in ref. (21) and the 3D Steerable CNN from ref. (22). Spherical CNN (21) is approximately rotationally invariant by performing 3D convolutions on a discretized grid over the 3D sphere. 3D Steerable CNN (22) is a rotationally invariant method by applying spherical convolutions in a steerable basis. H-CNN performs better than both of these methods in the classification task with an order of magnitude fewer parameters (Table 1).

Ablation study on the importance of charge and SASA. We studied the effect of our post-processing of atomic neighborhoods by performing ablation studies on charge and SASA. For each ablation, we retrained networks on inputs that did not include either charge or SASA information. During retraining, we reran the second stage of our hyperparameter optimization using the same learned bounds from the original hyperparameter optimization and only considered fully-connected networks. The overall accuracy from removing SASA was 57% while the overall accuracy from removing charge was 56%, in contrast to the 68% accuracy of the complete model shown in Fig. 2. A similar 10% drop in accuracy associated with SASA and charge was previously reported in ref. (20). Fig. S7 shows the confusion matrices for each model compared to the best model's confusion matrix.

4. Robustness of H-CNN performance to train/test set splits

We tested the robustness of H-CNN performance to similarities between proteins in the train and test sets. To quantify similarities between proteins, we use TM-score, which is a metric for assessing the topological similarity of protein structures (23). We assign max-TM-score to each protein in our test set, which is the TM-score of that protein its closest match in the training set (i.e., the one with the highest TM-score). Fig. S4A shows that the accuracy of H-CNN in predicting the amino acid identities in the test set proteins is only weakly dependent on the proteins' max-TM-score. Fig. S4B shows that H-CNN sequence recovery suffers in a small number of proteins with max-TM-score < 0.5 (i.e., those significantly distinct from the training set (24)). However, for the majority of cases the H-CNN recovery rate is insensitive to the TM-score; the average amino acid recovery rate for test proteins with max-TM-score < 0.5 is 66% which is comparable to 69% for those with max-TM-score ≥ 0.5.

To further assess the robustness of H-CNN, we retrained our network using the ProteinMPNN's CATH-based splitting of the PDB (4), which aimed to minimize structural similarity in the training, validation and testing sets. Fig. S4C shows the resulting confusion matrix, which yields a 68% accuracy in amino acid recovery, consistent with the H-CNN model trained on the ProteinNet's splits in Fig. 2.

These results indicate that H-CNN has learned a generalizable model for protein micro-environments that can be used for proteins far from its training set, with possible applications to *de novo* protein structures.

5. Comparing H-CNN with evolutionary models for amino acid preferences

As shown in the main text, the interchangeability of amino acids predicted by H-CNN is consistent with the evolutionary substitution patterns reflected by the BLOSUM62 matrix (Fig. 2D). In addition to this coarse-grained measure, we also assessed the consistency of substitutions at the single-site level between the H-CNN predictions and evolutionary data. Evolutionary variation at a given site in a protein reflects both the substitutability of amino acids at that position (e.g., due to their common physico-chemical properties) and the epistatic interactions with the other covarying residues in the protein.

To isolate the site-specific effects of substitutions, we used the software EV-Couplings (25) to infer a model for amino acid preferences (and their couplings), using amino acid covariations in multiple sequence alignments (MSAs) of protein families. Specifically, EV-Couplings infers a maximum likelihood Potts model for interactions between residues in a protein sequence (26). From this model, we can evaluate the relative preference for occurrence of different amino acids at a given position in a protein sequence.

We used EV-Couplings to infer models of amino acid preferences for all of the protein domains in our test set. For each protein chain in our test data, we used EV-Couplings to compute the MSA for the protein associated with UniProt ID listed under the associated PDB entry. The MSA is computed using UniRef90 (27) that clusters homologous proteins such that sequences within each cluster have at least 90% identity to and 80% overlap with the longest sequence (seed). For inference of the Potts model, we used the default EV-Couplings settings for the rest of the parameters. Ultimately we were able to infer evolutionary models on 67 protein families corresponding to 67 chains in our testing set. These protein families amounted to 11,221 unique sites, for which we compare the EV-Couplings predictions on amino acid preferences with the H-CNN predictions in Fig. 2F.

To measure the similarity between the predictions of H-CNN and EV Couplings, we define the profile overlap ρ for amino acid preferences at a given position. Specifically, for two sets of amino acid probabilities $P_A, P_B \in [\mathbb{R}^+_{\leq 1}]^{19}$, the vector overlap is defined as the normalized dot-product between the centered probability vectors,

$$\rho = \frac{\big(P_A - \langle P_A\rangle_{\text{sites}}\big)\cdot\big(P_B - \langle P_B\rangle_{\text{sites}}\big)}{\sqrt{|P_A - \langle P_A\rangle_{\text{sites}}|^2 |P_B - \langle P_B\rangle_{\text{sites}}|^2}} \tag{S19}$$

where $\langle\cdot\rangle_{\text{sites}}$ denotes the average of probability profile overs all sites in the data. Profile overlap takes values between -1 and 1, with 1 indicating a perfect alignment between the two (centered) vectors of amino acid preferences, and -1 indicating complete disagreement.

6. Effect of shear perturbation on protein micro-environments

We locally perturbed protein structures around their crystal conformation with the shearing deformation. As demonstrated in Fig. 3, shearing by an angle δ at a given residue of a protein is a perturbation to the backbone dihedral angles at two neighboring residues given by

$$(\phi_i, \psi_{i-1}) \longrightarrow (\phi_i + \delta, \psi_{i-1} - \delta) \tag{S20}$$

This perturbation was chosen because it can substantially change the local protein structure near the residue of interest while minimally affecting the far-away residues.

Physical interactions in a protein should be dependent on the pairwise distances between atoms. We define D_{ab} to be the pairwise distance in angstroms between a pair of atoms (a, b) in the protein structure. Then $\Delta D^{i,\delta}_{ab} = D^{i,\delta}_{ab} - D^0_{ab}$ measures the deviation of these pairwise distances from the pairwise distances in the native (unperturbed) structure D^0_{ab}, when site i is sheared by an angle δ (Fig. 3). We estimate the total shear-induced distortion in a protein structure as the root-mean-square of the deviations of pairwise distances RMS$\Delta D^{k,\delta}_{ij}$; here, the average is taken over all pairs of atoms that were at distances smaller than 10 Å in the unperturbed structure, i.e., $\{(a, b) \mid D^0_{ab} < 10\,\text{Å}\}$.

We characterize the model's response to the perturbation by the change in the protein network energy (Fig. 3), defined as the sum of pseudo-energies (logits) of amino acids over all sites in the protein,

$$E = \sum_{i\in\text{sites}} \tilde{E}^{\alpha_i}_i \tag{S21}$$

where α_i denotes the protein's amino acid at site i.

As shown in Fig. 3D, the native protein structure appears to be at the energy minimum of the network and shear perturbations often result in less favorable protein micro-environments. To assess the robustness of this result, we evaluated the protein network energy in eq. S21 for alternative amino acids, while keeping the surrounding protein structure intact (i.e., using the wild-type neighborhood for model evaluation). Fig. S8 shows that if amino acid substitutions preserve the physico-chemical properties of the residue (i.e., substitutions according to the BLOSUM62 matrix), the undistorted structure remains close to the energy minimum, yet the landscape becomes shallower. However, for random amino acid substitutions, no energy minima is recovered. These results further suggest that H-CNN has learned an effective physical potential for protein micro-environments.

7. Alternative structure-based models

We compared H-CNN to other equivariant, invariant, and non-symmetry-aware structure-based models that can be used for similar tasks. Specifically, H-CNN was compared to 3D CNNs that use grid convolution on voxelized protein structures (one from *Torng et al* (19) and one from *Shroff et al* (20) which we refer to as 3D CNN in Table 1 and MutCompute in Table 2 in the main text), Spherical CNNs that use spherical convolutions on protein structure data (one from *Boomsma et al* (21) and one from *Blaabjerg et al* (28) which we refer to as Spherical CNN-RaSP in the Table 2, since it is part of a larger model for predicting protein stability), Steerable CNN (22), and the message-passing ProteinMPNN (4).

3D CNNs are extensions of traditional CNNs in three dimensions. They do not respect rotational symmetry and require alignment of atomic environments along backbones. Spherical CNNs generalize convolutions to a spherical grid although due to the nature of discretizing the sphere, they are only approximately rotationally equivariant. Steerable CNNs utilize fully equivariant steerable kernels and are rotationally equivariant. ProteinMPNN is rotationally invariant operating on invariant pairwise distances of backbone atoms.

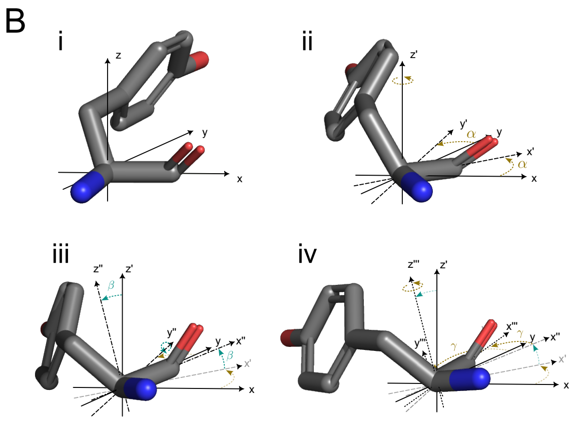
In Table 1, we compare H-CNN architecture, training time, and performances to networks for which these metrics were reported. All values were taken from the literature when accessible. In Table 2, we compare H-CNN and its ensembled estimates to models that were available for predicting the mutational effects on protein stability of binding. MutCompute (i.e., 3D CNN-Shroff) predictions were taken from the webserver https://mutcompute.com/. Spherical CNN-RaSP predictions were made using the Google Colab notebook associated with ref. (28). The full RaSP model is trained to predict protein stability. To compare zero-shot predictions in Table 2, Spherical CNN-RaSP predictions were obtained from only using the cavity model part of the network which was trained on the amino acid prediction task only. ProteinMPNN predictions were made using the github repository associated with ref. (4).

A

$$R_z(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$

$$R_{euler}(\alpha, \beta, \gamma) = R_z(\gamma) R_y(\beta) R_z(\alpha)$$



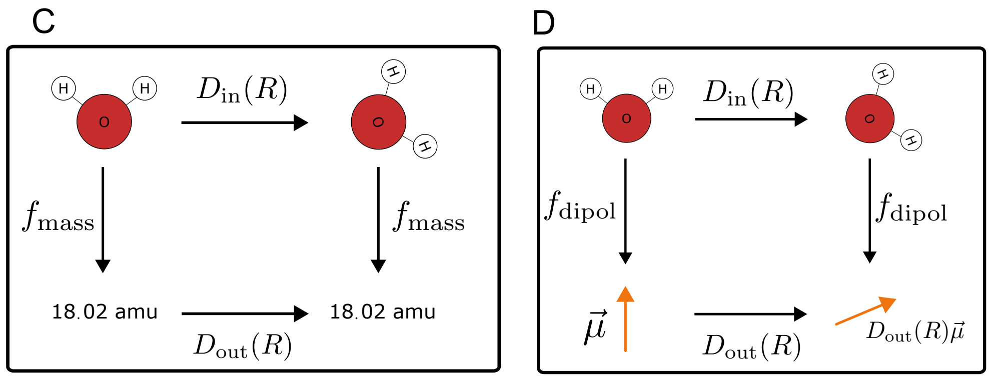


Fig. S1. Parameterization of rotations and rotational equivariance. We use x-y-z Euler angles to parameterize rotations in 3D. This parametrization **(A)** can be written in a matrix form, and **(B)** it can be visualized on a 3D object. Starting from a given orientation in (B.i), the object's rotation can be broken down into three steps: First, rotation around the z-axis of angle α (B.ii). Second, rotation around the new y-axis by angle β (B.iii). And finally, rotation around the new z-axis by angle γ (B.iv). A map from the geometric descriptions of an object to physical observables can be invariant or equivariant under rotations. **(C)** A rotationally invariant map, such as f_{mass} which gives the mass of a molecule, remains unchanged under the molecule's rotation. **(D)** A rotationally equivariant map, such as f_{dipol} which gives the dipole moment of a molecule, transforms in a well-defined way with the molecule's rotation, denoted by the operator $D_{\text{in}}(R)$. Specifically, in this case, the transformation operator for the dipole moment under rotation $D_{\text{out}}(R)$ acts linearly on the dipole moment in the original frame and is uniquely determined by the parameters of the rotation R.

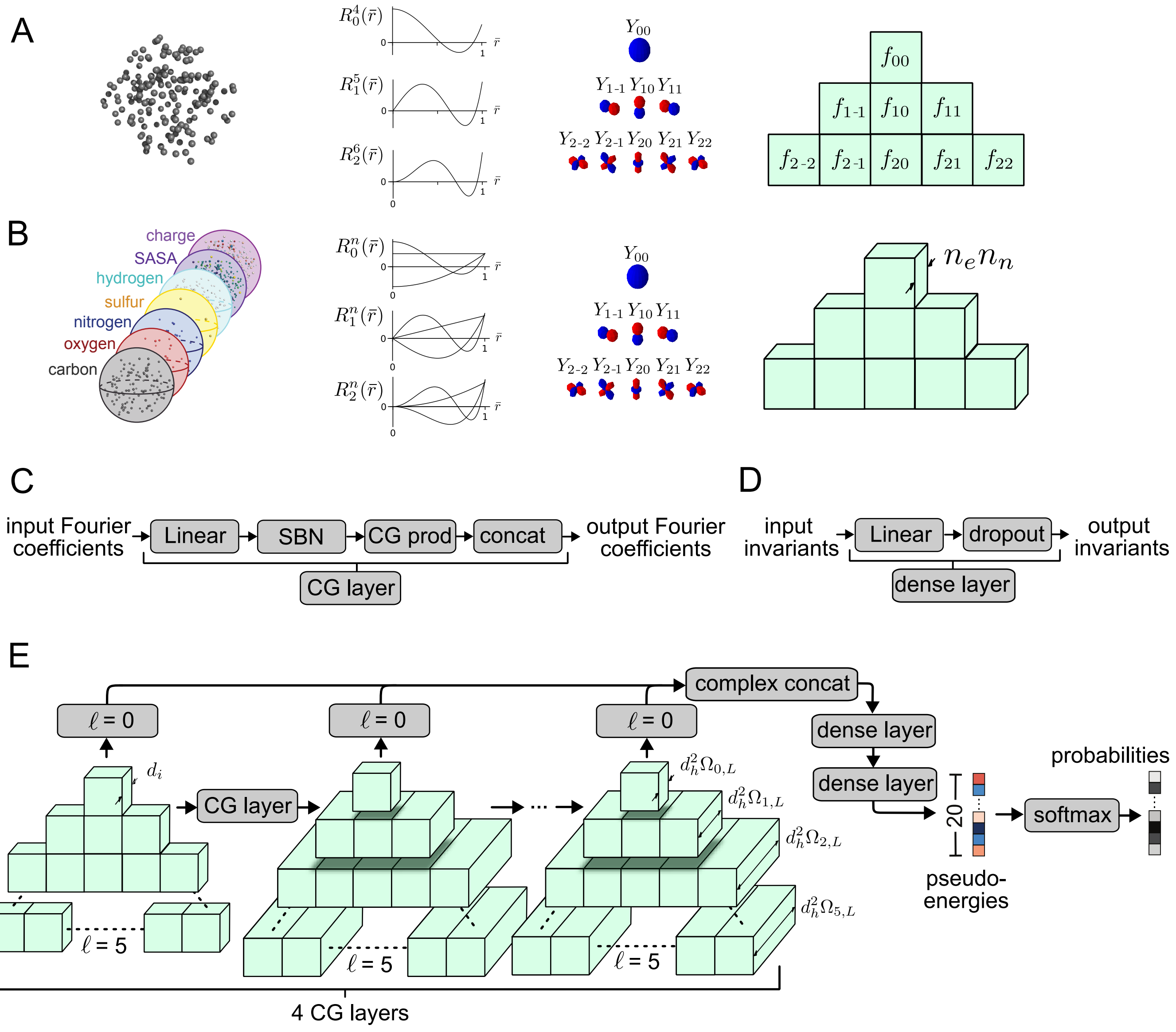
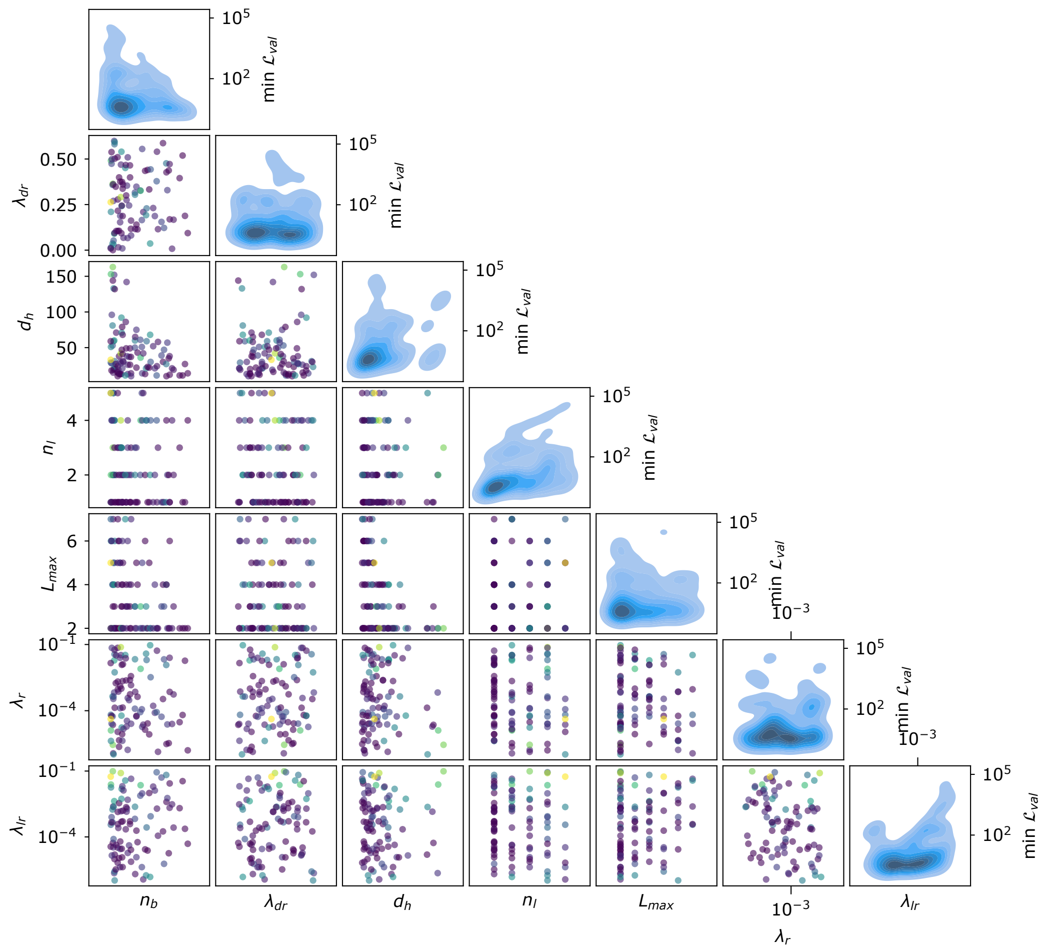
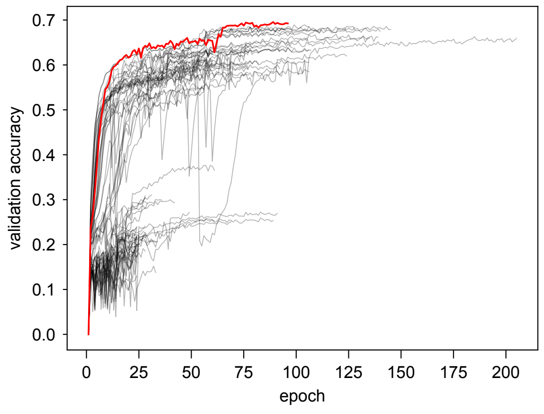


Fig. S2. Schematics of rotationally equivariant data encoding and network architecture in H-CNN. (A) Any point cloud (left) can be written in the equivariant spherical harmonic Fourier basis by integrating the point cloud density against the spherical harmonics along with a choice of a radial function (middle), which depends on the spherical harmonic degree ℓ. The triangular structure (right) demonstrates the resulting Fourier coefficients $f_{\ell m}$, in which the rows reflect the different degrees of spherical harmonics $0 \leq \ell$, which are non-negative integers. Individual cells within each row correspond to different orders m of the spherical harmonics, which are integers bounded by $-\ell \leq m \leq \ell$. **(B)** The Fourier transform becomes three dimensional when radial resolution is required thereby increasing the number of discrete n-values used n_n. The multiplicity of coefficients is also determined by the number of point clouds being transformed n_ρ (e.g., the different atomic channels on the left). Since both the point clouds and the radial information are invariant under rotations, these two dimensions can be combined in index to give preference in organizing information according to the degree ℓ and order m of the spherical harmonics. **(C)** The Clebsch-Gordan block (CG block) is fully equivariant and is comprised of a linear layer, a spherical batch norm (SBN), Clebsch-Gordan product (CG prod), and a degree-wise concatenation (concat). **(D)** Throughout all layers of the network, all invariants ($\ell = 0$) are collected and fed into a series of dense layers which are simple linear combinations with training dropout. **(E)** These two main components in (C,D) comprise the bulk of the entire network. First the input Fourier coefficients are processed through successive CG layers. Invariants are collected from the original input and from the output of every CG layer. The real and imaginary components of these invariants are split and concatenated (complex concat) and then fed into a series of dense layers. This schematic reflects the dimensions of the optimal model discovered in this work. Four CG layers were used. We note that the maximal width of the network as determined by the output of the nonlinearity depends on ℓ due to the selection rules of the CG prod. We denote this width $d_h^2\Omega_{\ell,L}$ where L is the maximal degree ℓ used in the network and are given in eqs. (S13) and (S14). Two dense layers were used in the optimal model. The dimensions of these layers are (4852×500) and (500×20). The output of these dense layers are termed pseudo-energies and act as logits in a softmax to estimate probabilities for each amino acid. We emphasize the only nonlinearities in the network are the Clebsch-Gordan products and the ultimate softmax.

A



B



C

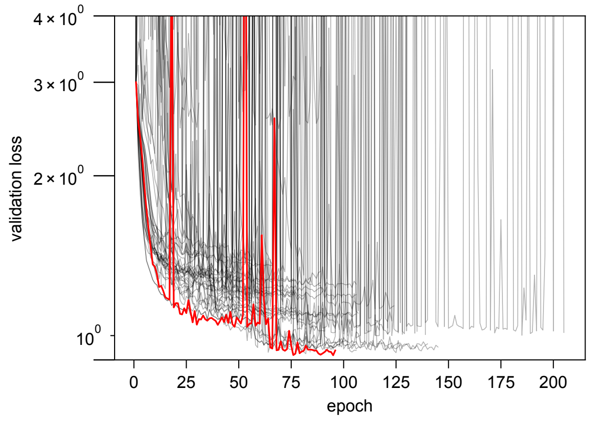


Fig. S3. Hyperparameter optimization for H-CNN. Hyperparameter optimization was performed in two steps for both the simply-connected and the fully-connected networks: First, hyperparameters were scanned across a wide regime of hyperparameters via random sampling. From this scan, a narrow band of hyperparameters were analyzed to determine the optimal network (**A**). Off the diagonal, each dot's color shows the validation loss seen during this first step of hyperparameter tuning for a network with the combination of hyperparameters shown on the two axes. On the diagonal, the density of networks is shown as a function of loss and the hyperparameter tested. Validation accuracy (**B**) and loss (**C**) are shown as a function of training epochs for all networks trained in the second stage of hyperparameter tuning. Red lines show the network with the best validation loss that is used throughout this work.

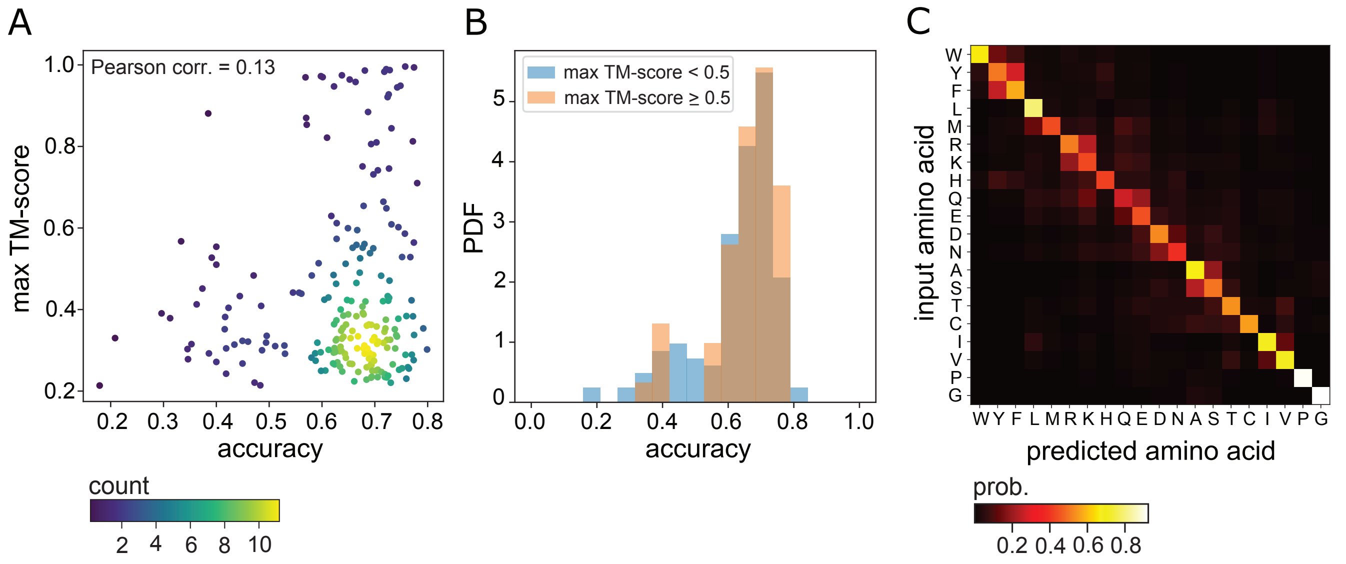


Fig. S4. Robustness of H-CNN performance to train/test set splits. **(A)** The TM-score of each of the 214 proteins in the test set with their respective closest match in the training set (max-TM-score) is shown against the average accuracy of H-CNN for amino acid retrieval in that protein. Colors indicate the density at each point (number of proteins). The retrieval accuracy is weakly dependent on the similarity (max-TM-score) between the test proteins and the training set, with Pearson correlation of 0.13. **(B)** The distributions of H-CNN retrieval accuracies for test proteins with max-TM-score < 0.5 (i.e., those significantly distinct from the training set), and with max-TM-score ≥ 0.5 (i.e., those similar to the training set) are shown. The average amino acid retrieval accuracy is comparable between the two sets, with 0.66 for proteins with max-TM-score < 0.5, and 0.69 for those with max-TM-score ≥ 0.5. **(C)** Shown is the confusion matrix for amino acid retrieval from the retrained H-CNN, using ProteinMPNN's CATH-based splitting of the PDB, which is consistent with that of the H-CNN trained on the ProteinNet's splits in Fig. 2. The accuracy in amino acid recovery of both H-CNN models is 68%.

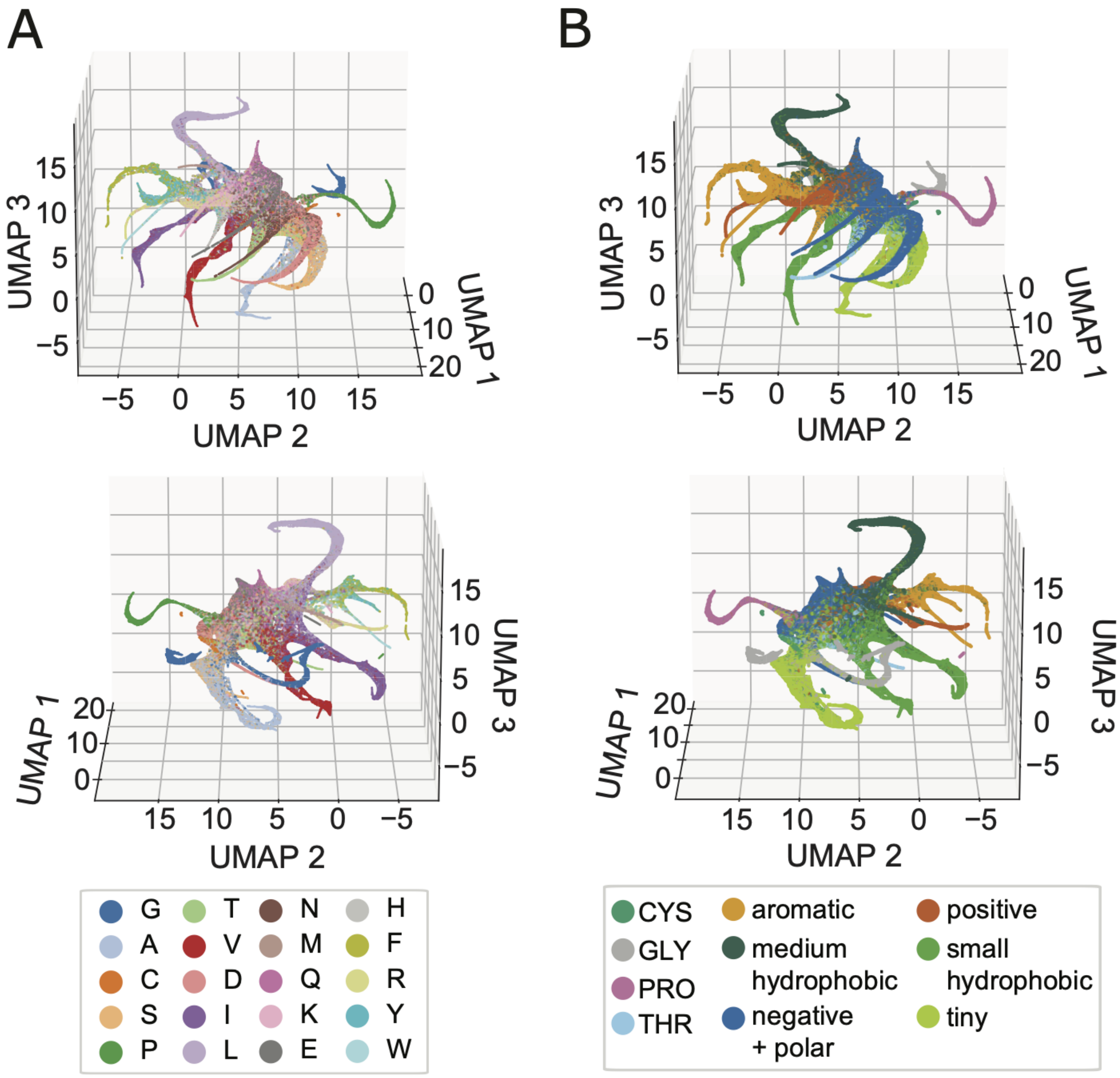


Fig. S5. Low dimensional mapping of H-CNN likelihood profiles reveals physico-chemical clustering. Low dimensional projections of the prediction outputs (3D UMAPs) are shown. UMAPs are annotated by **(A)** the amino acid types, and **(B)** the physico-chemical clusters in (Fig. 2A), with top/bottom panels showing a different view of the UMAP in each case. Neighborhoods are closely clustered by amino acid types (A), and are spatially arranged based on the physico-chemical properties of the side-chains (B). An interactive version of these plots are provide on the github page: https://github.com/StatPhysBio/protein_holography/tree/89c11e3160d0053308e8d84263e1c61a6df13807/protein_holography/publication

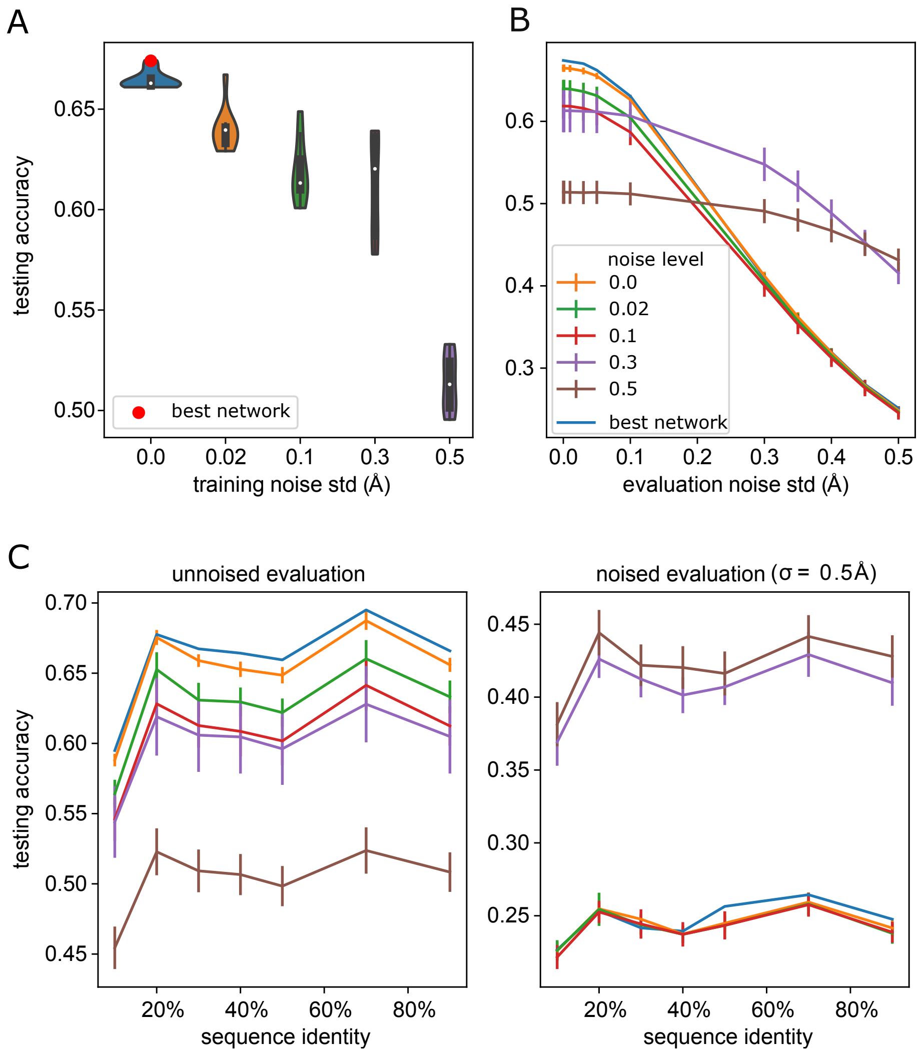
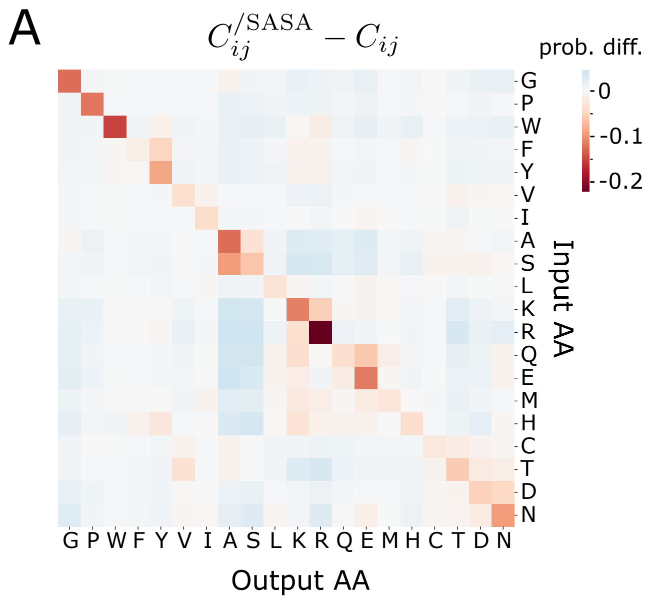


Fig. S6. Impact of noise on H-CNN performance. **(A-B)** H-CNN's performance on the test set was assessed as a function of Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ added to the coordinates of atoms in protein neighborhoods of the training data, as well as the noise added to the test data at evaluation time. **(A)** Violin plots show the distribution of testing accuracies as a function of the amplitude (standard deviation σ) of the training noise injected. The accuracy is measured based on the amino acid retrieval in an un-noised test data. White dots show the median testing accuracy, thick black lines show the interquartile range, and thin black lines show the extent of the distribution up to 1.5x the interquartile range. The limits of each violin show the true limits of the empirical distribution. 10 networks were used to create each distribution. Additionally, the network with the best validation loss trained on un-noised data (used in the main text) is indicated in red and labeled as "best network". **(B)** For the same sets of noised networks in (A), the accuracy of the model predictions are shown as a function of noise added to the coordinates of atoms in test set (evaluation noise on the horizontal axis). The performance of the best network with best validation loss is displayed as a line (blue), while the rest of the traces correspond to the mean testing accuracy of a given ensemble, trained with a specified training noise amplitude (colors) . Errorbars show one standard deviation. **(C)** H-CNN performance on the test set was assessed as a function of the sequence similarity of the training and the test sets, using different slicings from ProteinNet. Performance is assessed for networks trained on data with different levels of Gaussian noise injected to the coordinates of the atoms in protein neighborhoods (colors according to (B)), and tested on un-noised (left) and noised data with amplitude $\sigma = 0.5$ Å (right). For slicing with sequence identity $> 10\%$, H-CNN performance is insensitive to the exact splitting of sequences in training and test sets. We use 30% split for all analyses done in the manuscript.



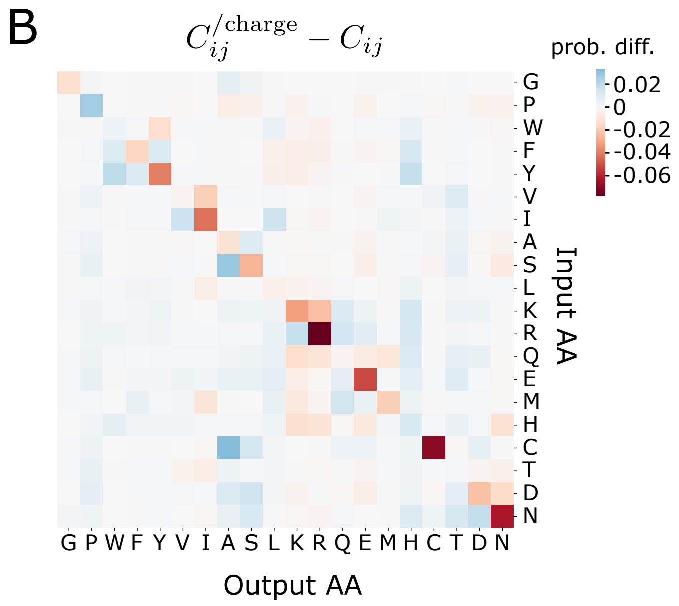
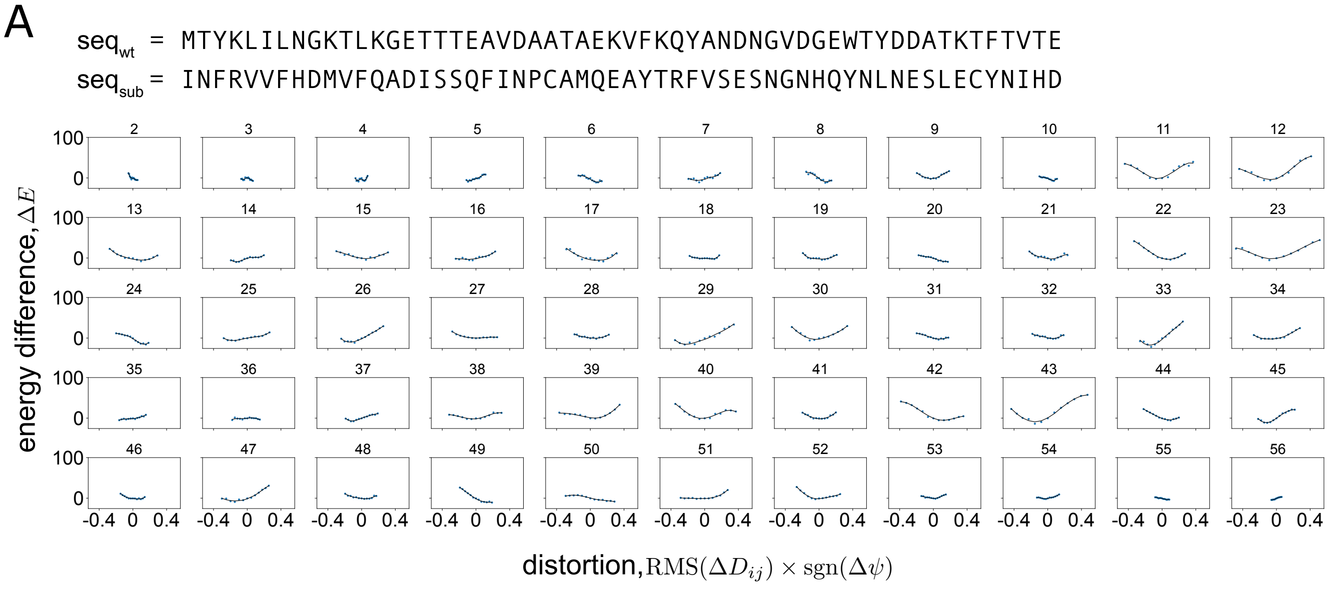


Fig. S7. H-CNN performance after charge and SASA ablation. The difference in the confusion matrix for networks with ablated **(A)** SASA and **(B)** charge with respect to the network with the full information (Fig. 2A) is shown. Negative (red) values demonstrate less probability assigned to the input/predicted pair when the input is ablated, while positive (blue) values show increased probability mass under ablation. The removal of SASA (A) appears to decrease the network's ability to predict all amino acids as seen on the diagonal. The effect is most pronounced for hydrophobic amino acids in the upper left, with some hydrophilic amino acids (R, K, E) also displaying strong effects. Interestingly, most of the probability mass is on average redistributed to small amino acids A and S. When charge is removed (B), the network demonstrates worse predictions on charged and polar amino acids most notably R, C, N, and E. The overall network's accuracy after removing SASA and Charge are 57% and 56%, respectively.

A



B

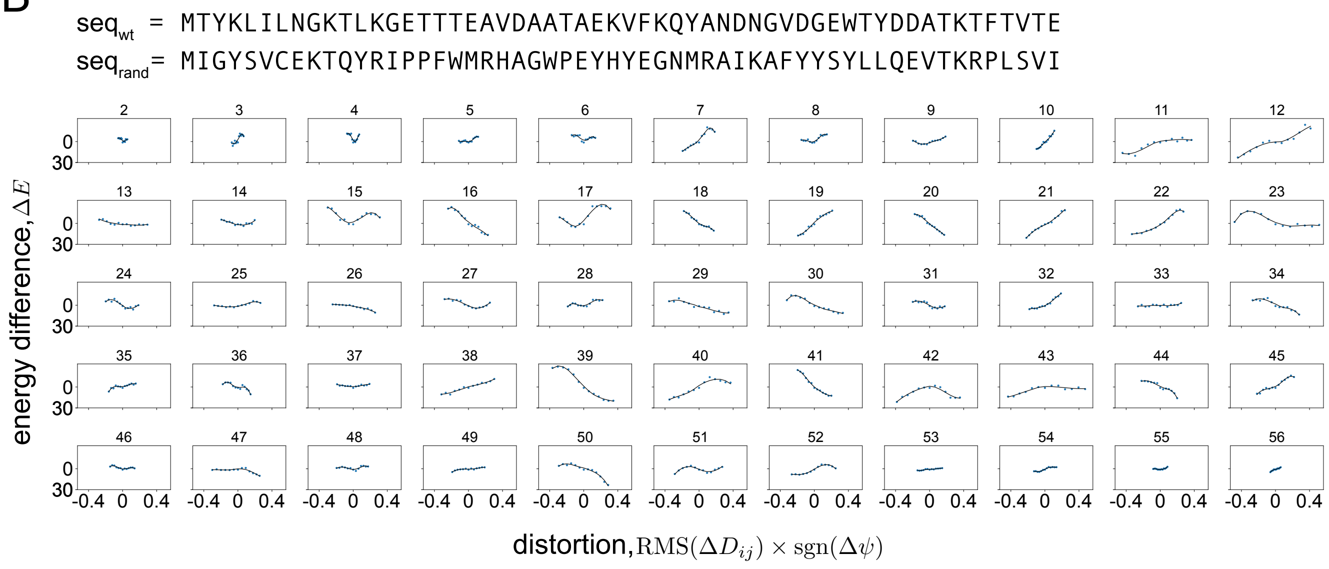
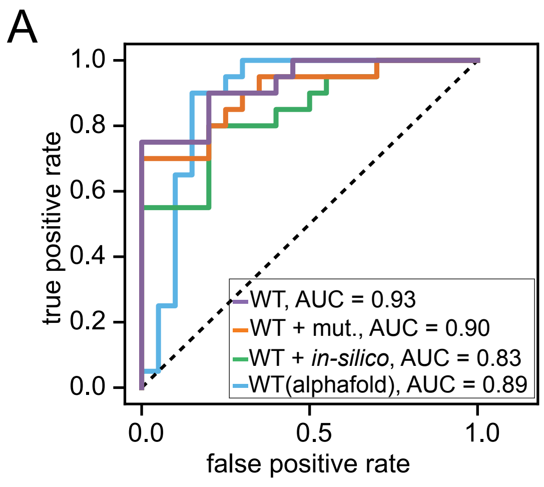


Fig. S8. Robustness of response to shear perturbation. Similar to Fig. 3, but the change in network energy is evaluated for alternative amino acids at the shear position, while keeping the surrounding protein structure intact (i.e., using the wild-type neighborhood for model evaluation). The alternative amino acids are drawn according to the BLOSUM62 matrix in **(A)**, and randomly in **(B)**; the wild-type and the alternative sequence are shown each panel. When amino acids are substituted according to their BLOSUM62 scores, potential wells are generally recovered (A), while no energy minima is recovered for random substitutions (B). Specific sequences used are given by seq$_{\mathrm{sub}}$ (substitutions according to the BLOSUM62 matrix), and seq$_{\mathrm{rand}}$ (random substitutions) in each panel and are shown in comparison to the wildtype sequence seq$_{\mathrm{wt}}$.



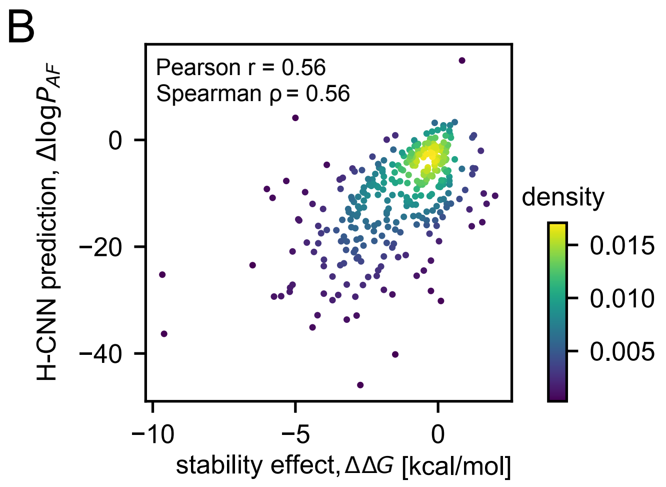


Fig. S9. H-CNN predictions for the stability effect of all available single point mutations in T4 Lysozyme. (A) The true positive vs. false positive rates (ROC curve) is shown for classification of the 40 amino acid mutations (from Fig. 4A) into deleterious and neutral/beneficial classes based on the H-CNN predicted log-probability ratios $\Delta \log P$, using the wild-type and the mutant structures (orange), only the wild-type structure (purple), the wild-type structure for the wild-type and an *in silico* relaxed structure for each mutant (green), and the AlphaFold predicted structure of the wild-type (blue). The corresponding area under the curves (AUCs) are reported in the figure. **(B)** H-CNN predictions for the relative log-probabilities using the AlphaFold predicted structure $\Delta \log P_{\text{AF}}$ are shown against the experimentally measured $\Delta\Delta G$ values for 310 single point mutation variants of T4 Lysozyme. Mean $\Delta\Delta G$ was used when multiple experiments reported values for the same variant. The colors show the density of points in each panel as calculated via Gaussian kernel density estimation. H-CNN predictions correlate well with the experimental values, with the Pearson and Spearman correlations indicated in each panel.

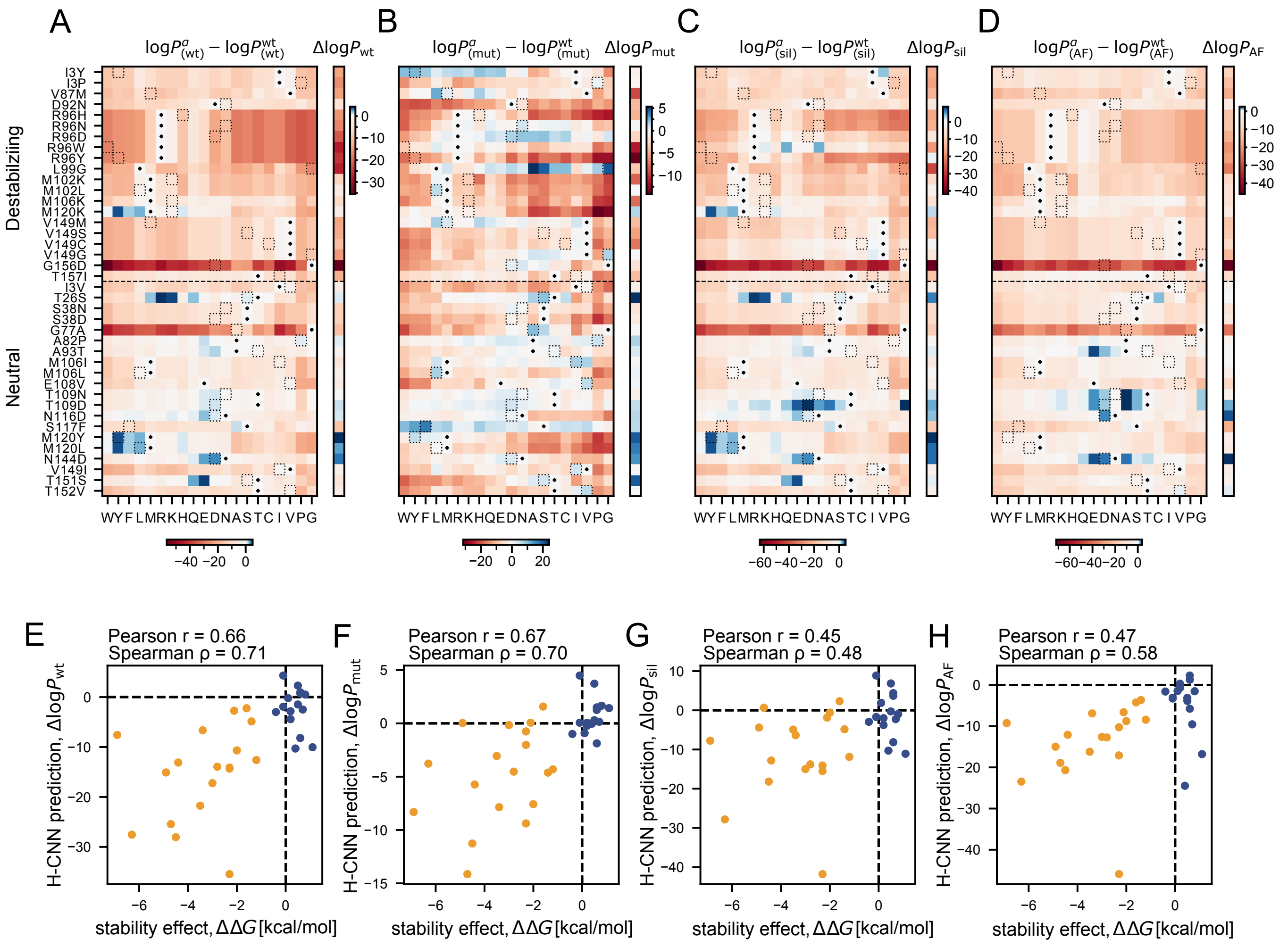


Fig. S10. Predictions for the stability effect of mutations in T4 Lysozyme with different protein structures. (A-D) The large heatmaps show the H-CNN predicted log-probability of different amino acids relative to the wild-type at the substituted positions along the T4 Lysozyme, indicated on the left-hand side of panel A (similar to Fig. 4A). In each heatmap, the dots indicate the identity of the wild-type amino acid, and the dotted squares indicate the amino acid in the specified position for the mutated variant in each row. The predictions for log-probabilities are evaluated using the wild-type structure (A), the mutant crystal structures (B), the wild-type structure relaxed *in-silico* via PyRosetta for the specified substitution (C), and the computationally predicted structure of the wild-type T4 lysozyme using AlphaFold (D). The relative log-probabilities of the mutations of interest are shown separately in one column next to each heatmap. They are calculated with the structural information that is available in different scenarios. For the wild-type, mutant, and *in silico* structures, $\Delta \log P_* = \log P^{\text{mut}}_{(*)}/P^{\text{wt}}_{(\text{wt})}$, with $*$ indicating the specified protein structure in each case. This definition is akin to $\Delta\Delta G$. For the AlphaFold structure, we use $\Delta \log P_{\text{AF}} = \log P^{\text{mut}}_{(\text{AF})}/P^{\text{wt}}_{(\text{AF})}$—a quantity which could be evaluated in the absence of any experimental structural knowledge for a protein. **(E-H)** The predicted relative log-probabilities shown in panels (A-D) are plotted against the experimentally measured $\Delta\Delta G$ values for each variant. Destabilizing mutations are shown in orange and neutral/beneficial mutations are shown in blue. Overall, H-CNN predictions correlate well with the mutational effects on protein stability, with the Pearson and Spearman correlations indicated in each panel and AUC values to discriminate between destabilizing and neutral mutations indicated in Fig. 4D. Accounting for changes in the local protein structures due to mutations in (B, C) sets the correct reference point in predictions such that the estimated log-probability ratio is overall positive for neutral/beneficial mutations and negative for destabilizing mutations.

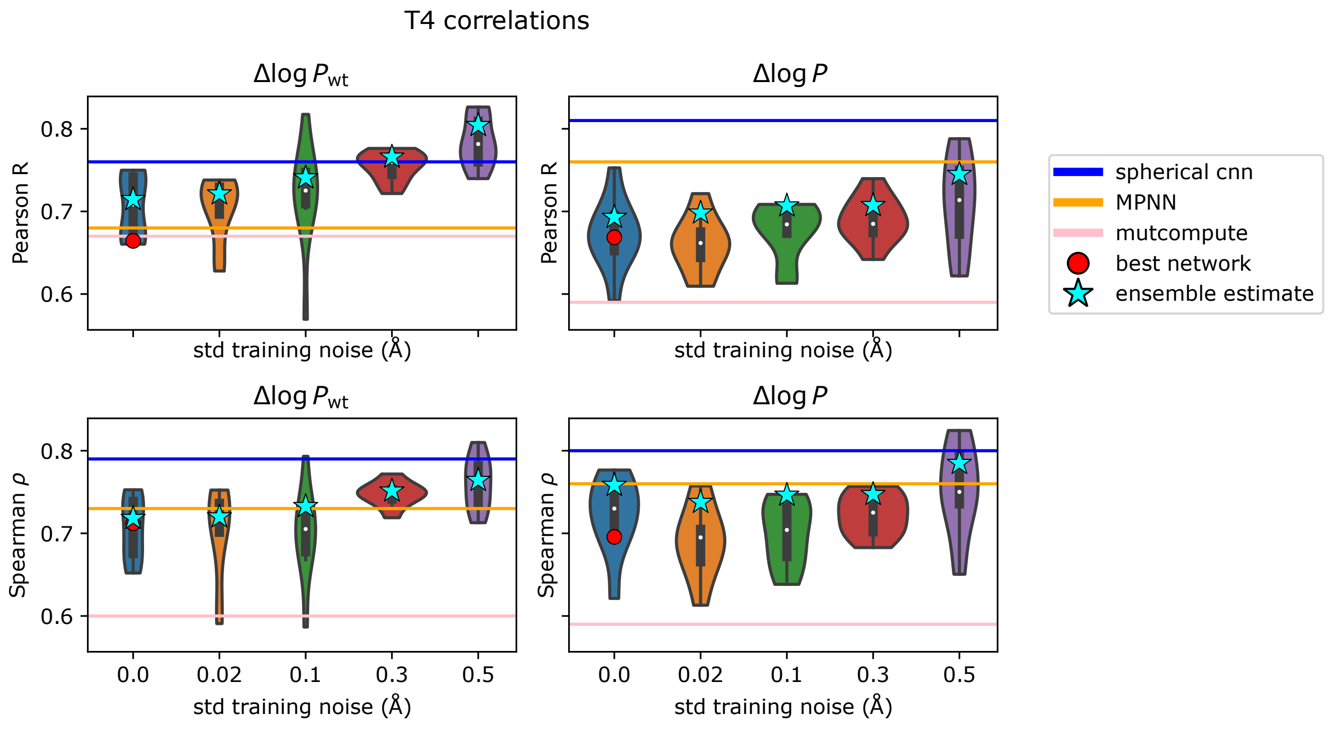


Fig. S11. **Impact of training noise on predicting the stability effect of mutations in T4 Lysozyme.** Violin plots show the distributions of Pearson and Spearman correlations between the experimentally evaluated $\Delta\Delta G$ of T4 Lysozyme mutations and the H-CNN predicted values, using only the wild-type structure $\Delta \log P_{\mathrm{wt}}$ (left panels), and both the wild-type and mutant structures $\Delta \log P$ of the 40 variants (right panels), for different amplitudes of noise in the training data. White dots show the median, thick black lines show the interquartile range, and thin black lines show the extent of the distribution up to points outside 1.5x the interquartile range. The prediction from the best un-noised network (i.e., with minimum validation loss) is shown as a red circle. Predictions from models trained at a specific noise level were averaged to determine an ensemble estimate. Correlations between these ensemble estimates with experimental data are indicated by cyan stars. Corresponding correlations from other zero-shot methods are shown with horizontal lines (see Table 2).

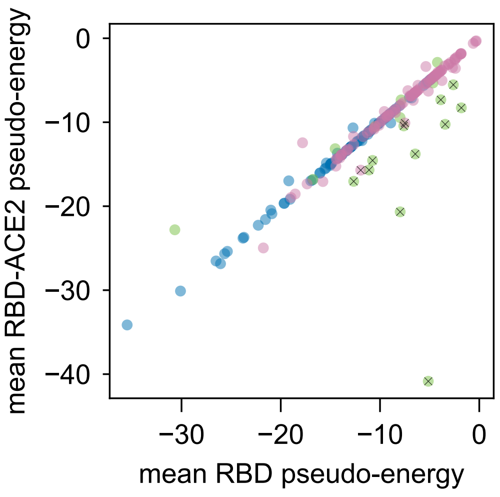


Fig. S12. H-CNN predictions for stability and binding of RBD. The H-CNN predicted mean pseudo-energy per site predicted from the RBD-ACE2 protein complex (measure of binding) is plotted as a function of the mean pseudo-energy per site, inferred from the isolated RBD structure (measure of stability). Points are colored according to their experimental values of expression and binding as illustrated in Fig. 5. Most points lie on the diagonal since the RBD pseudo-energy is determined from the RBD-ACE2 crystal structure with the ACE2 masked. However, sites at the RBD-ACE2 interface show deviations from the diagonal. H-CNN prediction for sites that are tolerant of mutations for stability but not binding are indicated by crosses (below the diagonal points with large stability values). These sites tend to be at the RBD-ACE2 interface and overlap with the green category from Fig. 5.

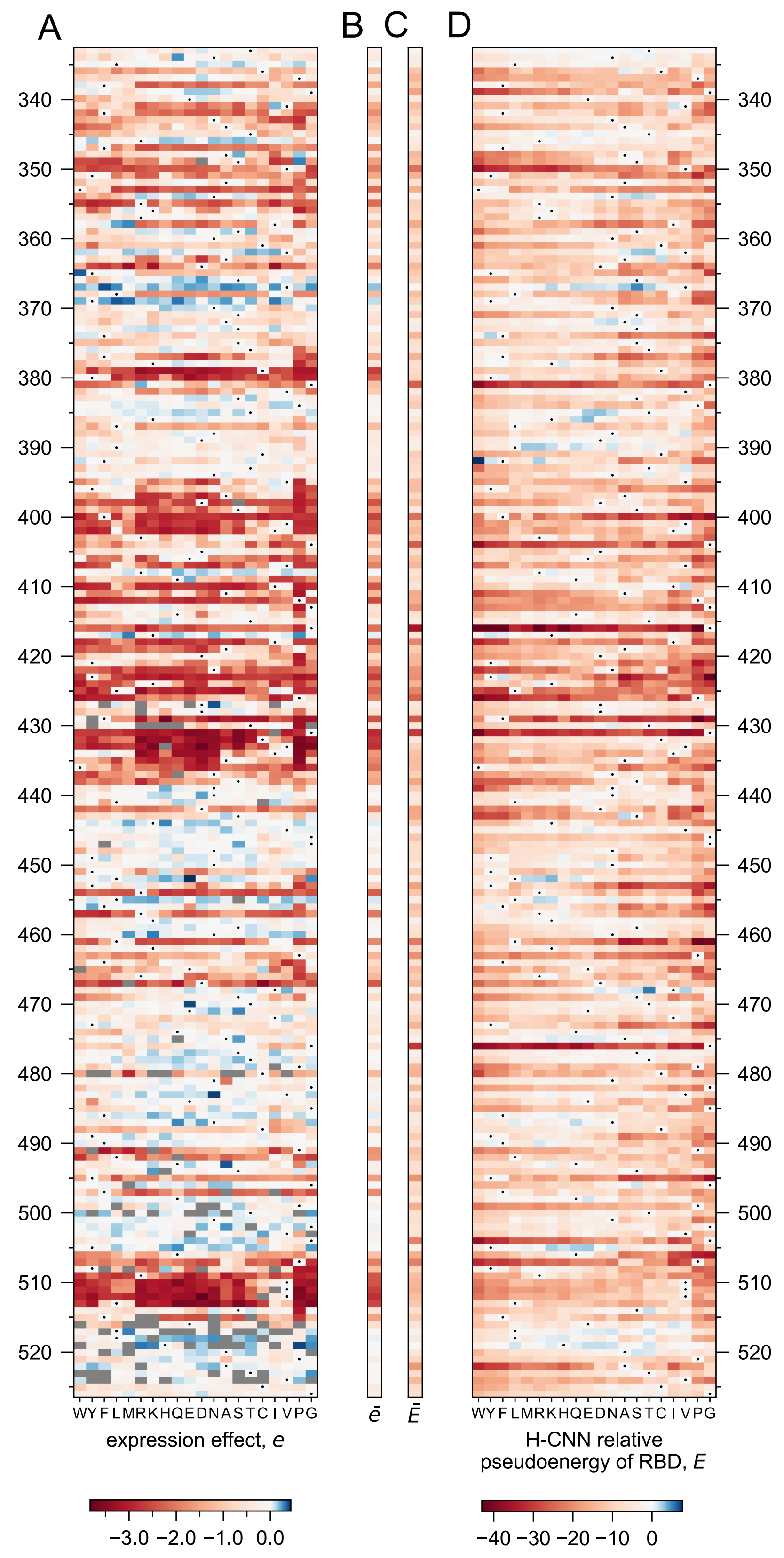


Fig. S13. Experimental RBD expression of SARS-CoV-2 in DMS experiments and the H-CNN predictions. (A) Experimental expression of single-point mutation variants of SARS-CoV2 RBD relative to the wildtype (dots) as measured using yeast surface display experiments. Expression effect e is calculated as $\Delta \log_{10}(\text{MFI})$ where MFI is the mean fluorescent intensity. Mutations to G or P dominate the negative tail of expression effects. To better show the effect of all mutations, the negative portion of the colorbar is bounded by the expression of the most detrimental variant that is not a mutation to G or P. **(B)** The mean expression effect of mutations $\bar{e}$ is listed next to each site. **(C,D)** The H-CNN predicted stability effect of mutations (D), and the mean predicted stability effect for each site $\bar{E}$ (C), evaluated based on the computationally isolated RBD structure, are shown. Again the negative portion of the colorbar is bounded from below by the most destabilizing prediction not involving mutations to G or P.

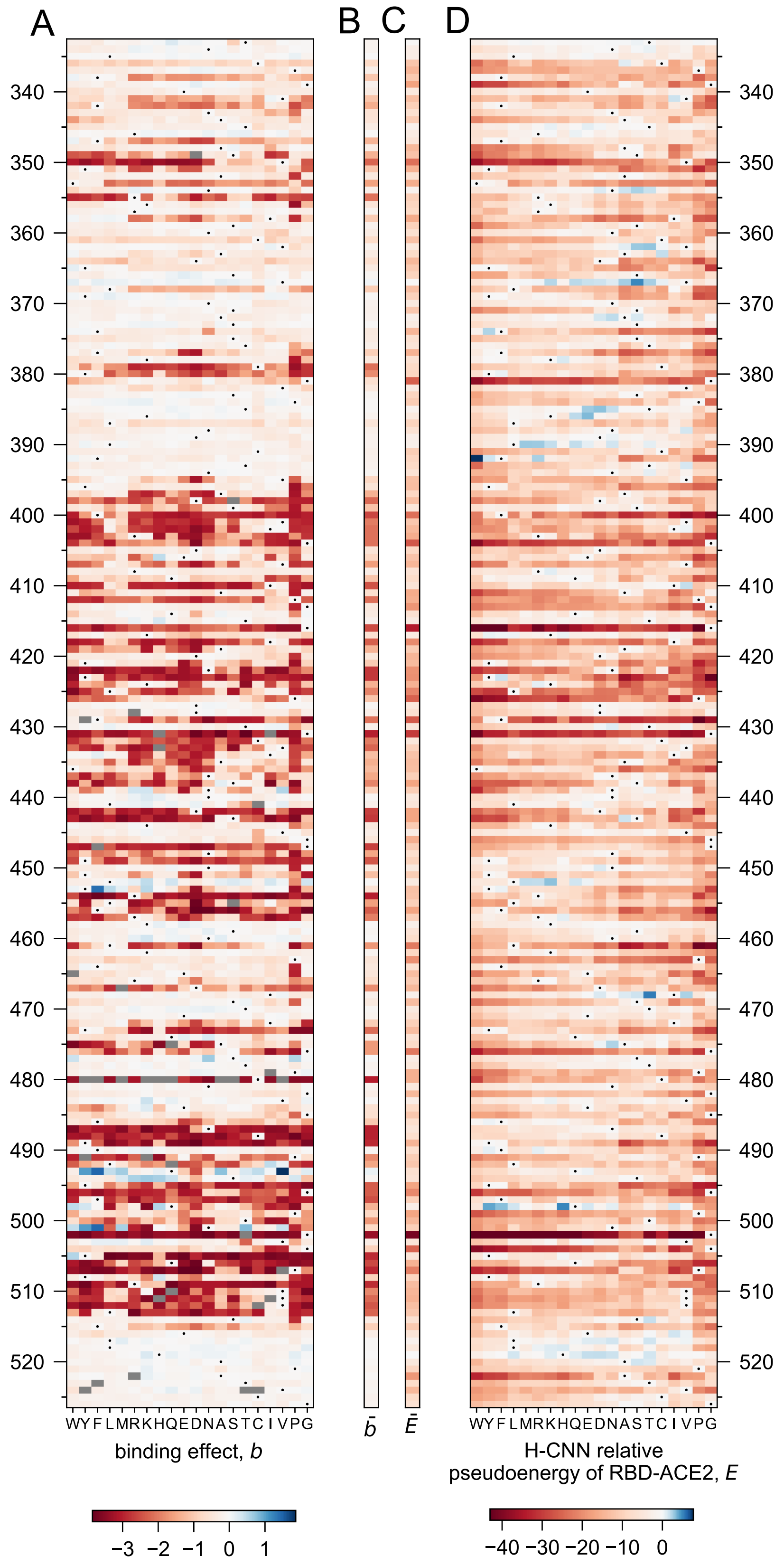


Fig. S14. Experimental RBD-ACE2 binding affinity in DMS experiments and the H-CNN predictions. **(A)** Experimental binding of single-point mutation variants of SARS-CoV2 RBD to the human ACE2 receptor relative to the wildtype (dots) as measured using yeast surface display experiments. Binding effect b is reported from $\Delta \log K_D$ where negative values represent weaker binding relative to the wildtype. Mutations to G or P dominate the negative tail of expression effects. To better show the effect of all mutations, the negative portion of the colorbar is bounded by the expression of the most detrimental variant that is not a mutation to G or P. **(B)** The mean binding effect of mutations $\bar{b}$ is listed next to each site. **(C,D)** The H-CNN predicted effect of mutations (D) and the mean predicted stability effect for each site $\bar{E}$ (C), evaluated based on the crystal structure of RBD-ACE2 complex, are shown. Again the negative portion of the colorbar is bounded from below by the most destabilizing prediction not involving mutations to G or P.

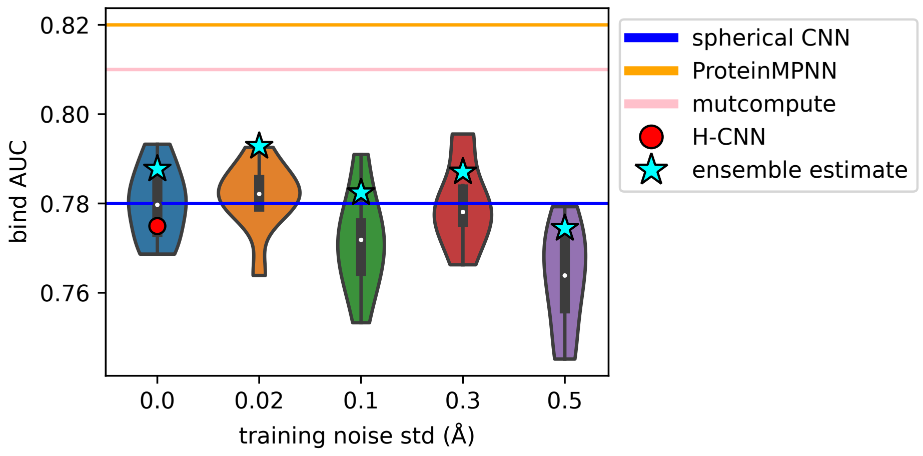


Fig. S15. Impact of training noise on predicting the effect of mutations on SARS-CoV-2 RBD binding to ACE2 receptor. Violin plots show the distribution of AUROCs for classifying mutations in bound vs. unbound for H-CNN models trained at various noise levels, indicated on the horizontal axis. White dots show the median, thick black lines show the interquartile range, and thin black lines show the extent of the distribution up to points outside 1.5x the interquartile range. The extent of the violins reflects the true extent of the data outliers included. The AUROCs associated with the best un-noised network (i.e., with minimum validation loss) is indicated by a red circle; see Fig. 5. Predictions from models trained at a specific noise level were averaged to determine an ensemble estimate. AUROCs based on these ensemble estimates are indicated by cyan stars. Corresponding AUROCs from other zero-shot methods are shown with horizontal lines (see Table 2).

Hyperparameter	variable	broad sampling space	sampling scheme	fine sampling space	optimal fc network value	optimal sc network value
batch size	n_b	[8, 5000]	uniform	256	256	256
dropout rate	λ_{dr}	[0,0.6]	uniform	$[0, 10^{-5}]$	5.49×10^{-4}	1.95×10^{-2}
hidden dimension	d_h	fc:[6,20], sc:[50,160]	uniform	fc:[8,20], sc:[100,500]	14	109
No. CG layers	n_l	[1,5]	uniform	[4,5]	4	5
No. dense layers	n_d	[1,5]	uniform	[2,4]	2	2
Max spherical degree	L_{max}	[2,7]	uniform	[4,5]	5	5
learning rate	λ_l	$[10^{-6}, 10^{-1}]$	exponential	10^{-3}	10^{-3}	10^{-3}
regularization strength	λ_r	$[10^{-6}, 10^{-1}]$	exponential	$[10^{-10}, 10^{-16}]$	1.2×10^{-16}	1.89×10^{-14}

Table S1. Hyperparameter bounds and optimal values. Hyperparameters varied during model optimization are listed. The two different bounds used during the two steps of hyperparameter optimization are shown in the broad and fine sampling spaces. Different bounds for fully-connected (fc) and simply-connected (sc) networks are shown when appropriate. Finally, the optimal value is listed for both the optimal fully-connected and simply-connected networks.

variant	I3Y	I3P	V87M	D92N	R96H	R96N	R96D	R96W	R96Y	L99G	M102K	M102L	M106K	M120K	V149M	V149S	V149C	V149G	G156D	T157I
PDB id	1L18	1L97	1CU3	1L55	1L34	3CDT	3C8Q	3FI5	3C80	1QUD	1L54	1L77	231L	232L	1CV6	1G06	1G07	1G0P	1L16	1L10
$\Delta\Delta G$ [kcal/mol]	-2.3		-2.3	-1.4		-3.0	-3.5	-4.5	-4.7	-6.3	-6.9	-2.11	-3.4	-1.6	-2.8	-4.4	-2.0	-4.9	-2.3	-1.2
pH	6.5		5.4	5.7 -5.9		5.35	5.35	5.35	5.35	5.4	5.3	5.7	3	3	5.4	5.4	5.4	5.4	6.5	6
source	(29)	(30)	(31)	(32)	(33)	(34)	(34)	(34)	(34)	(35)	(36)	(37)	(38)	(38)	(31)	(39)	(39)	(39)	(40)	(41)

variant	I3V	T26S	S38N	S38D	G77A	A82P	A93T	M106I	M106L	E108V	T109N	T109D	N116D	S117F	M120Y	M120L	N144D	V149I	T151S	T152V
PDB id	1L17	131L	1L61	1L19	1L23	1L24	129L	1P46	234L	1QUG	1L59	1L62	1L57	1TLA	1P6Y	233L	1L20	1G0Q	130L	1G0L
$\Delta\Delta G$ [kcal/mol]	-0.4		0.6		0.4	0.8		0.2	0.5	0.7	0.1	0.6	0.6	1.1	-0.1	0.5		-0.1		0.2
pH	6.5	5.4	5.7 -5.9	5	6.5	6.5	5.4	5	3	5.4	5.7 -5.9	5.7 -5.9	5.7 -5.9	5.4	5	3	5	5.4	5.4	5.4
source	(29)	(42)	(32)	(43)	(44)	(44)	(42)	(45)	(38)	(35)	(32)	(32)	(32)	(46)	(45)	(38)	(43)	(39)	(42)	(39)

Table S2. T4 Lysozyme variants. Each of the 40 variants, shown in Fig. 4, with the respective protein structure is listed along with the corresponding PDB entry. When available, the $\Delta\Delta G$ associated with a mutation and the pH of the experimental setup in which protein stability was measured are provided.

References

1. HM Berman, et al., The Protein Data Bank. *Nucleic Acids Res*. **28**, 235–242 (2000).
2. S Chaudhury, S Lyskov, JJ Gray, PyRosetta: a script-based interface for implementing molecular modeling algorithms using rosetta. *Bioinformatics* **26**, 689–691 (2010).
3. M AlQuraishi, ProteinNet: a standardized data set for machine learning of protein structure. *BMC Bioinforma*. **20**, 311 (2019).
4. J Dauparas, et al., Robust deep learning-based protein sequence design using ProteinMPNN. *Science* **378**, 49–56 (2022).
5. LH Weaver, BW Matthews, Structure of bacteriophage T4 lysozyme refined at 1.7 å resolution. *J. Mol. Biol*. **193**, 189–199 (1987).
6. J Stourac, et al., FireProtDB: database of manually curated protein stability data. *Nucleic Acids Res*. **49**, D319–D324 (2021).
7. J Jumper, et al., Highly accurate protein structure prediction with AlphaFold. *Nature* **596**, 583–589 (2021).
8. TN Starr, et al., Shifting mutational constraints in the SARS-CoV-2 receptor-binding domain during viral evolution. *Science* **377**, 420–424 (2022).
9. J Lan, et al., Structure of the SARS-CoV-2 spike receptor-binding domain bound to the ACE2 receptor. *Nature* **581**, 215–220 (2020) Number: 7807 Publisher: Nature Publishing Group.
10. Schrödinger, LLC, The PyMOL Molecular Graphics System, Version 1.8. (2015).
11. T Gallagher, P Alexander, P Bryan, GL Gilliland, Two Crystal Structures of the B1 Immunoglobulin-Binding Domain of Streptococcal Protein G and Comparison with NMR. *Biochemistry* **33** (1994).
12. S Batzner, et al., E(3)-Equivariant Graph Neural Networks for Data-Efficient and Accurate Interatomic Potentials. *Nat. Commun*. **13** (2022).
13. WK Tung, *Group theory in physics*. (World Scientific, Philadelphia), (1985).
14. R Kondor, Z Lin, S Trivedi, Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. *Proc. 32nd Int. Conf. on Neural Inf. Process. Syst. (NIPS 18)* pp. 10138–10147 (2018).
15. N Thomas, et al., Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv* **1802.08219** (2018).
16. A Musaelian, et al., Learning local equivariant representations for large-scale atomistic dynamics. *Nat. Commun.* **14**, 1–15 (2023).
17. F Chollet, others, Keras (2015) Retrieved from https://github.com/fchollet/keras.
18. DP Kingma, J Ba, Adam: A Method for Stochastic Optimization in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, eds. Y Bengio, Y LeCun. (2015).
19. W Torng, RB Altman, 3D deep convolutional neural networks for amino acid environment similarity analysis. *BMC Bioinforma*. **18**, 302 (2017).
20. R Shroff, et al., Discovery of novel Gain-of-Function mutations guided by Structure-Based deep learning. *ACS Synth. Biol.* **9**, 2927–2935 (2020).
21. W Boomsma, J Frellsen, Spherical convolutions and their application in molecular modelling in *Advances in Neural Information Processing Systems*. (Curran Associates, Inc.), Vol. 30, (2017).
22. M Weiler, M Geiger, M Welling, W Boomsma, T Cohen, 3D steerable CNNs: learning rotationally equivariant features in volumetric data in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18. (Curran Associates Inc., Red Hook, NY, USA), pp. 10402–10413 (2018).
23. Y Zhang, J Skolnick, Scoring function for automated assessment of protein structure template quality. *Proteins* **57**, 702–710 (2004).
24. J Xu, Y Zhang, How significant is a protein structure similarity with tm-score= 0.5? *Bioinformatics* **226**, 889–895 (2010).
25. R Sheridan, et al., Evfold.org: Evolutionary couplings and protein 3d structure prediction. *bioRxiv* (2015).
26. M Weigt, RA White, H Szurmant, JA Hoch, T Hwa, Identification of direct residue contacts in protein–protein interaction by message passing. *Proc. Natl. Acad. Sci*. **106** (2009).
27. BE Suzek, H Huang, P McGarvey, R Mazumder, CH Wu, UniRef: comprehensive and non-redundant UniProt reference clusters. *Bioinformatics* **23**, 1282–1288 (2007).
28. LM Blaabjerg, et al., Rapid protein stability prediction using deep learning representations. *Elife* **12** (2023).
29. M Matsumura, WJ Becktel, BW Matthews, Hydrophobic stabilization in T4 lysozyme determined directly by multiple substitutions of ile 3. *Nature* **334**, 406–410 (1988).
30. MM Dixon, H Nicholson, L Shewchuk, WA Baase, BW Matthews, Structure of a hinge-bending bacteriophage T4 lysozyme mutant, Ile3$\rightarrow$ Pro. *J. Mol. Biol.* **227**, 917–933 (1992).
31. NC Gassner, et al., Methionine and alanine substitutions show that the formation of wild-type-like structure in the carboxy-terminal domain of T4 lysozyme is a rate-limiting step in folding. *Biochemistry* **38**, 14451–14460 (1999).
32. H Nicholson, DE Anderson, S Dao-pin, BW Matthews, Analysis of the interaction between charged side chains and the alpha-helix dipole using designed thermostable mutants of phage T4 lysozyme. *Biochemistry* **30**, 9816–9828 (1991).
33. LH Weaver, et al., High-resolution structure of the temperature-sensitive mutant of phage lysozyme, arg 96—-his. *Biochemistry* **28**, 3793–3797 (1989).
34. BHM Mooers, WA Baase, JW Wray, BW Matthews, Contributions of all 20 amino acids at site 96 to the stability and structure of T4 lysozyme. *Protein Sci.* **18**, 871–880 (2009).
35. JW Wray, et al., Structural analysis of a non-contiguous second-site revertant in T4 lysozyme shows that increasing the rigidity of a protein can enhance its stability. *J. Mol. Biol.* **292**, 1111–1120 (1999).
36. S Dao-pin, DE Anderson, WA Baase, FW Dahlquist, BW Matthews, Structural and thermodynamic consequences of burying a charged residue within the hydrophobic core of T4 lysozyme. *Biochemistry* **30**, 11521–11529 (1991).
37. JH Hurley, WA Baase, BW Matthews, Design and structural analysis of alternative hydrophobic core packing arrangements in bacterio-

phage T4 lysozyme. *J. Mol. Biol.* **224**, 1143–1159 (1992).

38. LA Lipscomb, et al., Context-dependent protein stabilization by methionine-to-leucine substitution shown in T4 lysozyme. *Protein Sci.* **7**, 765–773 (1998).
39. J Xu, WA Baase, ML Quillin, EP Baldwin, BW Matthews, Structural and thermodynamic analysis of the binding of solvent at internal sites in T4 lysozyme. *Protein Sci.* **10**, 1067–1078 (2001).
40. TM Gray, BW Matthews, Structural analysis of the temperature-sensitive mutant of bacteriophage T4 lysozyme, glycine 156→aspartic acid. *J. Biol. Chem.* **262**, 16858–16864 (1987).
41. MG Grütter, TM Gray, LH Weaver, TA Wilson, BW Matthews, Structural studies of mutants of the lysozyme of bacteriophage t4. the temperature-sensitive mutant protein Thr157—-Ile. *J. Mol. Biol.* **197**, 315–329 (1987).
42. P Pjura, BW Matthews, Structures of randomly generated mutants of T4 lysozyme show that protein stability can be enhanced by relaxation of strain and by improved hydrogen bonding via bound solvent. *Protein Sci.* **2**, 2226–2232 (1993).
43. H Nicholson, WJ Becktel, BW Matthews, Enhanced protein thermostability from designed mutations that interact with alpha-helix dipoles. *Nature* **336**, 651–656 (1988).
44. BW Matthews, H Nicholson, WJ Becktel, Enhanced protein thermostability from site-directed mutations that decrease the entropy of unfolding. *Proc. Natl. Acad. Sci. United States Am.* **84**, 6663–6667 (1987).
45. BHM Mooers, et al., Repacking the core of T4 lysozyme by automated design. *J. Mol. Biol.* **332**, 741–756 (2003).
46. DE Anderson, JH Hurley, H Nicholson, WA Baase, BW Matthews, Hydrophobic core repacking and aromatic-aromatic interaction in the thermostable mutant of T4 lysozyme Ser 117 → Phe. *Protein Sci.* **2**, 1285–1290 (1993).