Real-valued Prediction of Side-chain Dihedrals in Proteins With Relation-Shape Convolution

Xiyao Long
College of Computing and Informatics
Drexel University
Fox Chase Cancer Center
Philadelphia, PA 19111
Xiyao.Long@fccc.edu

Maxim Shapovalov
Fox Chase Cancer Center
Philadelphia, PA 19111
Maxim.Shapovalov@fccc.edu

Roland Dunbrack
Fox Chase Cancer Center
Philadelphia, PA 19111
roland.dunbrack@gmail.com

Abstract

Modeling of protein side-chain conformations is a long-standing subproblem in protein structure prediction. It helps to refine experimental structures with poor resolution, and is used for sampling side chains in computational protein design. Related studies date back to the 1980s starting from statistically analyzing side-chain conformations, developing energy functions, and implementing algorithms for decomposing the side-chain interaction graph as subgraphs such as in SCWRL4. Here we employ a geometric deep-learning method Relation-Shape Convolution(RSConv) originally applied to point clouds, to the side-chain problem. With features consisting of the backbone atom Cartesian coordinates (in a local frame), backbone dihedral angles, and residues types of neighbors, we achieve a favorable testing set accuracy of the chi1 dihedral angle of 89% and chi2 accuracy of 83% given correct chi1 angles on our test set. Our prediction accuracy strongly correlates with the experimental atomic displacement B-factors of the side chains. The chi1 dihedrals with B-factor less than 30° representing about 53% of all side chains in our dataset have prediction accuracy of 93%. The 93% rate is comparable to the chi1 accuracy in AlphaFold2 when it achieves high backbone structure recovery (100 IDDT C_α).

1 Introduction and related work

1.1 Past work on side-chain modeling

Correct packing of protein sidechains is crucial for both protein-structure prediction and refinement of experimentally determined protein structures. Side-chain degrees of freedom expressed in dihedral angles mostly fall into three clusters referred to as the g-(-60), t(180), and g+(+60) chi rotamers. Their frequencies, angles means and variances are reported in “rotamer libraries". [1, 2, 3] Rotamer frequencies and mean angles can be backbone conformation independent or dependent. The latter are constructed over a grid of backbone phi and psi angles in 10 degrees bins. [4, 5, 6] Backbone-dependent rotamer libraries result in more accurate side-chain conformation prediction and more

Machine Learning for Structural Biology Workshop, NeurIPS 2021.

efficient protein design. These libraries can be estimated accurately with adaptive kernel density estimates and regression [7, 8]. Protein side-chain conformation prediction proceeds by sampling conformations from a rotamer library and a search algorithm over the available conformations utilizing a scoring or energy function. For instance, the SCWRL4 algorithm identifies impossible combinations of neighbor conformations with a collision detection method and finds the global minimum of its energy function with a graph tree decomposition algorithm. Methods such as SCWRL4 and other similar strategies [9, 10, 11, 12, 8] are limited by discrete rotamer sampling and the ability of the energy function to identify the native structure.

More recent deep-learning based methods [13, 14, 15] which directly predict protein side-chain conformations conditioned on their molecular environments, can model conformations to fine granularity and substantially reduce the modeling time required for sampling. It enables representing the complex geometric environment as learned by neural networks. This is different compared to Monte-Carlo based ROSETTA or molecular dynamics with explicitly learned energy [16, 17, 3] with one-body, two-body, three-body energies [18]. The neural network approach immediately projects the neighboring geometric information to a representation that can capture complex relations.

1.2 Geometric deep-learning on point cloud and protein structure data

Geometric deep learning has been applied to point clouds for protein data and other domains [19, 20, 21, 22, 23]. Graph convolutional networks have been used for geometric learning problems, where node features are generally properties associated with a node and edge features describing relative positions between two points. Convolution operations can enable interactions between node and edge features and learn over a neighborhood in the graph. Several methods that accomplish this have been described. In dynamic graph CNNs(DGCNN) [21], an edgeConv operation is a function of the edge $x_i - x_j$ as relative coordinates and features of the graph neighbors of node i. A permutation invariant aggregation function or max pooling applies to the neighbors of i, expressed as $\max_{j \in N(i)} \left\{ f^{'}\left(x_i - x_j\right) + f(x_i) \right\}$ where N_i denotes the neighborhood of i. In the Crystal GNN [23], a more complex neighborhood function is used which updates node representation v_i from the previous iteration as: $v_i + \sum_{j \in N(i)} \left\{ F\left([v_i \parallel v_j \parallel e_{i,j}]\right) \odot f([v_i \parallel v_j \parallel e_{i,j}]) \right\}$ where $\odot$ denotes channel-wise multiplication. Channel-wise multiplication is applied between the outputs of F and outputs of f before symmetric aggregation. In Edge-Conditioned-Convolution(ECC) [20], the neighborhood function $\sum_{j \in N(i)} \left\{ F\left(e_{i,j}\right) \odot f(x_i) \right\}$ has $F\left(e_{i,j}\right)$ generating weights that are specifically conditioned on $e_{i,j}$ and multiplies them with $f(x_i)$. In relation-shape convolution(RS-Conv) [22], the neighborhood is learned as $f_{N(x_i)} = \mathcal{A}_{j:\in N(i)} \{ M\left(e_{i,j}\right) \odot f\left(x_i\right) \}$, where $\mathcal{A}$ is a symmetric aggregation function. This is similar to ECC but with more choices for edge features $e_{i,j}$ and neighborhood selection $N(x_i)$.

In the side-chain prediction problem, we start with an input backbone atom coordinates and amino acid types of each residue in the sequence. Neighboring residues in the sequence are connected by backbone dihedral angles that can be computed from the atomic coordinates. Two residues spatially close to each other are governed by van der Waals and electrostatic interactions. Learning to aggregate local geometric relations may lead to an efficient local protein structure learning function. In this paper, we define specific node and edge features, utilize RS-Conv to learn geometric relation-dependent weights applied to the node features, and build an effective architecture to learn a neighborhood representation of each side-chain for their conformation prediction.

2 Notation and featurization

In order to mimic the experimental data closely, from the asymmetric unit (ASU) of the protein crystal, we build all neighboring proteins in the crystal that contact the ASU. The residues in a constructed crystal are divided into two sets: one set consists of residues in the chain(s) we are interested in ($\mathcal{V}_{query}$), while the other set consists of all residues in the structure($\mathcal{V}_{all}$). For each residue in $\mathcal{V}_{query}$, edges are built between the residue and its neighboring residues in $\mathcal{V}_{all}$. The neighborhood consists of either the 12 nearest residues or those falling within 10 Å $C\alpha - C\alpha$ distances. The constructed graph $\mathcal{G} = (\mathcal{V}_{all}, \mathcal{E})$ with a directional-edge set $\mathcal{E}$ records the neighboring residues for each query residue. The residues in $\mathcal{V}_{all}$ are considered as nodes with node features $X \in \mathbb{R}^{N_{\mathcal{V}_{all}} \times F}$ and edges in $\mathcal{E}$ have edge features $E \in \mathbb{R}^{N_{\mathcal{E}} \times S}$. The node features x_i include amino acid types and the $\sin$ and

cos of the backbone φ and ψ angles. The edge features $e_{i,j}$ are angular orientations and distances of neighboring residue pairs, one-hot encoding of whether the residues are immediate neighbors of the query residue along the protein sequence, and neighboring residue backbone coordinates derived from the backbone reference frame in the sense that the query residue backbone coordinates always orient in the same direction with $C\alpha_q$ at the origin(more details in Sup 6.1). We represent each side-chain dihedral as (sin, cos) pair and enforce the continuous dihedral prediction by the network. Structure correction and side-chain filtering are noted in Sup 6.2.

3 Highlights of the architecture and feature incorporation:

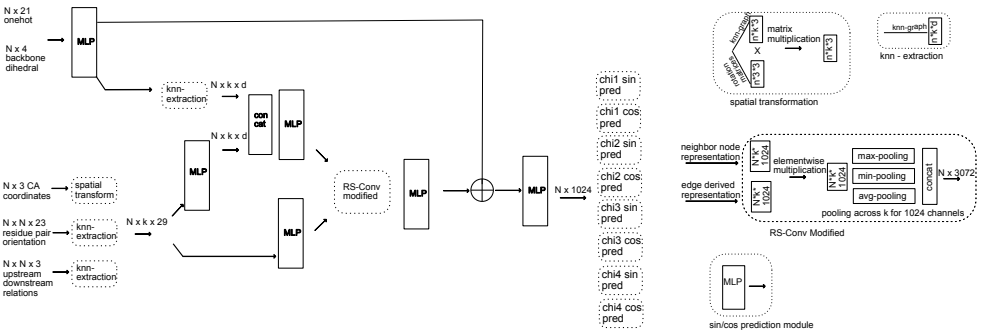


Figure 1: Schematics of feature extraction and network architecture.

The neighborhood learning operation around each query residue is based on RSConv [22] network and implemented as shown as Figure 1. Residues are abstracted as nodes with features x_i, and any pair of residues connected in graph $\mathcal{G} = (\mathcal{V}_{all}, \mathcal{E})$) are referred as edges with edge features $e_{i,j}$ being the geometric descriptors between the residues as distances and orientations. The neighborhood structure of a node i is learned from $f_{N(x_i)} = \mathcal{A}_{j \in N(i)}\{M\left(e_{i,j}\right) \odot f\left([x_j \parallel e_{i,j}]\right)\}$ where $[x_j \parallel e_{i,j}]$ is the concatenation of the node and edge features for node j in the neighborhood of node i. The f takes in the concatenated features and outputs what can be perceived as node representations. The M maps the edge features to a higher-dimension to be multiplied with the output of f. The $\mathcal{A}$ operation can be any symmetric aggregation function to achieve permutation invariance of the neighboring residues. In $e_{i,j}$, the residue-pair dihedrals, angles, distances, one hot features which are all invariant to the rigid rotation of the Cartesian coordinates. The Cartesian coordinates of neighbors in the local frame are derived from the local backbone frame of the query residue and do not change with rigid protein rotation either. In our basic implementation, a concatenation of max, min, average mixed pooling is used to extract richer information [24]. Comparing to previous models, our implementation "RSConv-basic" has the following items: (1) node and edge features $[x_i \parallel e_{i,j}]$ are input to f, whereas RS-Conv takes the form $M\left(e_{i,j}\right) \odot f(x_i)$; (2) we tested both k-nearest-neighbors with k=12 similar to DGCNN [21] and the spherical neighborhood as in XENet [19] with a radial threshold of 10Å; (3) the representation of the query residue and its neighborhood are combined by adding f_{0,x_i} with the $f_{N(x_i)}$ as $f_{1,x_i} = f_{0,x_i} + f_{N(x_i)}$, and we tried both only one message passing(MP) iteration and two rounds of MP in our experiments. (4) inclusion of atom coordinates $X_{rotatedcoordinates}$ from the set of neighborhood residues derived in the fixed backbone frame of the query residue, which is not explored in DGCNN, RS-Conv, or XENet.

4 Experiments and Results:

We first implemented and trained the basic model and tested several variations of it (Table 1). Our basic model "**RSConv-basic**" has convolution in the form of $\mathcal{A}_{j \in N_i}\{M\left(e_{i,j}\right) \odot f\left([x_j \parallel e_{i,j}]\right)\}$ as introduced in the previous section, with neighborhood N_i being k-nearest neighbors with k equal to 12, and concatenates max, min, avg pooling adding up to a depth of 3072. We tested different variations of the model architecture and feature combinations, measured the percentage of predicted chi1 and chi2 within 40° of the ground truth values, for chi2 we only included the ones with the correctly

predicted chi1(because correct prediction of chi2 is irrelevant if chi1 is wrong). Our experiment indicates that two variations have the most important impact on the prediction performance. First, if we use only $f([x_i])$ instead of $f([x_i \parallel e_{i,j}])$ without the concatenation as in "**RSConv-f** $([\mathbf{x_i}])$**"**, the chi1 accuracy drops by 2%. Not using transformed C_α coordinates in the edge feature as in model "**RSConv-**$(\mathbf{e_{i,j}}$: $\mathbf{nocartesian})$" does not decrease prediction accuracy much. Second, inclusion of all four backbone atom coordinates as in "**RSConv-**$(\mathbf{e_{i,j}} : \mathbf{X_{i,j}})$ $\&\mathbf{10}$Å" increases the prediction performance by 0.9% for chi1 and 2.2% for chi2 given chi1 is correct and leads to the highest accuracy among all variants.

Table 1: Benchmark on test set using different network architectures and feature combinations. $X_{i,j}$ are the four $N, C\alpha, C, O$ backbone coordinates. All accuracies are based on 40° difference criterion.

Model	MP iteration #	Test accuracy		dataset
		chi1	chi2 given chi1 correct	
RSConv- $(e_{i,j} : X_{i,j})$& 10Å	1	**0.886**	**0.826**	40% less identity
RSConv-basic	1	0.877	0.804	40% less identity
RSConv-basic	1	0.868	0.817	Ecod - split
RSConv-two-MP-iter	2	0.875	0.818	40% less identity
RSConv-Maxpooling	1	0.871	0.812	40% less identity
RSConv-$f([x_i])$	1	0.857	0.802	40% less identity
RSConv- $(e_{i,j} : nocartesian)$	1	0.871	0.807	40% less identity

Comparing to one message passing in "**RSConv-basic**", implementation of the model with two rounds of message passing denoted as "**RSConv-two-MP-iter**" shows some increase in chi2 given chi1 correct accuracy but similar chi1 accuracy. We speculate that the correct packing is mostly determined by the immediate neighbor residues and the more distant residues do not bring any significant benefit. Further, the test accuracies with more rigorous dataset splitting technique "Ecod-split" are a little lower but within 1% for chi1. In this dataset we split training/testing/validation based on Ecod[25] homology groups to guarantee low homology between sequences within the same split, and the similar results suggests that previous model accuracies are not observed due to possible homologies between training and testing sets. For "**RSConv-Maxpooling**" only using max pooling, the chi1 and chi2 accuracies are similar to the ones from "**RSConv-basic**"model; it suggests that mixed pooling is not essential for getting chi1 accuracy above 87%.

In the test result of the best performing model "**RSConv-**$(\mathbf{e_{i,j}} : \mathbf{X_{i,j}})$ $\&\mathbf{10}$Å", the predicted dihedral angle distribution overlaps with the ground-truth distribution as shown in Suppl. Figure 1 and 2, including some low-density regions as shown in Suppl. Figure 2. It is apparent that the prediction accuracies are dependent on the relevant atomic displacement B-factor as shown in Suppl. Figure 3. Different residue types have different prediction accuracies, and how steeply the prediction accuracies decrease as B-factor values increase also varies among different residue types. This is likely due to the fact that some polar protein side chains sample multiple rotameric conformations within a single protein crystal. Many of those with high B-factors are likely multi-conformational, while the deposited coordinates may reflect only one possible conformation [26, 27, 28, 29]. This may contribute to the prediction accuracies that we observe: the current single value prediction model is incomplete and the mispredictions may correspond to flexible parts of the structure.

5 Conclusion:

Side-chain conformation prediction typically depends on some form of "packing" algorithm, where side chain rotamers are sampled to minimize clashes between them. In contrast, we find that a deep-learning algorithm can perform as well or better than packing algorithms with features based on residue type, relative orientations, and backbone dihedral angles of each residue and its neighbors. We extract residue level node features and between residues edge features to learn the local neighborhood chemical environment of any residue through convolution operations originally applied to pointclouds. We find that concatenating backbone dihedrals as node features and relations between residues as edge features most obviously improves the prediction performance. We find that including the backbone

atomic coordinates of neighboring residues in the fixed reference frame defined by the query residue also increases the model performance. So far in our experiments, a spherical neighborhood is comparable to using the 12 nearest neighbors. A future direction is to find side chains which are particularly affected by more distant residue conformations and further study an effective architecture to learn those cases. Furthermore, a probabilistic framework may be necessary to account for the flexible side chain conformations.

References

[1] Lovell S C et al. "The penultimate rotamer library". In: *Proteins* 40 (3 Aug. 2000), pp. 389–408.

[2] Bhat T, Sasisekharan V, and Vijayan M. "An analysis of sidechain conformation in proteins". In: *International Journal of Peptide and Protein Research* 13 (2 1979), pp. 170–184. DOI: 10.1111/J.1399-3011.1979.TB01866.X.

[3] Dunbrack R. "Rotamer libraries in the 21st century". In: *Current Opinion in Structural Biology* 12 (4 Aug. 2002), pp. 431–440. DOI: 10.1016/S0959-440X(02)00344-5.

[4] Dunbrack R and Karplus M. "Backbone-dependent rotamer library for proteins. Application to side-chain prediction". In: *Journal of molecular biology* 230 (2 Mar. 1993), pp. 543–574. DOI: 10.1006/JMBI.1993.1170.

[5] Shapovalov M and Dunbrack R. "A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions". In: *Structure* 19 (6 June 2011), p. 844. DOI: 10.1016/J.STR.2011.03.019.

[6] Dunbrack R and Cohen F. "Bayesian statistical analysis of protein side-chain rotamer preferences". In: *Protein Science* 6 (8 Aug. 1997), pp. 1661–1681. DOI: 10.1002/pro.5560060807.

[7] Ting D et al. "Neighbordependent ramachandran probability distributions of amino acids developed from a hierarchical dirichlet process model". In: *PLoS Computational Biology* 6 (4 Apr. 2010), p. 1000763. DOI: 10.1371/journal.pcbi.1000763.

[8] Krivov G, Shapovalov M, and Dunbrack R. "Improved prediction of protein side-chain conformations with SCWRL4". In: *Proteins: Structure, Function and Bioinformatics* 77 (4 2009), pp. 778–795. DOI: 10.1002/prot.22488.

[9] Xu G et al. "OPUS-Rota2: An improved fast and accurate sidechain modeling method". In: *Journal of Chemical Theory and Computation* 15 (9 Sept. 2019), pp. 5154–5160. DOI: 10.1021/ACS.JCTC.9B00309.

[10] Goldstein R F. "Efficient rotamer elimination applied to protein side-chains and related spin glasses". In: *Biophysical journal* 66 (5 1994), pp. 1335–1340. DOI: 10.1016/S0006-3495(94)80923-3.

[11] Shetty R et al. "Advantages of finegrained side chain conformer libraries". In: *Protein Engineering, Design and Selection* 16 (12 Dec. 2003), pp. 963–969. DOI: 10.1093/PROTEIN/GZG143.

[12] Desmet J et al. "The dead-end elimination theorem and its use in protein sidechain positioning". In: *Nature* 356 (6369 1992), pp. 539–542. DOI: 10.1038/356539A0.

[13] Yang J et al. "Improved protein structure prediction using predicted interresidue orientations". In: *Proceedings of the National Academy of Sciences of the United States of America* 117 (3 Jan. 2020), pp. 1496–1503. DOI: 10.1073/pnas.1914677117.

[14] Anand N et al. "Protein Sequence Design with a Learned Potential". In: (Mar. 2020). DOI: 10.1101/2020.01.06.895466.

[15] Akpinaroglu D et al. "Improved antibody structure prediction by deep learning of side chain conformations". In: *bioRxiv* (Sept. 2021), p. 2021. DOI: 10.1101/2021.09.22.461349.

[16] Lamb M and Jorgensen W. "Computational approaches to molecular recognition". In: *Current Opinion in Chemical Biology* 1 (4 Dec. 1997), pp. 449–457. DOI: 10.1016/S1367-5931(97)80038-5.

[17] Karplus M and Mccammon J. "Molecular dynamics simulations of biomolecules". In: *Nature Structural Biology* 9 (9 2002), pp. 646–652. DOI: 10.1038/nsb0902-646.

[18] Leman J et al. “Macromolecular modeling and design in Rosetta: recent methods and frameworks”. In: *Nature Research* 17 (7 Jan. 2020), pp. 665–680. DOI: 10.1038/s41592-020-0848-2.

[19] Maguire J et al. “XENet: Using a new graph convolution to accelerate the timeline for protein design on quantum computers”. In: *bioRxiv* (May 2021). DOI: 10.1101/2021.05.05.442729.

[20] Simonovsky M and Komodakis N. “Dynamic edge-conditioned filters in convolutional neural networks on graphs”. In: *Proceedings -30th IEEE Conference on Computer Vision and Pattern Recognition* (Jan. 2017), pp. 29–38.

[21] Wang Y et al. “Dynamic graph CNN for learning on point clouds”. In: *ACM Transactions on Graphics* 38 (5 Jan. 2018), p. 13.

[22] Liu Y et al. “Relation-shape convolutional neural network for point cloud analysis”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (June 2019). DOI: 10.1109/CVPR.2019.00910.

[23] Xie T and Grossman J. “Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties”. In: *Physical Review Letters* 120 (14 Oct. 2017). DOI: 10.1103/PhysRevLett.120.145301.

[24] Yu D et al. “Mixed pooling for convolutional neural networks”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics* 8818 (Oct. 2014), pp. 364–375. DOI: 10.1007/978-3-319-11740-9_34.

[25] Cheng H. “ECOD: An evolutionary classification of protein domains”. In: *PLoS Computational Biology* 10 (12 Dec. 2014), p. 1003926. DOI: 10.1371/journal.pcbi.1003926.

[26] Miao Z and Cao Y. “Quantifying side-chain conformational variations in protein structure”. In: *Scientific Reports* 6 (1 Nov. 2016), pp. 1–10. DOI: 10.1038/srep37024.

[27] Zavodszky M. “Side-chain flexibility in protein-ligand binding: The minimal rotation hypothesis”. In: *Protein Science* 14 (4 Mar. 2005), pp. 1104–1114. DOI: 10.1110/ps.041153605.

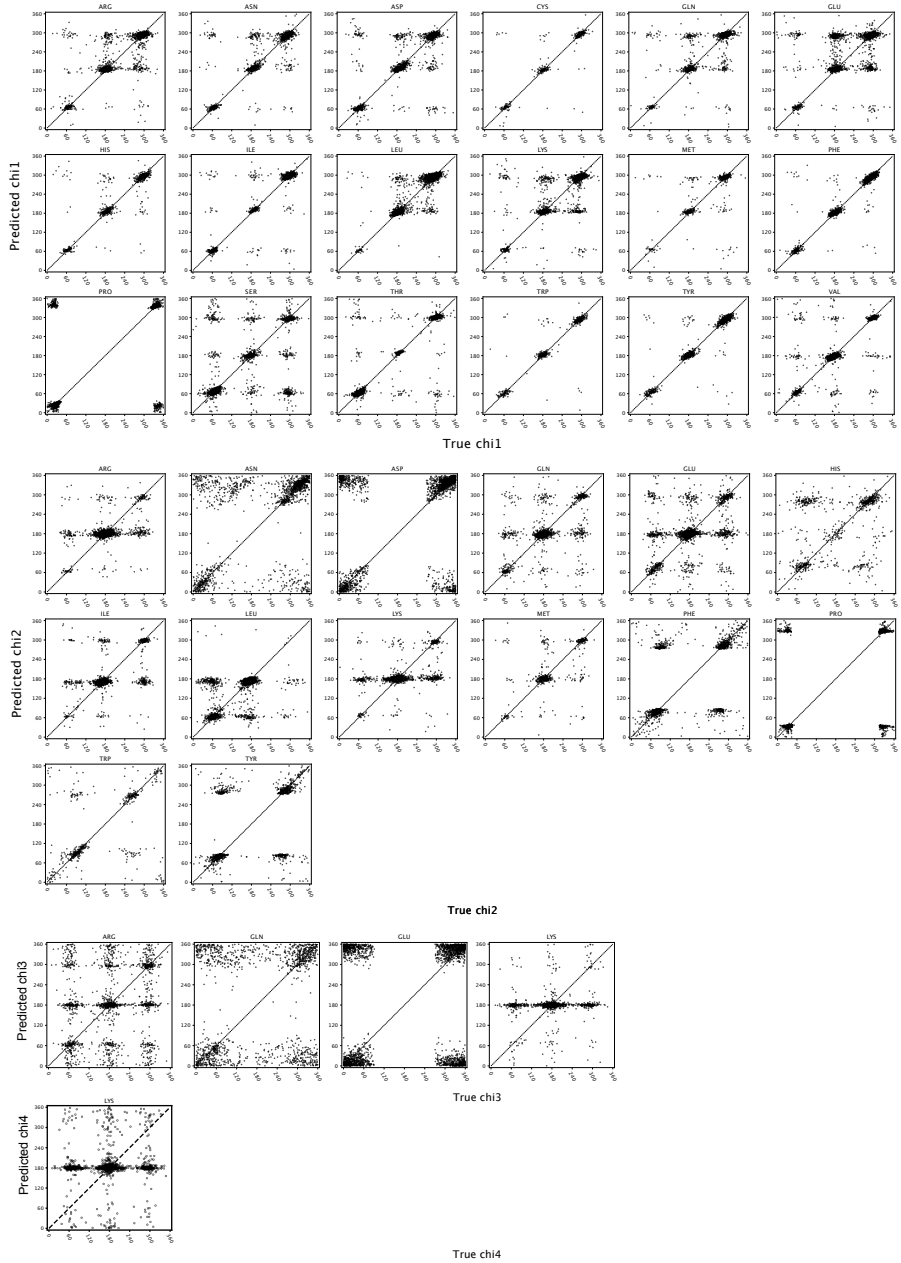
[28] Shalit Y and Tuvi-Arad I. “Side chain flexibility and the symmetry of protein homodimers”. In: *PLoS ONE* 15 (7 July 2020). DOI: 10.1371/journal.pone.0235863.

[29] Ruvinsky A et al. “Side-chain conformational changes upon protein-protein association”. In: *Journal of Molecular Biology* 408 (2 Apr. 2011), pp. 356–365. DOI: 10.1016/j.jmb.2011.02.030.

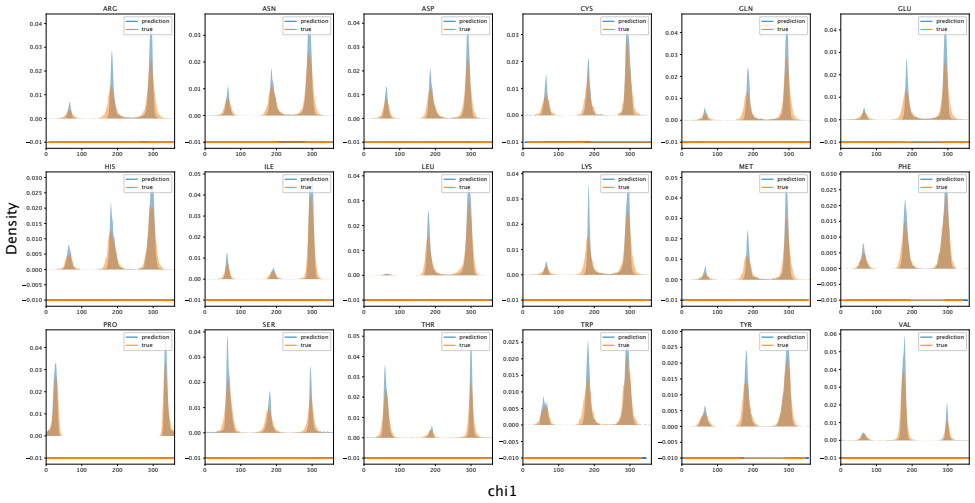
[30] Jumper J et al. “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* 2021 (July 2021), pp. 1–11. DOI: 10.1038/s41586-021-03819-2.

[31] Word J et al. “Asparagine and glutamine: Using hydrogen atom contacts in the choice of side-chain amide orientation”. In: *Journal of Molecular Biology* 285 (4 Jan. 1999), pp. 1735–1747. DOI: 10.1006/jmbi.1998.2401.

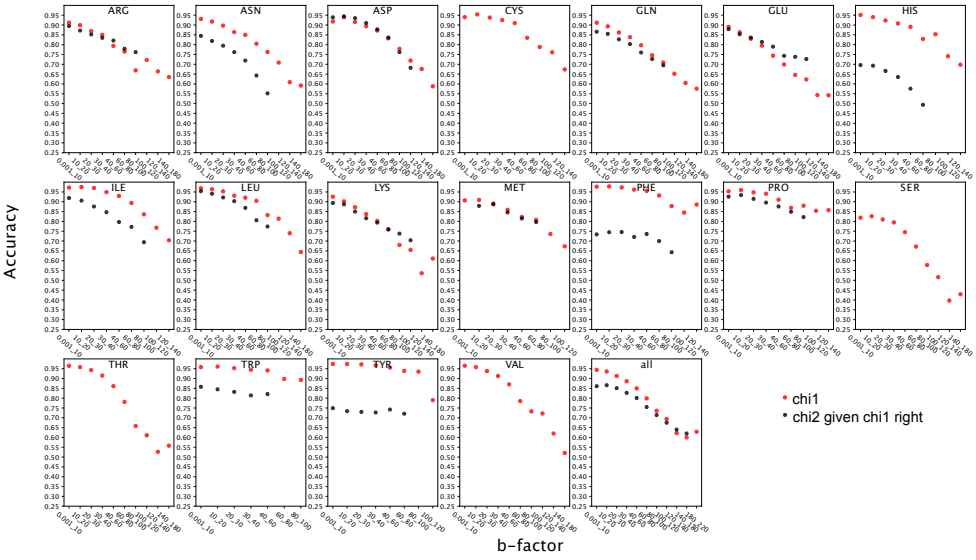
6 Supplementary results:



Suppl Figure 1: The true vs predicted chi1 dihedrals.
The model learned the finer details as the predictions positively correlate with the true dihedrals within each residue type and rotamer class in many cases – most obviously in TYR, PHE, PRO.



Suppl Figure 2: Predicted and true chi dihedrals density map overlap.
The x-axis corresponds to the chi1 dihedrals and y-axis indicates densities. The predicted dihedrals are more concentrated compared to the experimental values. The overall data distribution are successfully learned by the model.



Suppl Figure 3: Chi dihedrals prediction accuracy by b-factor interval.
The y-axis in each plot corresponds to prediction accuracies and the x-axis is the B-factor interval; chi1 accuracy and the chi2 accuracy given correct chi1 prediction are shown in red and black respectively. The B-factor is recorded using the fourth atom of the atoms involved in defining the corresponding dihedral. The prediction accuracies all tend to decrease with higher B-factor intervals. Some residues have a steeper decreases with increasing B-factor such as in ARG, GLU and LYS. SER has low prediction accuracy compared to other residues even at low b-factor interval. TRP, TYR and PHE all have high prediction accuracies and are relatively unaffected by the increase in b-factor values.

6.1 Features:

1. Edge features: $e_{i,j} = [e_{i,j}^{residue-pair-geometric} || e_{i,j}^{positional-encoding} || e_{i,j}^{rotated-coordinates}]$
   (a) Pairwise features between any two residues in a protein:

For each pair of residues, a vector of length 23 :

- Backbone dihedrals:
  - $C_{n-1} - N - C\alpha - C(res1)$
  - $N - C\alpha - C - N_{n+1}(res1)$
  - $C_{n-1} - N - C\alpha - C(res2)$
  - $N - C\alpha - C - N_{n+1}(res2)$
- Angles:
  - $N_1 - C\alpha_1 - C\alpha_2$
  - $C\alpha_1 - C\alpha_2 - N_2$
  - $C\alpha_2 - N_2 - C_2$
- Residue-pair Dihedrals:
  - $C_1 - N_1 - C\alpha_1 - C\alpha_2$
  - $N_1 - C\alpha_1 - C\alpha_2 - N_2$
  - $C\alpha_1 - C\alpha_2 - N_2 - C_2$
- Distances:
  - $C\alpha_1 - C\alpha_2$
  - $N_2 - C_2$
  - $C\alpha_2 - N_2$

In total there are ten angular values coded into sine and cosine values, make it a vector of length 20. Adding in 3 distance values it becomes a vector of length 23. This vector is used as the edge features between any two residues although not all elements are describing pairwise relation. The angle $C\alpha_2 - N_2 - C_2$ and bond lengths $N_2 - C_2, C\alpha_2 - N_2$ are included because they are needed for rebuilding the 2nd residue backbone coordinates given the first residue. The backbone dihedrals are included as they imply the side-chain orientations(e.g., as in the backbone-dependent rotamer library) and are needed to fully describe two residues orientations in space.

(b) The relative position of a residue in the protein sequence relative to another residue:

- Positional coding: If it is direct upstream in the sequence, it is [1,0,0]; if direct downstream in sequence, it is [0,0,1], if neither [0,1,0]

(c) Pairwise features computed for a set of neighboring residues w.r.t. their query residue:

- Transformed (x, y, z) coordinates of the $N, C\alpha, C, O$ atoms of the 12 neighbor residues of any query residue.

i. A translation vector and a rotation matrix is computed for each query residue, so that after applying the translation and rotation to the query residue coordinates in the pdb file, the $C\alpha$ atom of query residue is at (0,0,0), the coordinate vector $\overrightarrow{C\alpha - C}$ points to the direction of $\overrightarrow{(1,1,1)}$ and the $N, C\alpha, C$ atom coordinates lie on the plane defined by points (0,0,0), (1,1,1), (1,1,0) in the Cartesian space.

ii. The rotation matrix is computed using two vectors $\overrightarrow{C\alpha - C}$ and $\overrightarrow{C\alpha - N}$. A rotation matrix needed for rotating vector $\overrightarrow{C_\alpha - C}$ to align with the vector $\overrightarrow{(1,1,1)}$ is computed. The rotation matrix is then applied to $C\alpha - N$. Another rotation matrix needed for rotating the rotated vector $\overrightarrow{C\alpha - N}\prime$ around an axis defined by $\overrightarrow{(1,1,1)}$ to the plane defined by coordinates $(0,0,0)$, $(1,1,1)$, $(1,1,0)$ is calculated. The second computed rotation matrix is multiplied with the first rotation matrix and results in rotation matrix that can to applied to the sets are first translated query backbone coordinates and neighbors backbone coordinates. The original coordinates of the query and its neighbor residues atom set is first translated by the $C\alpha$ position so that the query $C\alpha$ is at $(0,0,0)$, then they are rotated by the rotation matrix.

iii. The above operation generates neighbor coordinates derived from the local reference. In these frames, the query residue always orients in a specific direction. The features are similar to the backbone frame shown in AlphaFold2 [30], but we use a slightly different definition to construct the frame. The math is described in section 6.3. When applying the rotation and translation to both a query residue and its neighboring residues, the relative coordinates derived from the frame along with other parts of the edge features describe the spatial relations between any pair of a neighbor residue and the query residue.

2. Node features: $x_i = [x_{onehot}||x_{backbonedihedral}]$
   (a) One hot encoded residue identity by a vector of 21.
   (b) Backbone dihedrals coded into sine and cosine values, this corresponds to a vector of length 4 for each residue.
   - $C(n-1) - N - C\alpha - C$
   - $N - C\alpha - C - N(n+1)$

6.2 Data filtering and splitting

The side-chain atoms coordinates are extracted and the corresponding side chain dihedrals are calculated**.** We correct for inconsistencies in atom name assignment for symmetric side chains, such as **Asp, Glu, Tyr and Phe**. We use the REDUCE program [31] to fix the atoms in the terminal dihedral angles of **Asn, Gln, His** based on their analysis of hydrogen bonding interactions. Residues are marked out for having chain breaks if the C atom of the residue and the N atom of the next residue have distances greater than 2Å and the related backbone dihedrals are substituted with -90°. In some past studies, crystal contacts are found to help to recover the chi dihedrals [8] and we reconstruct the crystal units which lies within 5Å of any $C\alpha$ atom in the deposited asymmetric units(ASU) and include neighboring asymmetric units for feature extraction. We use two datasets for our side-chain prediction modeling. The first dataset is a set of PDB chains culled using Pisces with selection criterion of sequence identity below 40%, resolution better than 3 and R value smaller than 0.3. The set of chains are then divided into three sets with 80% of the chains as training cases, 10% as validation cases, 10% as testing cases. We also construct another dataset with data splitting between training/validation/testing based on ECOD homology groups [25]. Groups are divided into each of the three sets rather than the sequences. As proteins of different ECOD groups are mostly evolutionarily unrelated to each other therefore this splitting method can serve to prevent overtraining. Proteins in each ECOD in the same set are further filtered by resolution cutoff of 2.0Å and lower than 80% sequence identity.

6.3 Rotating local coordinates

To extract geometric information from point clouds, experimenting with different geometric priors sometimes leads to better model performance [21]. In our case the points are the neighboring residues of a query residue. We use backbone atoms in the query residue to define a backbone reference frame, then use the residue neighbor coordinates derived in the backbone reference frame as the geometric descriptors. Such backbone reference frame are calculated for each query residue by performing a sequence of translation and rotations of its original coordinates in the pdb file to the predefined atoms positioning.

1. For a query residue, identify the coordinates of $C\alpha, C$ and N atoms, translate these points so that $\mathbf{C}\alpha = (\mathbf{0}, \mathbf{0}, \mathbf{0})$.
2. Get the rotation matrix which rotates $\overrightarrow{(\mathbf{C}\alpha - \mathbf{C})}$ to align with $\overrightarrow{(\mathbf{1}, \mathbf{1}, \mathbf{1})}$
   (a)
   $$\overrightarrow{\mathbf{m}} = \frac{\overrightarrow{(\mathbf{C}\alpha - \mathbf{C})}}{\left|\overrightarrow{(\mathbf{C}\alpha - \mathbf{C})}\right|}$$
   (b)
   $$\overrightarrow{\mathbf{n}} = \frac{\overrightarrow{(\mathbf{1}, \mathbf{1}, \mathbf{1})}}{\left|\overrightarrow{(\mathbf{1}, \mathbf{1}, \mathbf{1})}\right|}$$
   (c)
   $$\overrightarrow{\mathbf{l}} = \overrightarrow{\mathbf{m}} \times \overrightarrow{\mathbf{n}} = (\mathbf{l_1}, \mathbf{l_2}, \mathbf{l_3})$$
   (d) Take skew:
   $$\mathbf{A} = \begin{bmatrix} \mathbf{0} & -\mathbf{l_3} & \mathbf{l_2} \\ \mathbf{l_3} & \mathbf{0} & -\mathbf{l_1} \\ -\mathbf{l_2} & \mathbf{l_1} & \mathbf{0} \end{bmatrix}$$

(e)

$$\mathbf{p} = \overrightarrow{\mathbf{m}} \cdot \overrightarrow{\mathbf{n}}$$

(f)

$$\mathbf{R} = \mathbf{I_3} + \mathbf{A} + \mathbf{A} \cdot \mathbf{A} \cdot \left(\frac{\mathbf{1}}{\mathbf{1} + \mathbf{p}}\right)$$

3. Apply the rotation matrix R to $\overrightarrow{\mathbf{C}\alpha - \mathbf{N}}$:

$$\overrightarrow{\mathbf{q}} = \mathbf{R} \circ \overrightarrow{\mathbf{C}\alpha - \mathbf{N}}$$

4. Get the rotation matrix which rotates $\overrightarrow{\mathbf{q}}$ around the axis $\overrightarrow{\mathbf{n}}$ so that it will be on the same plane as defined by $\overrightarrow{(\mathbf{1}, \mathbf{1}, \mathbf{0})}$ and $\overrightarrow{\mathbf{n}}$,
   (a) Calculate the normal to the plane where $\overrightarrow{\mathbf{q}}$ and $\overrightarrow{\mathbf{n}}$ lies in

$$\overrightarrow{\mathbf{o}} = \overrightarrow{\mathbf{n}} \times \overrightarrow{\mathbf{q}}$$

   (b) Calculate the normal to the target plane

$$\overrightarrow{\mathbf{r}} = \overrightarrow{\mathbf{n}} \times \overrightarrow{(\mathbf{1}, \mathbf{1}, \mathbf{0})}$$

   (c) Calculate the angle between the two planes which normals are $\overrightarrow{\mathbf{o}}$ and $\overrightarrow{\mathbf{r}}$.
      i.

$$\mathbf{s} = \overrightarrow{\mathbf{r}} \cdot \overrightarrow{\mathbf{o}}$$

      ii.

$$\mathbf{t} = \overrightarrow{\mathbf{n}} \cdot (\overrightarrow{\mathbf{r}} \times \overrightarrow{\mathbf{o}})$$

      iii.

$$\theta = \mathbf{arctan2}\,(\mathbf{t}, \mathbf{s})$$

   (d) Find orthogonal vectors that extend $\overrightarrow{\mathbf{n}}$, and make an orthonormal basis:
      i.

$$\overrightarrow{\mathbf{u}} = \mathbf{rand} = \overrightarrow{(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3})}$$

      ii.

$$\overrightarrow{\mathbf{v}} = -\left(\overrightarrow{\mathbf{u}} \times \overrightarrow{\mathbf{n}}\right) \cdot \overrightarrow{\mathbf{n}}$$

      iii.

$$\overrightarrow{\mathbf{v}} = \frac{\overrightarrow{\mathbf{v}}}{\overrightarrow{|\mathbf{v}|}}$$

      iv.

$$\overrightarrow{\mathbf{w}} = \overrightarrow{\mathbf{n}} \times \overrightarrow{\mathbf{v}}$$

      v.

$$\mathbf{X} = \begin{bmatrix} \overrightarrow{\mathbf{n}} & \overrightarrow{\mathbf{v}} & \overrightarrow{\mathbf{w}} \end{bmatrix}^{\mathbf{T}}$$

   (e)

$$\mathbf{Y} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \cos\theta & \sin\theta \\ \mathbf{0} & -\sin\theta & \cos\theta \end{bmatrix}$$

   (f)

$$\mathbf{Z} = [\mathbf{XY}]\,\mathbf{X^{-1}}$$

5. Multiply the two rotation matrices to get the overall rotation matrix
   (a)

$$\mathbf{T} = \mathbf{ZR}$$