On overcoming PDB data scarcity with AlphaFoldDB for protein side-chain packing

Sriniketh Vangaru
Department of Computer Science
Virginia Tech
Blacksburg, VA, 24060
sriv04@vt.edu

Debswapna Bhattacharya
Department of Computer Science
Virginia Tech
Blacksburg, VA, 24060
dbhattacharya@vt.edu

Abstract

Protein side-chain packing (PSCP) is the problem of predicting the coordinates of side-chain atoms given fixed backbone coordinates, and it is useful across various tasks in the field of structural biological modeling. While traditional PSCP methods are primarily trained on experimentally-determined structures from the Protein Data Bank (PDB), AlphaFold—a paradigm-shifting, machine learning-based, protein structure prediction tool released by DeepMind—has made available AlphaFoldDB, a database of high-quality synthetic protein structures which massively expands the structural coverage space from the PDB by multiple orders of magnitude. Herein, we aimed to determine whether PSCP methods could benefit from substituting their training data with AlphaFoldDB structures. Using a recent protein encoder named the orientation-aware graph neural network as our testing framework, we find that the high-confidence predicted protein structures from AlphaFoldDB are not suitable direct replacements for native chains in side-chain modeling, with such a replacement causing significant degradations in performance across various evaluation metrics. We also explore an approach of cross-distilling knowledge from AlphaFold's network by using both datasets simultaneously for our model to learn, which displays better results than using either dataset individually, regardless of the backbone type supplied during inference.

1 Introduction

The side-chains of a protein are what dictate its functions and interactions [1–3], so the accurate prediction of their 3-dimensional conformations given the structural arrangement of the protein's backbone atoms, termed the protein side-chain packing (PSCP) task, has significant implications in the computational modeling of macromolecular structures and dynamics [4–7]. As such, this task has garnered considerable attention and many methods have been developed to solve it in recent decades [8–15]. More recent methods use deep learning, such as diffusion [16] or tensor field networks [17], while previous methods search through libraries of side-chain configurations to minimize the biophysical free energy of the structure [18–20]. However, of note is that all of these methods were developed using experimentally-determined, or native, structural data present in the Protein Data Bank (PDB), whether as training data for their machine learning models or as the database from which the statistical distributions of rotational isomer libraries and energy functions were drawn [21, 22].

Since experimental methods that are commonly used for obtaining these deposited PDB structures—like X-ray crystallography and NMR spectroscopy—can be cost-prohibitive [23], a scalable alternative to these structural data would be computationally-generated protein structures. Specifically, we consider the proteins predicted by AlphaFold [24], a machine learning-based tool which generates

Submitted to the AI for Science workshop (NeurIPS 2025)

full-atom structures based on amino acid sequences and which has revolutionized computational biology [25, 26]. A massive dataset of AlphaFold2-predicted structures called the AlphaFoldDB has grown to over 214 million protein structures in less than 5 years, eclipsing the comparatively scarce PDB of approximately 240,000 structures by almost 900x [27, 28]; therefore, there is a large gap in structurally covering known protein sequences that the AlphaFoldDB has, effectively, successfully filled.

Though AlphaFold's structure prediction accuracy has been demonstrated to be on par with experimental lab-based methods for some use cases [26, 29, 30], there are notable differences between the distributions of its predicted side-chains and the side-chains of experimental proteins deposited into the PDB [31, 32]. So, we tested the extent to which such conformational variations have a tangible effect on downstream tasks by taking advantage of the AlphaFoldDB as a training dataset for a PSCP method, effectively treating AlphaFoldDB structures as substitutes for experimentally-solved structures based on the confidence scores AlphaFold assigns to its own predictions, and compared that to only using experimental structures during training. Our contributions are the following:

- We empirically show that such a substitution of native proteins with synthetic proteins causes both consistent and significant degradations in performance in side-chain prediction, meaning that protein structures from the AlphaFoldDB may not function as effective training data for sequences not structurally covered, despite their volume.
- To alleviate this, we investigate an approach that merely combines both datasets, learning from experimental backbone-sidechain distributions while distilling knowledge from AlphaFold2's built-in side-chain prediction network via the structures in the AlphaFoldDB. This data augmentation method consistently outperforms the usage of either dataset individually, and is also more resilient to the type of backbone (experimental vs. synthetic) given as input during inference.
- For testing these different datasets and arriving at these results, we develop a PSCP tool based on the orientation-aware graph neural network, a state-of-the-art protein structure representation embedder [33].

The relevant data and code used can be found at `https://zenodo.org/records/16937575` and `https://github.com/snkv04/CrossDistillationPSCP`, respectively.

2 Methods

2.1 Preparation of training datasets

The first two training datasets we use for experimentation in this study consist of experimentally-determined protein structures and computationally-generated structures, respectively. For the former, we use BC40 [34], a publicly available snapshot of the PDB from 2020 that is clustered at the standard threshold of 40% sequence identity [35] and which has been used by several other PSCP methods [14–17]. For the latter, we retrieve high-confidence entries from the AlphaFoldDB [27], which contains approximately 999,000 AlphaFold2-predicted protein tertiary structures hosted centrally in the PDB as of May 2025. Further details on data retrieval and confidence filtering are in **Appendix A**.

To avoid data leakage, the BC40 training set is cleansed of any protein chains above the 40% sequential similarity cutoff with any of our test sets. The same is done for the AFDB structures, with 2 extra filtering operations: they are internally clustered by sequence identity to eliminate data redundancy, and the structures that are above the similarity cutoff to any chains in BC40 are removed to avoid over-representing known, structurally covered amino acid chains. To do this, we used the CD-HIT and CD-HIT 2-Dataset algorithms [36] with a word length of 2, maximum memory usage of 16,000 MB, and 8 threads. After filtering, we ended up with 35,094 BC40 chains and 69,390 AFDB chains.

Our third dataset simply comprises the union of the chains from these two sets. We term it "cross-distillation" (abbreviated as "CD" in tables) because it creates a cross-dataset training strategy that simultaneously performs supervised learning using the ground-truth side-chain labels from BC40 and undergoes supervised cross-architecture knowledge distillation [37] by learning from AlphaFold2's joint backbone-sidechain distribution in AFDB.

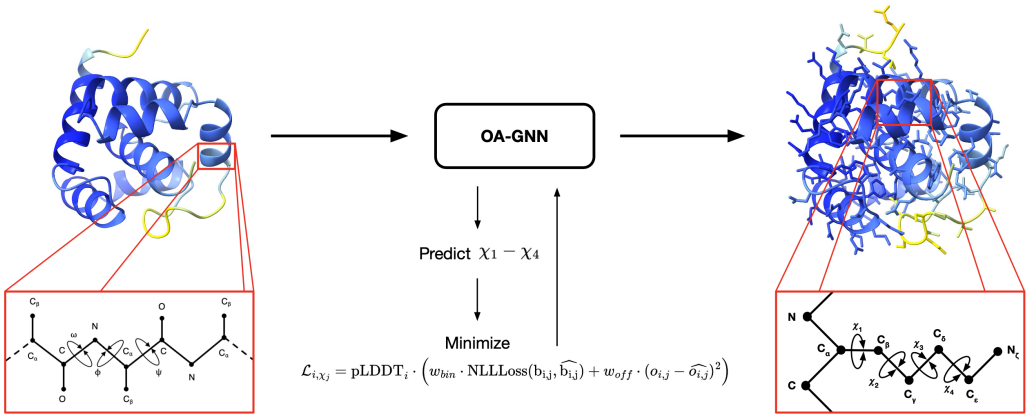

Figure 1: The simplified PSCP framework. AlphaFold2-predicted CASP15 target T1104 (PDB ID: 7ROA) is shown, with the chain colored according to ChimeraX's pLDDT-based "AlphaFold" color palette [38]. The OA-GNN intakes backbone encodings and predicts side-chain torsional angles.

2.2 Preparation of test datasets

Our test dataset of proteins consists of the prediction targets used in the 16$^{\text{th}}$ Critical Assessment of Structure Prediction (CASP) competition, and they were retrieved directly from the Prediction Center contest site. We additionally obtain predictions from both AlphaFold2 and AlphaFold3 for the corresponding acid sequences, in order to assess the accuracy and generalizability of our training datasets when synthetic backbones are provided as inputs during evaluation. The full target list is provided in **Appendix B**, and further details on the reference set retrieval are given in **Appendix A**.

2.3 Testing framework

Formally, the PSCP task aims to approximate the function $f : \mathcal{A}^n \times \mathbb{R}^{n\times4\times3} \to \mathbb{R}^{n\times10\times3}$ that, given a protein's amino acid sequence $\mathcal{S} \in \mathcal{A}^n$ and the coordinates $\mathbf{X}_{bb}$ of the 4 backbone atoms for each of the n amino acids in the sequence, outputs the side-chain atom coordinates $\mathbf{X}_{sc}$, where $\mathcal{A}$ contains 20 amino acid types and each can have up to 10 side-chain heavy atoms (e.g., tryptophan [24]). See **Figure 1** for an example.

To perform this prediction and test out the different training datasets, we adapt the default convolutional variant of the orientation-aware graph neural networks (OA-GNNs) [33], as it is a multi-purpose representation embedder that has demonstrated SOTA results in learning geometric information from protein structures. At inference-time, for a given protein chain, we produce input embeddings from the backbone $\mathbf{X}_{bb}$ and sequence $\mathcal{S}$ (see **Appendix C** for more details on input features), propagate the representations through encoder and decoder layers which perform SO(3)-equivariant message passing, apply directed-weight perceptrons [33] to obtain the side-chain torsion angles $\chi_1 - \chi_4$, and reconstruct the coordinates $\mathbf{X}_{sc}$ using ideal bond angles and lengths as others have done [14–16]; **Appendix D** contains further information on our integration of the OA-GNN for side-chain prediction. Additionally, by performing linear binning on the range of possible χ angles $[-\pi, \pi]$, thereby translating each χ into a bin index and an offset from the center of the corresponding bin (similarly to [14]), we effectively treat the problem as a classification task during training. Further details regarding training are provided in **Appendix E**.

2.4 Experimental setup

After training the model on each dataset, we provided it the native protein backbones from CASP16, ran inference on each, and computed the evaluation metrics for the outputted side-chains (described below in Section 2.5), taking the average across protein targets. The same is done for the AlphaFold2- and AlphaFold3-generated backbones for the sequences in this test set.

Table 1: Results of training on different datasets when evaluating the tester network on native backbones from CASP16 ($n = 50$ proteins). (For each metric, bold indicates best performance.)

	RMSD (Å)			χ-MAE (°)				χ-RR. (%)	Steric Clashes (#)		
Train set	All	Core	Surface	χ_1	χ_2	χ_3	χ_4	χ_{1-4}	100%	90%	80%
CD	**0.82**	**0.42**	**1.04**	**23.20**	**27.27**	**44.24**	55.12	**57.8**	**118.0**	**8.9**	**0.7**
BC40	0.85	0.42	1.08	24.41	27.85	45.51	55.74	56.6	119.8	9.4	0.8
AlphaFoldDB	1.14	0.68	1.37	36.33	36.18	53.80	**54.13**	45.1	187.3	27.4	3.3

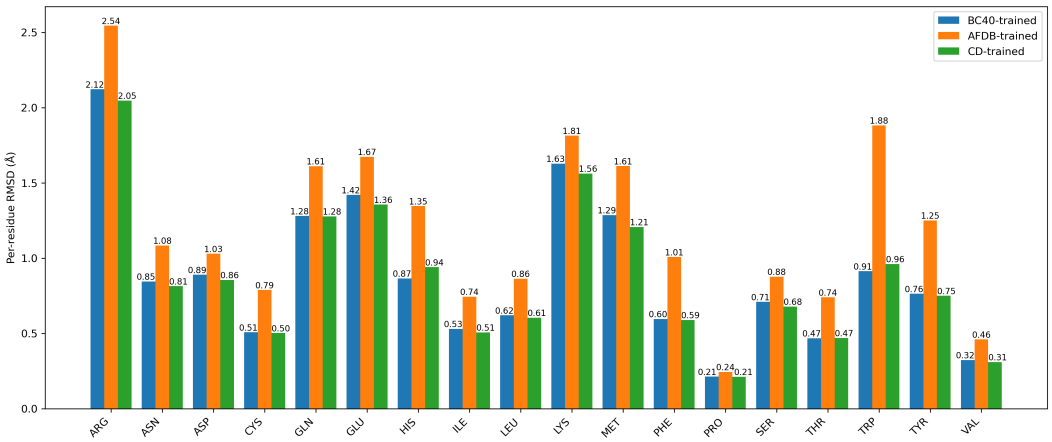

Figure 2: Average per-residue RMSDs on each amino acid type, when evaluated on native backbones from CASP16 ($n = 16,074$ residues). Alanine and glycine were excluded as they do not possess any side-chain dihedral angles for the network to output.

2.5 Evaluation metrics

In determining the accuracy of a PSCP method, we compare its side-chains to the ground-truth, native side-chains using 4 main evaluation metrics: **(1)** the average root mean square deviation (RMSD) over all side-chain atoms when predicted and target structures are optimally superposed, **(2)** the mean absolute error (MAE) of the side-chain dihedral angles χ_1 through χ_4, **(3)** the angular recovery rate, which is the proportion of residues with all 4 angle MAEs under 20°, and **(4)** steric clash count, which measures biophysical realism by tallying the number of atom pairs whose positions overlap. The distance threshold for overlapping is computed at different percentages (100, 90, 80) of the sum of the atoms' van der Waals radii. As is common in the literature [14–17], we can further break down metrics by residue category, where "core" residues have ≥ 20 other residues' C_β atoms within a 10Å radius of theirs and "surface" residues have ≤ 15 such other residues.

3 Results and discussion

3.1 Evaluation on native backbones

PDB vs. AFDB: A summary of our primary experiment is shown in **Table 1**. We observe that the AFDB-trained model displays significantly lower performance than the BC40-trained model across almost every metric, such as with its all-atom RMSD jumping to over 1 Å and its rotameric recovery rate dropping by over 11%. Considering that "AlphaFold is trained to produce the protein structure most likely to appear as part of a PDB structure" [24], providing the AFDB-trained model with native backbones during inference represents the best case for such a model, but we empirically see that this model incurs higher errors on both coordinate-based and angle-based metrics. Looking further, **Figure 2** demonstrates that the increase in atomic deviations is consistent across every amino acid, with over a 2x jump in RMSD on tryptophan residues. As a result, training on AFDB structures for sequences that do not possess any experimentally-determined tertiary structure may not be an accurate method of capturing atom-level side-chain patterns for such sequences. Similar conclusions can be

Table 2: Results of training on different datasets when evaluating the tester network on AlphaFold-generated backbones from CASP16 ($n = 50$ proteins). The "Baseline" rows refer to AF2's or AF3's (corresponding to the evaluation set) baseline side-chain predictions for the same sequences. (For each metric, bold indicates best performance and underline indicates second-best.)

	RMSD (Å)			**χ-MAE (°)**				**χ-RR. (%)**	**Steric Clashes (#)**		
Train set	All	Core	Surface	χ_1	χ_2	χ_3	χ_4	χ_{1-4}	100%	90%	80%
AlphaFold2-generated backbones											
Baseline	**1.02**	**0.61**	**1.22**	**32.41**	**31.96**	**48.06**	54.51	**46.5**	**137.0**	50.1	28.9
CD	1.08	0.62	1.28	34.90	32.60	48.90	**52.36**	46.2	141.2	27.9	**10.3**
BC40	1.11	0.64	1.32	35.79	33.96	51.69	60.74	45.8	142.1	**27.5**	10.4
AlphaFoldDB	1.15	0.67	1.34	37.64	34.40	51.25	52.36	44.4	156.5	36.1	13.2
AlphaFold3-generated backbones											
Baseline	**0.96**	**0.54**	**1.17**	**30.37**	**29.72**	**46.57**	53.31	**51.5**	**86.0**	13.4	1.7
CD	1.02	0.58	1.23	32.37	31.93	50.62	53.91	49.3	101.5	**7.8**	0.6
BC40	1.03	0.59	1.26	32.57	32.66	51.60	61.23	49.1	104.1	7.9	**0.4**
AlphaFoldDB	1.05	0.59	1.27	33.46	32.39	50.30	**53.07**	48.5	105.2	8.3	0.7

drawn from evaluation on backbones from two older CASP competitions, CASP14 and CASP15, for which the results are given in **Appendix F**.

Training with cross-distillation: Another clear outcome is that by augmenting the BC40 training dataset with the synthetic structures from AFDB, the network learns to output more accurate side-chains on nearly all metrics than when using either dataset individually. This indicates that while a model cannot purely depend upon the latter, if ground-truth data are present to learn the prior backbone-sidechain patterns from, then the AlphaFold-generated structures can supplement the model's training.

A disadvantage of this cross-dataset, cross-architecture distillation approach, however, is the longer training time; given the 3x larger dataset making each epoch lengthen, and the noiser synthetic structures resulting in more epochs before convergence, the training still takes over 4x longer than training on only BC40 even with double the GPUs (see **Appendix E**). This might be mitigated through alternative data preprocessing strategies as discussed in Section 4. Additionally, the cross-distillation dataset presents marginally higher atomic coordinate errors than the BC40 dataset on the histidine, threonine, and tryptophan acids as seen in **Figure 2**, though this may be a result of acids with polar functional groups like these having lower accuracies in AlphaFold-produced synthetic structures [39].

3.2 Evaluation on AlphaFold-generated backbones

The results from our supplementary experiments, where we provide the OA-GNN AlphaFold-generated protein backbones during evaluation, are presented in **Table 2**. It can be seen that the backbone-sidechain pairings from AF2-outputted structures contain enough noise that training on them (via AFDB) and then using AF2-outputted backbones again as input at inference-time results in less accurate side-chains than simply training on native structures (corresponding to the BC40 row). The same holds when AF3 backbones are used as inputs. However, the cross-distillation dataset consistently outperforms the other two on chains generated by AF3, which is a property shared by no chains in its constituent datasets, showing that augmentation using synthetic data can possibly enable greater generalizability as synthetic protein structures grow more abundant [40].

We acknowledge that none of these training datasets for the OA-GNN exceed AlphaFold's baseline performance across most metrics, though this is a known problem with all major PSCP methods [41]; a likely reason for this is that AlphaFold (both v2 and v3) jointly refines its backbones and side-chains during its "recycling" update steps [24, 35], conditioning them on each other, while PSCP methods are effectively post-processing steps that are only informed of the final backbone conformations. However, there is undeniably room for improvement in the prediction of side-chains on AlphaFold's backbones beyond AlphaFold's own performance, given the lower error distributions of PSCP methods when experimental backbones are supplied as input (see **Table 1**).

4 Conclusion

For the first time, we evaluate the usage of AlphaFoldDB structures as training data for the task of protein side-chain packing. Our main finding is that high-confidence AFDB structures empirically cannot function as substitutes of experimental PDB structures when training a PSCP method, even when the former set is nearly twice in the number of protein chains; this demonstrates difficulties in addressing PDB data scarcity for tasks that can be approached with deep learning such as this. As an additional study, we investigate a cross-dataset approach combining native and synthetic proteins during training, and find that it generalizes better than either individual training dataset to all evaluation sets. Further work involves 2 primary directions: **(1)** experimenting with different methods of combining PDB and AFDB data, such as pre-training/fine-tuning [42] or curriculum learning [43] which have leveraged synthetic data in other domains, and **(2)** refining AlphaFold structures before training on them, for instance through physics-informed molecular dynamics simulations [44].

References

[1] Yazan Haddad, Vojtech Adam, and Zbynek Heger. Rotamer Dynamics: Analysis of Rotamers in Molecular Dynamics Simulations of Proteins. *Biophysical Journal*, 116(11):2062–2072, Jun 2019.

[2] Pinak Chakrabarti and Debnath Pal. The Interrelationships of Side-Chain and Main-Chain Conformations in Proteins. *Progress in Biophysics and Molecular Biology*, 76(1):1–102, 2001.

[3] Tim Clackson and James A. Wells. A Hot Spot of Binding Energy in a Hormone-Receptor Interface. *Science*, 267(5196):383–386, 1995.

[4] Paul K. Warme and Richard S. Morgan. A Survey of Amino Acid Side-Chain Interactions in 21 Proteins. *Journal of Molecular Biology*, 118(3):289–304, 1978.

[5] Annett Bachmann, Dirk Wildemann, Florian Praetorius, Gunter Fischer, and Thomas Kiefhaber. Mapping Backbone and Side-Chain Interactions in the Transition State of a Coupled Protein Folding and Binding Reaction. *Proceedings of the National Academy of Sciences*, 108(10):3952–3957, 2011.

[6] Xiaoqiang Huang, Robin Pearce, and Yang Zhang. Toward the Accuracy and Speed of Protein Side-Chain Packing: A Systematic Study on Rotamer Libraries. *Journal of Chemical Information and Modeling*, 60(1):410–420, 2020. PMID: 31851497.

[7] Maximiliano Vásquez. Modeling Side-Chain Conformation. *Current Opinion in Structural Biology*, 6(2):217–221, 1996.

[8] Tim Harder, Wouter Boomsma, Martin Paluszewski, et al. Beyond Rotamers: A Generative, Probabilistic Model of Side Chains in Proteins. *BMC Bioinformatics*, 11(1):306, Jun 2010.

[9] Shide Liang, Dandan Zheng, Chi Zhang, et al. Fast and Accurate Prediction of Protein Side-Chain Conformations. *Bioinformatics*, 27(20):2913–2914, Aug 2011.

[10] Zhichao Miao, Yang Cao, and Taijiao Jiang. RASP: Rapid Modeling of Protein Side Chain Conformations. *Bioinformatics*, 27(22):3117–3122, Sep 2011.

[11] Ken Nagata, Arlo Randall, and Pierre Baldi. SIDEpro: A Novel Machine Learning Approach for the Fast and Accurate Prediction of Side-Chain Conformations. *Proteins: Structure, Function, and Bioinformatics*, 80(1):142–153, 2012.

[12] Jiale Liu, Changsheng Zhang, and Luhua Lai. GeoPacker: A Novel Deep Learning Framework for Protein Side-Chain Modeling. *Protein Science*, 31(12):e4484, 2022.

[13] Mikita Misiura, Raghav Shroff, Ross Thyer, et al. DLPacker: Deep Learning for Prediction of Amino Acid Side Chain Conformations in Proteins. *Proteins: Structure, Function, and Bioinformatics*, 90(6):1278–1290, 2022.

[14] Nicholas Z. Randolph and Brian Kuhlman. Invariant Point Message Passing for Protein Side Chain Packing. *Proteins: Structure, Function, and Bioinformatics*, 92(10):1220–1233, 2024.

[15] Jin Sub Lee and Philip M Kim. FlowPacker: Protein Side-Chain Packing with Torsional Flow Matching. *Bioinformatics*, page btaf010, Jan 2025.

[16] Yangtian Zhang, Zuobai Zhang, Bozitao Zhong, et al. DiffPack: A Torsional Diffusion Model for Autoregressive Protein Side-Chain Packing. In *37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

[17] Matthew McPartlon and Jinbo Xu. An End-to-End Deep Learning Method for Protein Side-Chain Packing and Inverse Folding. *Proceedings of the National Academy of Sciences*, 120(23):e2216438120, 2023.

[18] Georgii G. Krivov, Maxim V. Shapovalov, and Roland L. Dunbrack Jr. Improved Prediction of Protein Side-Chain Conformations with SCWRL4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, 2009.

[19] Andrew Leaver-Fay, Michael Tyka, Steven M. Lewis, et al. Rosetta3: An Object-Oriented Software Suite for the Simulation and Design of Macromolecules. In Michael L. Johnson and Ludwig Brand, editors, *Computer Methods, Part C*, volume 487 of *Methods in Enzymology*, pages 545–574. Academic Press, 2011.

[20] Xiaoqiang Huang, Robin Pearce, and Yang Zhang. FASPR: An Open-Source Tool for Fast and Accurate Protein Side-Chain Packing. *Bioinformatics*, 36(12):3758–3765, Apr 2020.

[21] Maxim V. Shapovalov and Roland L. Dunbrack Jr. A Smoothed Backbone-Dependent Rotamer Library for Proteins Derived from Adaptive Kernel Density Estimates and Regressions. *Structure*, 19(6):844–858, Jun 2011.

[22] Rebecca F. Alford, Andrew Leaver-Fay, Jeliazko R. Jeliazkov, et al. The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design. *Journal of Chemical Theory and Computation*, 13(6):3031–3048, 2017. PMID: 28430426.

[23] Stephen K. Burley, Andrzej Joachimiak, Gaetano T. Montelione, and Ian A. Wilson. Contributions to the NIH-NIGMS Protein Structure Initiative from the PSI Production Centers. *Structure*, 16(1):5–11, Jan 2008.

[24] John Jumper, Richard Evans, Alexander Pritzel, et al. Highly Accurate Protein Structure Prediction with AlphaFold. *Nature*, 596(7873):583–589, Aug 2021.

[25] Bernard Moussad, Rahmatullah Roche, and Debswapna Bhattacharya. The Transformative Power of Transformers in Protein Structure Prediction. *Proceedings of the National Academy of Sciences*, 120(32):e2303499120, 2023.

[26] Thomas J. Lane. Protein Structure Prediction Has Reached the Single-Structure Frontier. *Nature Methods*, 20(2):170–173, Feb 2023.

[27] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, Augustin Žídek, Tim Green, Kathryn Tunyasuvunakool, Stig Petersen, John Jumper, Ellen Clancy, Richard Green, Ankur Vora, Mira Lutfi, Michael Figurnov, Andrew Cowie, Nicole Hobbs, Pushmeet Kohli, Gerard Kleywegt, Ewan Birney, Demis Hassabis, and Sameer Velankar. AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, Nov 2021.

[28] Mihaly Varadi, Damian Bertoni, Paulyna Magana, et al. AlphaFold Protein Structure Database in 2024: Providing Structure Coverage for Over 214 Million Protein Sequences. *Nucleic Acids Research*, 52(D1):D368–D375, Nov 2023.

[29] Irène Barbarin-Bocahu and Marc Graille. The X-ray crystallography phase problem solved thanks to *AlphaFold* and *RoseTTAFold* models: a case-study report. *Acta Crystallographica Section D*, 78(4):517–531, Apr 2022.

[30] Andriy Kryshtafovych, Torsten Schwede, Maya Topf, Krzysztof Fidelis, and John Moult. Critical assessment of methods of protein structure prediction (CASP)—Round XIV. *Proteins: Structure, Function, and Bioinformatics*, 89(12):1607–1617, 2021.

[31] Thomas C. Terwilliger, Dorothee Liebschner, Tristan I. Croll, Christopher J. Williams, Airlie J. McCoy, Billy K. Poon, Pavel V. Afonine, Robert D. Oeffner, Jane S. Richardson, Randy J. Read, and Paul D. Adams. AlphaFold predictions are valuable hypotheses and accelerate but do not replace experimental structure determination. *Nature Methods*, 21(1):110–116, Jan 2024.

[32] Valeria Scardino, Juan I. Di Filippo, and Claudio N. Cavasotto. How good are AlphaFold models for docking-based virtual screening? *iScience*, 26(1):105920, 2023.

[33] Jiahan Li, Shitong Luo, Congyue Deng, Chaoran Cheng, Jiaqi Guan, Leonidas Guibas, Jian Peng, and Jianzhu Ma. Orientation-Aware Networks for Protein Structure Representation Learning. In *Research in Computational Molecular Biology: 29th International Conference, RECOMB 2025, Seoul, South Korea, April 26–29, 2025, Proceedings*, page 1–16, Berlin, Heidelberg, 2025. Springer-Verlag.

[34] Qin Wang, Jun Wei, Yuzhe Zhou, Mingzhi Lin, Ruobing Ren, Sheng Wang, Shuguang Cui, and Zhen Li. Prior knowledge facilitates low homologous protein secondary structure prediction with DSM distillation. *Bioinformatics*, 38(14):3574–3581, Jun 2022.

[35] Josh Abramson, Jonas Adler, Jack Dunger, et al. Accurate Structure Prediction of Biomolecular Interactions with AlphaFold 3. *Nature*, 630(8016):493–500, Jun 2024.

[36] Limin Fu, Beifang Niu, Zhengwei Zhu, Sitao Wu, and Weizhong Li. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152, Oct 2012.

[37] Yufan Liu, Jiajiong Cao, Bing Li, Weiming Hu, Jingting Ding, and Liang Li. Cross-Architecture Knowledge Distillation. In *Computer Vision – ACCV 2022: 16th Asian Conference on Computer Vision, Macao, China, December 4–8, 2022, Proceedings, Part V*, page 179–195, Berlin, Heidelberg, 2022. Springer-Verlag.

[38] Thomas D. Goddard, Conrad C. Huang, Elaine C. Meng, Eric F. Pettersen, Gregory S. Couch, John H. Morris, and Thomas E. Ferrin. UCSF ChimeraX: Meeting modern challenges in visualization and analysis. *Protein Science*, 27(1):14–25, 2018.

[39] Haifan Zhao, Heng Zhang, Zhun She, Zengqiang Gao, Qi Wang, Zhi Geng, and Yuhui Dong. Exploring AlphaFold2s Performance on Predicting Amino Acid Side-Chain Conformations and Its Utility in Crystal Structure Determination of B318L Protein. *International Journal of Molecular Sciences*, 24(3), 2023.

[40] Joan Segura, Jose Duarte, Sebastian Bittrich, Chunxiao Bi, Charmi Bhikadiya, Maryam Fayazi, Jeremy Henry, Igor Khokhriakov, Robert Lowe, Dennis W. Piehl, Brinda Vallat, Maria Voigt, John Westbrook, Yana Rose, and Stephen K. Burley. Exploring experimental structures and computed structure models from artificial intelligence/machine learning at RCSB Protein Data Bank (RCSB PDB, RCSB.org). *Biophysical Journal*, 122(3):282a, Aug 2023.

[41] Sriniketh Vangaru and Debswapna Bhattacharya. To pack or not to pack: revisiting protein side-chain packing in the post-AlphaFold era. *Briefings in Bioinformatics*, 26(3):bbaf297, Jun 2025.

[42] Maan Qraitem, Kate Saenko, and Bryan A. Plummer. From Fake to Real: Pretraining on Balanced Synthetic Images to Prevent Spurious Correlations in Image Recognition. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LVIII*, page 230–246, Berlin, Heidelberg, 2024. Springer-Verlag.

[43] Yijun Liang, Shweta Bhardwaj, and Tianyi Zhou. Diffusion Curriculum: Synthetic-to-Real Generative Curriculum Learning via Image-Guided Diffusion, 2024.

[44] Lim Heo, Giacomo Janson, and Michael Feig. Physics-based protein structure refinement in the era of artificial intelligence. *Proteins: Structure, Function, and Bioinformatics*, 89(12):1870–1887, 2021.

[45] Valerio Mariani, Marco Biasini, Alessandro Barbato, and Torsten Schwede. lDDT: a local superposition-free score for comparing protein structures and models using distance difference tests. *Bioinformatics*, 29(21):2722–2728, Aug 2013.

[46] Yanli Wang, Frimpong Boadu, and Jianlin Cheng. MPBind: Multitask Protein Binding Site Prediction by Protein Language Models and Equivariant Graph Neural Networks. *bioRxiv*, 2025.

[47] Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael J.L. Townshend, and Ron Dror. Learning from Protein Structure with Geometric Vector Perceptrons. In *International Conference on Learning Representations*, 2021.

[48] Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020.

[49] Sidhartha Chaudhury, Sergey Lyskov, and Jeffrey J. Gray. PyRosetta: A Script-Based Interface for Implementing Molecular Modeling Algorithms Using Rosetta. *Bioinformatics*, 26(5):689–691, Jan 2010.

A Data sources

BC40: We downloaded the per-chain secondary structure files from `https://drug.ai.tencent.com/protein/bc40/download.html`, extracted the corresponding per-protein PDB identifiers, and downloaded the corresponding tertiary/quarternary structure files from the RCSB PDB, using a variation of PIPPack's downloader script found at `https://github.com/Kuhlman-Lab/PIPPack/blob/main/data/bc40_dataset.py` that was modified to use multithreading. We then proceeded to split these PDB files into their chains, and only included the subset of them that corresponded to chains in the initial secondary structure set.

AlphaFoldDB: All of the AlphaFold2-generated protein structures we used were downloaded from the Computed Structure Models repository hosted on the RCSB PDB. Since these were predicted by AlphaFold2 and not AlphaFold2-Multimer, each of these proteins had a single chain, and thus did not need to be separated by chains like those in BC40.

AlphaFold (both v2 and v3) provides self-assessment metrics to quantify how accurate it estimates its own predictions to be [24, 35]. Two such metrics are pLDDT, which approximates the superposition-free local distance difference test (lDDT) metric [45] on a scale of $[0, 100]$ with higher values corresponding to greater confidence, and predicted aligned error (pAE), which is the expected deviation in Ångströms of atoms on residue i if the backbone atoms of residue j are aligned with the ground truth for every pair (i, j). Similar to the procedure in [46] used for finding high-quality human proteome structures for evaluation, we select structures with a pLDDT ≥ 70 and an average pAE across all residue pairs ≤ 10 Å.

CASP14 and CASP15: The data retrieval process is the same as described in [41].

CASP16:

- **Native:** Downloaded the "whole" tertiary structure files from `https://predictioncenter.org/download_area/CASP16/targets/_4invitees/`
- **AF2-generated:** Downloaded the in-contest predictions from Group 145 in CASP16, located at `https://predictioncenter.org/download_area/CASP16/predictions/regular/`. These were generated by ColabFold (`https://github.com/sokrypton/ColabFold`), the open-sourced version of AlphaFold2.
- **AF3-generated:** Downloaded the in-contest predictions from Group 304 in CASP16, located at `https://predictioncenter.org/download_area/CASP16/predictions/regular/`. These were generated using the AlphaFold3 server (`https://alphafoldserver.com/`) by the team that submitted the predictions.

B Target lists for the CASP datasets

The methodology of our curation of targets for CASP14 and CASP15 is the same as described in [41]. For CASP16, the targets were curated as follows:

- For prediction targets with multiple versions (e.g., "T1357v2") present on the Prediction Center, we took the version without a version tag if present, and the "v1" version otherwise. Following this strategy, we removed the following 10 targets:
  - T1214v1, T1214v2 (and kept T1214)
  - T1228v2, T1228v3, T1228v4 (and kept T1228v1)
  - T1239v2, T1239v3, T1239v4 (and kept T1239v1)
  - T1249v2 (and kept T1249v1)
  - T1294v2 (and kept T1294v1)
- These 11 targets were removed as corresponding AlphaFold2 or AlphaFold3 predictions were not present for them within Prediction Center's CASP16 contest submissions: T1271s1, T1271s2, T1271s3, T1271s4, T1271s5, T1271s6, T1271s7, T1271s8, T1272s1, T1279, T1295

The full list of targets for each dataset is given in Table 3.

Table 3: List of targets used in our experiments.

Dataset	Target List
CASP14 ($n = 66$)	T1024, T1026, T1027, T1029, T1030, T1031, T1032, T1033, T1034, T1035, T1037, T1038, T1039, T1040, T1041, T1042, T1043, T1045s2, T1046s1, T1046s2, T1047s1, T1047s2, T1048, T1049, T1050, T1052, T1053, T1054, T1055, T1056, T1057, T1058, T1060s2, T1060s3, T1061, T1062, T1064, T1065s1, T1065s2, T1067, T1068, T1070, T1072s1, T1073, T1074, T1076, T1078, T1079, T1080, T1082, T1083, T1084, T1087, T1088, T1089, T1090, T1091, T1092, T1093, T1094, T1095, T1096, T1098, T1099, T1100, T1101
CASP15 ($n = 71$)	T1104, T1106s1, T1106s2, T1109, T1110, T1112, T1113, T1114s1, T1114s2, T1114s3, T1119, T1120, T1121, T1122, T1123, T1124, T1125, T1127, T1129s2, T1130, T1131, T1132, T1133, T1134s1, T1134s2, T1137s1, T1137s2, T1137s3, T1137s4, T1137s5, T1137s6, T1137s7, T1137s8, T1137s9, T1139, T1145, T1146, T1147, T1150, T1151s2, T1152, T1153, T1154, T1155, T1157s1, T1157s2, T1158, T1159, T1160, T1161, T1162, T1163, T1170, T1173, T1174, T1175, T1176, T1177, T1178, T1179, T1180, T1181, T1182, T1183, T1184, T1185s1, T1185s2, T1185s4, T1187, T1188, T1194
CASP16 ($n = 50$)	T1201, T1206, T1207, T1208s1, T1208s2, T1210, T1212, T1214, T1218, T1220s1, T1226, T1227s1, T1228v1, T1231, T1234, T1235, T1237, T1239v1, T1240, T1243, T1244s1, T1244s2, T1245s1, T1245s2, T1246, T1249v1, T1257, T1259, T1266, T1267s1, T1267s2, T1269, T1270, T1272s2, T1272s3, T1272s4, T1272s5, T1272s6, T1272s7, T1272s8, T1272s9, T1274, T1276, T1278, T1280, T1284, T1292, T1294v1, T1298, T1299

C Input features

When constructing the input to the neural network model, we first represent the protein's tertiary structure as a graph, with nodes corresponding to each amino acid residue and edges connecting each node to its k-nearest neighbors (where we set $k = 30$) based on the Euclidean distance between the residues' C_α backbone atoms. The OA-GNN framework enables the usage of vector-list features, as their directed-weight operations are designed to semantically encode directional information. So, for each node and each edge in this protein graph, we use both scalar-list and vector-list features, as described below.

C.1 Scalar features

Node-level scalar features: We used AttnPacker's input feature generation and embedding implementations for encoding backbone information. The features used include the residue's sequence number, the backbone dihedral angles ϕ, ψ, and ω, the count of other residues' C_β atoms within a short distance (12 Å) of that residue's C_β atom, and the residue's amino acid type. We additionally encode the secondary structure of each residue as a node-level feature, which can be an alpha helix, a beta sheet, or a coil/loop.

Edge-level scalar features: We similarly use AttnPacker's implementation for generating and embedding the edge features. For each edge (i, j), we encode the distance of the $(C_{\alpha_i}, C_{\alpha_j})$ atom pair to provide the chain's virtual bond geometry, the (C_{α_i}, N_j), (C_{α_i}, C_j), and $(C_{\alpha_i}, C_{\beta_j})$ atom pairs to capture the remaining polypeptide chain geometry, and the (N_i, O_j) atom pair to represent hydrogen bonds. In order to avoid unfairly embedding the real C_β atom positions, since we are predicting side-chain atom coordinates, the C_β coordinates for residue k are imputed by calculating the direction from residue k's C_α to C_β as done in [47]:

$$C_{\beta_k} - C_{\alpha_k} = \sqrt{\frac{1}{3}} \left(\frac{\mathbf{n} \times \mathbf{c}}{||\mathbf{n} \times \mathbf{c}||_2} \right) - \sqrt{\frac{2}{3}} \left(\frac{\mathbf{n} + \mathbf{c}}{||\mathbf{n} + \mathbf{c}||_2} \right)$$

where $\mathbf{n} = N_k - C_{\alpha_k}$, $\mathbf{c} = C_k - C_{\alpha_k}$, and $||\mathbf{v}||_2$ is the L^2-norm of vector $\mathbf{v}$. The remaining two scalar edge features are the sequence distance for the edge (i.e., $j - i$) and the ϕ, ψ, and ω transform-restrained Rosetta (trRosetta) dihedral angles between residues i and j as defined in [48].

All of these scalar features, for both nodes and edges, are first one-hot encoded and then concatenated together.

C.2 Vector features

Node-level vector features: We use the same vector features provided by the OA-GNN library. The first two features are the unit vectors from residue i's C_α to $C_{\alpha_{i-1}}$ and to $C_{\alpha_{i+1}}$. In the case that the sequence numbers are non-contiguous between residues i and $i+1$, indicating a break in the chain, we still use these $C_{\alpha_{i+1}} - C_{\alpha_i}$ and $C_{\alpha_i} - C_{\alpha_{i+1}}$ vectors, as they point to where the chain resumes either in the forward or backward direction. The third vector feature is the unit vector $C_{\beta_i} - C_{\alpha_i}$, using the same imputation process as described in Section C.1. The lengths of these distances are fixed in native structures, so there is no need to additionally use scalar features to encode the distances for these atom pairs.

Edge-level vector features: We use the same 5 cross-residue atom pairs as we used for the scalar edge features in Section C.1 except we now compute the unit vectors in those directions. By capturing both the scalar magnitude and the unit vector for the direction of each of these atom pairs, we encode their full orientations in 3D space.

D Architecture and inference details

Network configurations: Our OA-GNN uses 3 encoder and 3 decoder convolutional message-passing layers, with the hidden scalar-vector node representations having dimension $(s = 128, v = 32 \times 3)$ and edge representations having dimension $(s = 64, v = 16 \times 3)$. We use a standard dropout rate of 0.1 for the graph convolutional layers, for the 4 final dense layers that map from each node's hidden representation $\mathbf{h}_u$ to its per-bin logits, and for the 4 final dense layers that map from each node's hidden representation to its bin's offset.

Bin prediction: Instead of directly outputting the 4 χ angles for each node from the OA-GNN, we partially discretize the output space by splitting the range $[-\pi, \pi]$ into bins and then predict both the discrete-valued bin index and the offset from the bin's center for each χ angle. As recommended by [14], we use $n_{bin} = 72$ bins so that a 5° interval constitutes each bin.

Multi-channel outputs: The hidden scalar-vector tuple representation $\mathbf{h}_u \in \left(\mathbb{R}^{s_{dim}}, \mathbb{R}^{v_{dim} \times 3}\right)$ of node u after the last convolution layer is shared across all 4 χ angles for that node. We apply 4 separate directed-weight perceptrons (DWPs) $\mathcal{F}_i : \left(\mathbb{R}^{s_{dim}}, \mathbb{R}^{v_{dim} \times 3}\right) \to (\{0, \ldots, n_{bin} - 1\}, 0)$ to $\mathbf{h}_u$ in order to predict the final bin for each χ_i, as experimentation revealed that these separate channels for independently performing the final prediction of each torsion angle's bin resulted in lower errors across all 4 χ angles during cross-validation than using a single DWP $\mathcal{F} : \left(\mathbb{R}^{s_{dim}}, \mathbb{R}^{v_{dim} \times 3}\right) \to \left(\{0, \ldots, n_{bin} - 1\}^4, 0\right)$. The same is done for predicting the real-valued offset from the bin center for each χ_i.

Idealizer: For obtaining side-chain coordinates given backbone coordinates and side-chain torsional angles for each residue, we use FlowPacker's `Idealizer` module, which reconstructs the atomic positions using canonical bond geometries.

Energy minimization: Similarly to post-processing procedures used in [14, 17], we apply Rosetta's MinMover function [19] to our final structure using the official implementation of Rosetta in Python, PyRosetta [49]. This shifts the side-chain atoms by small amounts to minimize the REF2015 energy function [22], reducing steric clashes and outputting a more biophysically-realistic structure.

E Training details

Loss function: Given that we train on synthetic structures, we want to incentivize our network to trust losses on higher-confidence (and therefore, more reliable) residues more heavily. So, we first compute the average pLDDT p_i across backbone atoms for each residue i, scaling down to $[0, 1]$. (We do this as the granularity of pLDDT scores is at the residue-level for AlphaFold2 and atom-level for AlphaFold3, so even though all AFDB structures are from AlphaFold2, this formula is extendable to AlphaFold3-predicted structures.) Then, since we perform bin prediction for each residue i and

dihedral angle χ_j, we simply compute a weighted negative log-likelihood loss on the predicted bin probabilities $\widehat{b_{i,j}} \in \mathbb{R}^{n_{bin}}$ and mean-squared loss on the predicted offset $\widehat{o_{i,j}} \in [-\frac{\pi}{n_{bin}}, \frac{\pi}{n_{bin}}]$:

$$\mathcal{L}_{bin_{i,j}} = -p_i \cdot \sum_{k=0}^{n_{bin}-1} \text{one\_hot}(b_{i,j})_k \cdot \log\left(\widehat{b_{i,j,k}}\right)$$

$$\mathcal{L}_{off_{i,j}} = p_i \cdot (o_{i,j} - \widehat{o_{i,j}})^2$$

where $b_{i,j}$ and $o_{i,j}$ are the ground-truth bin index and offset, respectively. Then, the total loss for a single protein chain is just a weighted combination of the masked means of those two terms across all residues and all side-chain dihedral angles (so that we only consider such dihedral angles that actually exist):

$$\mathcal{L}_{total} = w_{bin} \cdot \text{masked\_mean}(\text{mask}_\chi, \mathcal{L}_{\text{bin}}) + w_{off} \cdot \text{masked\_mean}(\text{mask}_\chi, \mathcal{L}_{\text{off}})$$

where we set $w_{bin} = 1$ and $w_{off} = 100$ through grid search. This allows us to train on both native and synthetic protein chains using the same loss function (even if they're present in the same batch, such as when training on CD) by simply assuming all experimentally-determined residues' atoms to have full confidence (i.e., pLDDT $= 100$).

Dataset split and training configurations: For each of our 3 training datasets (BC40, AFDB, and CD), we imposed an 80-20 train-validation split. Cross-validation was performed every epoch, with the final checkpoint that was used for inference being the one with the lowest validation loss. A mini-batch size of 16 was used, and for the CD dataset, the proportion of native to non-native chains was fixed in each epoch but could vary freely within each batch.

The OA-GNN was trained on 4 Nvidia A100 GPUs for both BC40 and AFDB, taking 70 epochs (1.5 days) to converge—i.e., not have a decreasing validation loss for the next 20 epochs—on the former and 210 epochs (8.5 days) on the latter. We used 8 Nvidia A100 GPUs for CD, which took 245 epochs (7 days) to converge. For all datasets, a maximum gradient norm of 1.0 was used for clipping gradient weight updates. The Adam optimizer was used with $\alpha = 10^{-3}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$, and the exponential scheduler was used with $\gamma = 0.99$. The seed used for random initialization was 1048596.

F PSCP results on other testing datasets

The results from using different training datasets when CASP14 and CASP15 are used for evaluation are provided in Tables 4 and 5, respectively.

Table 4: Results of training on different datasets when evaluating the tester network on both native and non-native backbones from CASP14 ($n = 66$ proteins). The "Baseline" rows refer to AF2's or AF3's (corresponding to the evaluation set) baseline side-chain predictions for the same sequences. (For each metric, bold indicates best performance and underline indicates second-best.)

	RMSD (Å)			**χ-MAE (°)**				**χ-RR. (%)**	**Steric Clashes (#)**		
Train set	All	Core	Surface	χ_1	χ_2	χ_3	χ_4	χ_{1-4}	100%	90%	80%
Native backbones											
CD	**0.83**	**0.46**	**1.02**	**23.83**	**26.29**	**44.28**	50.60	**55.4**	**84.5**	10.2	2.3
BC40	0.85	0.49	1.04	24.47	26.82	44.86	52.81	54.9	86.7	**9.6**	**1.9**
AlphaFoldDB	1.16	0.70	1.38	37.98	34.65	52.79	**50.47**	42.4	132.2	23.0	4.7
AlphaFold2-generated backbones											
Baseline	**1.07**	**0.66**	1.26	**34.59**	31.51	50.81	51.42	**46.0**	**41.4**	**1.9**	**0.0**
CD	1.07	0.67	**1.26**	34.96	**31.48**	50.81	**49.84**	46.0	62.2	3.3	0.2
BC40	1.11	0.69	1.31	36.35	33.72	51.79	56.69	44.6	66.9	4.1	0.2
AlphaFoldDB	1.07	0.67	1.27	35.17	31.49	**50.69**	50.44	46.0	61.3	3.1	0.0
AlphaFold3-generated backbones											
Baseline	**1.04**	**0.64**	**1.25**	**34.14**	**30.35**	49.42	50.31	**47.4**	**45.8**	5.2	0.7
CD	1.07	0.67	1.29	35.31	31.65	**48.05**	49.32	45.9	59.7	**3.5**	**0.3**
BC40	1.09	0.68	1.29	35.82	32.67	49.22	54.16	45.4	61.1	3.8	0.4
AlphaFoldDB	1.12	0.70	1.34	37.23	32.76	50.22	**48.98**	44.2	63.3	4.4	0.5

Table 5: Results of training on different datasets when evaluating the tester network on both native and non-native backbones from CASP15 ($n = 71$ proteins). The "Baseline" rows refer to AF2's or AF3's (corresponding to the evaluation set) baseline side-chain predictions for the same sequences. (For each metric, bold indicates best performance and underline indicates second-best.)

	RMSD (Å)			**χ-MAE (°)**				**χ-RR. (%)**	**Steric Clashes (#)**		
Train set	All	Core	Surface	χ_1	χ_2	χ_3	χ_4	χ_{1-4}	100%	90%	80%
Native backbones											
CD	**0.72**	**0.36**	**0.93**	**19.72**	**23.17**	**41.52**	**51.59**	**64.3**	**84.6**	**6.2**	**0.5**
BC40	0.75	0.38	0.95	20.36	23.93	43.16	53.95	63.2	88.1	7.5	0.6
AlphaFoldDB	1.09	0.67	1.30	34.87	33.95	52.18	56.79	48.0	147.3	25.1	4.5
AlphaFold2-generated backbones											
Baseline	**0.90**	**0.58**	**1.11**	**28.05**	**27.90**	**48.04**	55.00	53.9	**48.2**	**2.0**	**0.0**
CD	0.93	0.60	1.14	29.05	28.44	48.33	**53.41**	**54.8**	78.4	5.0	0.3
BC40	0.98	0.65	1.19	31.25	30.30	50.94	56.68	53.0	84.2	6.6	0.4
AlphaFoldDB	0.93	0.59	1.14	28.94	28.93	48.33	56.19	54.2	79.3	5.1	0.2
AlphaFold3-generated backbones											
Baseline	**0.95**	**0.60**	**1.16**	**30.18**	**28.90**	**48.92**	**53.94**	**53.8**	**58.4**	8.1	1.0
CD	0.98	0.63	1.19	31.53	29.85	50.39	54.19	52.6	74.3	**5.0**	**0.3**
BC40	1.01	0.66	1.22	32.13	31.09	51.75	57.41	51.8	74.9	5.6	0.4
AlphaFoldDB	1.02	0.68	1.23	33.02	30.88	50.43	55.20	50.7	78.1	6.1	0.5