Predictive and Generative Models of Protein Sequence and Structure

by

Zeming Lin

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

January, 2024

Dr. Yann LeCun

Acknowledgments

I am fortunate to be advised by the esteemed Yann LeCun. Yann, thank you for your unconditional support. Your dedication to science and discovery is truly admirable. I found your wholehearted devotion to intellectual curiosity inspiring, and I hope to uphold the values you have imparted on me in my life and my work. I thank Rob Fergus and Kyunghyun Cho for your valuable feedback through my PhD and my thesis committee members Richard Bonneau and Ellen Zhang for their advice.

My PhD would not have been possible without the collaboration and support of the entire FAIR protein team at Meta AI. Alex Rives - you are a second advisor to me and I am still learning from your vision and drive. I am grateful to have worked closely with Halil Akin, Sal Candido, Brian Hie, Chloe Hsu, Roshan Rao, Tom Sercu, and Robert Verkuil through my time there. I would also like to thank Adam Lerer, Allan dos Santos Costa, Ammar Rizvi, Maryam Fazel-Zarandi, Nate Gruver, Ori Kabeli, Pascal Sturmfels, Shikai Qiu, Wenting Lu, Yilun Du, and Zhongkai Zhu, for all the wonderful discussions and collaborations.

Though my time at NYU was lonely due to joining at the start of the pandemic, I still made some great friends along the way. Thanks to Aishwarya Kamath, Alfredo Canziani, Jiachen Zhu, Nichola Carion, and Vlad Sobal, for brightening up my time at NYU.

My four years at FAIR as a research engineer, prior to my time at NYU, were incredibly formative. It was amazing to be in such an environment full of the smartest scientists in the world. My manager and friend during this time, Gabriel Synnaeve, was a kindred

spirit. Collaboration over research was seamless for us but we really bonded over our love for engineering and video games - I hope to be as kind and as talented as you one day. There were too many people to name during my time here, but I extend my sincerest appreciation for all the people I interacted with at FAIR.

I first embarked on this journey into machine learning and biology during my masters program at UVa. My advisor Yanjun (Jane) Qi was my first introduction into deep neural networks - and really set me up for success. Thank you for your support and advice. My journey into research realistically started in Thomas Jefferson High School for Science at Technology. I would like to show appreciation for all the wonderful teachers of this high school and the hard work they put in.

I would like to express my deepest gratitude to my family for their unwavering support throughout my academic journey. To my parents, whose guidance and encouragement have been invaluable, and I am profoundly grateful for everything you have done for me. To Anna, my wife, your love, encouragement, and understanding have been my pillars of strength. Thank you for being in my life.

Abstract

Historically, protein engineering has predominantly involved a bottom-up strategy, utilizing naturally occurring components as the building blocks. However, the problem of designing arbitrary protein sequences and structures for specific problems present significant challenges due to the complexity of biological systems. In this work, we tackle the problem of developing models of protein sequences and structures for prediction and generation. We show that neural networks can learn the patterns inherent to these systems and provide results for modeling protein through predicting protein structures from a given sequence and vice versa. Generative models can also model the unconditional distributions of protein sequence and structure.

To model protein structures, we present an autoencoder architecture that can produce a wide array of protein backbones to model protein structures. These structures exhibit both local and global coherence in terms of secondary and tertiary structures. Using classical techniques to design sequences that fold to generated backbones, we show that the model can generate novel sequences which are validated in-silico. To generate better sequences for these backbones, we then present ESM-IF1, a model for fixed backbone protein design. We designed a large-scale system to predict millions of structures using AlphaFold. By training on the synthetic data, we were able to obtain state of the art results and obtain over 50% sequence recovery.

We then scale large protein language models to 15 billion parameters (ESM-2) as an unconditional model of protein sequences. ESM-2 is capable of replacing multiple sequence alignment (MSA) features to obtain nearly state-of-the-art structure prediction results from a single sequence Removing MSA features gives a 60x speed up, allowing us to catalog the largest database of predicted protein structures. We open-sourced the ESM Metagenomic Atlas, a database of over 225 million high-confidence predicted structures, giving us an unprecedented view into the vast breadth and diversity of natural proteins. Finally, the speed and single sequence nature of our model allows us to directly optimize the protein sequence with respect to the protein structure. We show that black box optimization techniques can enable the design of proteins with structural constraints as symmetry, scaffolding, and binding. In sum, we present a series of models that are able to model the conditional and unconditional distributions of protein sequence and structure.

Contents

List of Figures

List of Tables

1 | Introduction

Over the past decade, in the fields of natural language processing and computer vision - hand design features have slowly given way to universal models with learned representations. With intelligent model design such as convolutional networks and transformers, we can learn the structure of text and pixels through data. In biology, protein sequences are often reasoned about using multiple sequence alignment (MSA), a method which searches genetic databases for similar sequences to represent the data. Similarly, techniques operating on protein structures use a combination of physics based methods and statistical force fields to model of these complex interactions. This thesis builds on prior works to model protein sequence and structures with neural systems learned on large amounts of data.

Protein structures are polymers of amino acids. Canonically, there are twenty distinct amino acids, which can be thought of as an alphabet that encodes all possible protein structures. Individually, protein chains in the human body tend to be anywhere between 100-1000 amino acids long, though short peptides are useful in signally and longer chains are found in places like human muscles. Many biological systems are a complex of a few to many tens of protein chains that act together in one machine, although in this work we mainly regard individual chains. This protein chains make up almost live in biological organisms, and interactions between these molecular machines result in the complexities of life.

Though one might imagine structures purely determine function, it turns out that our current imaging techniques are not capable of capturing the complexities of mobile protein structures. The standard way of imaging proteins via x-ray crystallography means that we must be able to capture molecules in a crystal lattice, which reduces dynamic proteins to a fixed structure in space. Fast acting functions such as catalysis can often not be captured in our structural datasets.

Because evolution and natural selection selects for sequences that tend to be more functional and helpful to an organisms' survival, the sequence distribution of natural proteins is then correlated with function. Rives et al. [16] shows that learning a model over the probability distribution of sequences is correlated to protein structural *and* functional properties. Furthermore, the exponential rise of sequence datasets due to shotgun sequencing techniques leads to a dramatic difference between number of known sequences and number of known structures. UniRef [17] at the time of this work has cataloged over 250 million protein sequences, whereas there are fewer than 500,000 protein chains available in PDB [18].

Classical state of the art methods for modeling protein structure depended on Rosetta [19], a physics based library that uses a force field approach learned through statistics. When the protein structure is close to the protein of interest, this is generally easy, but determining an optimal 3D scaffold was generally difficult, and monte-carlo methods using Rosetta tend to be trapped in local minima. As such, there is enormous interest in using neural models to learn distributions of protein structure from data. Anand and Huang [20] presents a model that can generically generate protein structure at a low fidelity and Eguchi et al. [21] specialize structure generation to antibodies. Furthermore, designs are often restricted to the protein backbone, which begs the question of which sequences might fold to the backbone.

That problem is referred to as fixed backbone protein design, an easier task since the amino acid identity is confined by the local interactions of the protein structure. Rosetta has several routines to do this, including FastDesign, though it requires a large number of monte-carlo calls and does not work reliably in practice. The advent of equivariant graph neural networks has led to large gains in performance, where one can construct a graph where the edges are determined by spatial distance [22, 23].

Finally, the protein sequence to structure task has been known as a "grand challenge" in biology. Successful methods [24, 25] tend to learn an energy function as input to Rosetta or other similar system - generally a pairwise distance map which constrains the global conformation. The culmination of these systems used distograms and pairwise angle constraints to build well predicted protein structures. AlphaFold during CASP14 [3] was a step change in structure prediction performance, almost halving the error rate with an entirely learned model. The main innovation of this work was to represent amino acids as a cloud of reference frames in space formed by the C-CA-N atoms, which are fixed in a tetrahedral bond. By disentangling the amino acids from each other, the optimization problem became easier, and neural models are able to make atom level coordinate predictions. Additionally, the tetrahedral bond gives gives an equivariant representation unique to protein structures, allowing for creative models to be tailor made for inference on protein structures.

With the advent of these methods, designing proteins became an easier process. Traditional protein design and engineering often took on approach based on segmentation and search [26, 27, 28, 29, 30, 31, 32]. Basic building blocks such as segments of proteins structures are used and the full protein is iteratively constructed from such blocks. This search problem tends to be highly non-trivial, and designing full proteins to meet a set of specifications is high complexity. Generation of protein structures via neural methods already hold promise, generating peptides and proteins from specific families [20, 21, 33, 34, 35, 36].

This thesis builds on top of these earlier works, to reach state of the art performance in both structure prediction and inverse folding with learned models. We tackle these four topics: (a) learning distributions of protein sequences, (b) learning distributions of protein structures, (c) predicting protein sequence from a given protein structure backbone, and (d) predicting the protein structure directly from protein sequences alone.

1.1 Statement on Prior Works

The work shown in this PhD thesis was a collective effort by many of my collaborators, and it would not have been possible without them. This section details the various publications and my part in the work. I list the main technical contributors and my contributions to the work.

[37] Oral at Machine Learning in Structural Biology Workshop 2021. I led this work and was the main technical contributor.

[6] Outstanding paper runner up at Internation Conference on Learning Representations 2022. Chloe Hsu was the main technical contributor on this work. My main contribution was working on the dataset generation procedure.

[38] Published at Science. I led the work on building out the structure prediction model, using the dataset from the previous paper as augmentations, as well as building out the scaling infrastructure for folding proteins for the metagenomic database, ESMAtlas. Halil Akin led the work in building out the language model scaling efforts. Roshan was responsible for many of the modeling ablations and experiments. Brian first proposed ESMAtlas and was key in analyzing our results from building the database. Thanks to all other contributors for their advice, building out the website, and the other work necessary for this effort.

[39] I contributed on the research direction of this work, with the first implementation of the MCMC loop for protein design. The majority of this work was led by Brian Hie and Sal Candido.

1.2 Overview

This dissertation attempts to describe several computational systems that model protein sequence and structure.

Chapter 2 models the distribution of protein structures by describing a system centered around a vector-quantized variational autoencoder (VQ-VAE) for generating protein backbones. By using a set of quantized vectors to describe the distance and angle distribution for protein structures, we create energy functions for Rosetta that are capable of folding into high quality protein backbone structures. We perform an evaluation of the various modeling choices in the VQ-VAE, and find a setting where the model is able to capture local and global topology and generate diverse protein backbones. Then, by designing sequences for these backbones with Rosetta-FixBB and validating sequence-structure generations with Alphafold, we discover the model is able to somewhat generalize and propose protein sequences with few hits in genetic databases. The work in this chapter predates AlphaFold2, which obsoletes some of this work with much better inductive biases.

Chapter 3 models the distribution of protein sequence from a fixed structure, i.e. fixed backbone protein design. We show that existing systems are data-bottlenecked, and improve inverse folding by augmenting structural data with high quality predictions from AlphaFold2. By making predictions for millions of protein sequences in UniRef, we achieve a state-of-the-art result. The proposed model combines a equvariant graph network with a transformer along with a autoregressive decoder to propose sequences for given structures.

Chapter 4 first trains a language model that models the distribution of protein sequences, and then tackles the structure prediction problem by predicting protein structures from sequences. We establish a link between language model pseudo-perplexity and structure prediction fidelity by training a set of LMs from 35M to 15B parameters. Existing systems use a genetic MSA search as an inital step in creating features for structure prediction, which can be slow and cumbersome. By replacing this with a language model, we can match the performance of Alphafold on a majority of protein sequences with inference speeds that are an order of magnitude faster. We then use our system ESMFold to make predictions for Mgnify, a gigantic database of metagenomic protein sequences. ESMFold predicts high confidence structures for 225M of these, creating the largest database of predicted protein structures to date. Cursory analysis is able to discover proteins with structures dissimilar to existing natural proteins in this database.

Chapter 5 ties together the work by designing sequences and structures jointly. We use black-box optimization techniques to optimize for sequences that fold to particular structures with ESMFold. The speed of ESMFold allows us to specify structural constraints in the form of energy functions, and use metropolis-hastings to sample protein sequences that abide by this function. We describe a programming language to build up arbitrary constraints, and show that without any modification, we are able to generate proteins that conform to many different structural and sequence specifications.

Chapter 6 concludes the dissertation and discusses future work.

2 Deep generative models of protein structure create new and diverse proteins

Most designed proteins are variations on existing proteins. It is of great interest to create *de novo* proteins that go beyond what has been invented by nature. A line of recent work has explored generative models for protein structures [20, 21, 33, 34, 35, 36]. The main challenge for a generative model is to propose stable structures that can be realized as the minimum energy state for a protein sequence, i.e. the endpoint of folding. The space of possible three-dimensional conformations of a protein sequence is exponentially large [40], but out of this set of possible conformations, most do not correspond to stable, realizable structures.

In this work we explore the use of modern variational autoencoders (VAEs) as generative models of protein structures. We find that the models can produce coherent local and global structural organization while proposing varied and diverse folds. We use AlphaFold to assess the viability of sampled sequences, finding that many sequences are predicted to fold with high confidence to their designed structures. To assess the novelty of the generated sequences, we search sequence databases including metagenomics data for homologous sequences, finding no significant matches for a large fraction of the generations.

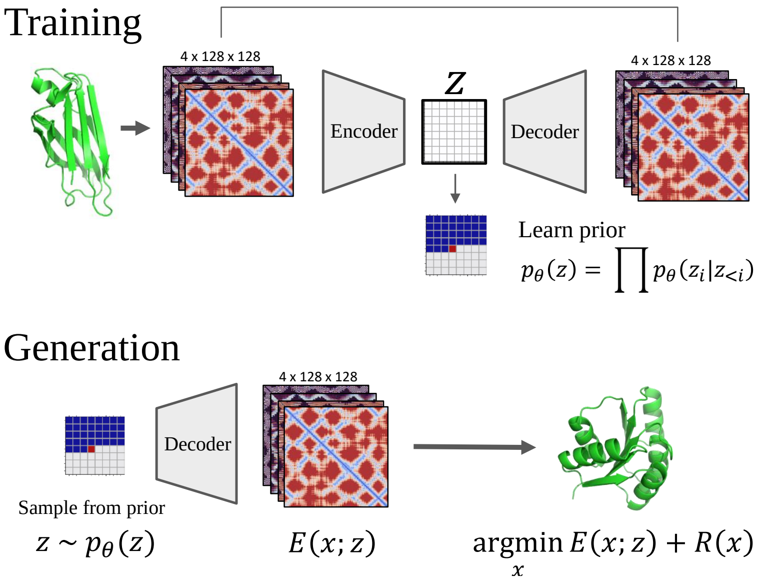

Figure 2.1: Overview of the method. Each protein structure is represented as a discretized distance map and set of angular coordinates. The model is trained to reconstruct natural protein structures. The decoder output can be interpreted as specifying an energy function over three-dimensional conformations of the protein encoding the structure at its minimum. New structures are generated by passing samples from the prior through the decoder to obtain an energy function specifying the structure. Three-dimensional coordinates are obtained by fitting the angular degrees of freedom of an idealized rigid-body representation of the protein chain to minimize the energy.

2.1 Methods

2.1.1 Overview

Figure 2.1 overviews the modeling approach. The model can be interpreted as generating a coarse set of constraints that specify the shape of the protein. The structure is implicitly encoded as the minimum of an energy over possible conformations of the protein chain. The three-dimensional coordinates can be obtained by fitting the internal angles of a rigid body representation of the protein chain to minimize the energy. In practice we can combine the energy function output by the neural network with other potentials such as the Rosetta score function. We write the structure x^* as the outcome of this minimization:

$$x^* = \arg\min_x E(x; z) + R(x) \tag{2.1}$$

Here $E(x; z)$ is the output of a decoder. Optionally $R(x)$ subsumes additional energy terms. During training an encoder and decoder are fit to natural structures to learn a discrete latent code z. To produce samples, new codes are drawn from a learned autoregressive prior $p_\theta(z)$, and passed through the decoder to obtain $E(x; z)$. The generated structure is the result of the minimization.

We compare with several other VAE modeling approaches. Structures are represented through a discretized pairwise distance map and set of angular coordinates. Pairwise representations have been useful in structure prediction [41, 42, 24], and have been recently applied to generative models of protein structures [20, 21]. We use Rosetta to perform the minimization using the all-atom `ref2015` score function [43].

2.1.2 VAE Models

We consider several VAE model variations to learn the energy $p(x|z)$. All models described use a convolutional encoder-decoder architecture unless otherwise specified. A downsampling convolutional encoder $q(z|x)$ compresses the information in the input structure x into a latent variable with a prior $p(z)$; an upsampling convolutional decoder $p(x|z)$ outputs a potential over pairwise distances and angles that reconstructs the original structure. We fix models to work on 128 length, training on a dataset of full-chain SCOP [44] domains, filtered to the length cutoff.

For baselines we use a basic encoder-decoder model, where $q(z|x)$ is a ResNet [45]. For **Conv-VAE**, $p(x|z)$ is composed of transposed convolutional blocks. For **MLP-VAE**, $p(x|z)$ is a factorized residual MLP architecture, similar to Tolstikhin et al. [46], with independent MLPs that run across the height, width, and channel dimensions. We use the classical VAE formulation [47] to maximize the Evidence Lower Bound.

HVAE is a hierarchical VAE proposed by [48, 49, 50, 51]. Although we tried up to 20 levels of latent variables, we discovered the generations were not as robust and tend to overfit, producing local artifacts without much global coherence despite near-perfect reconstruction loss. We selected a model with just 3 spatial latent variable hierarchies. $q(z|x)$ is a ResNet, and $p(x|z)$ is composed of transposed convolutional blocks.

VQ-VAE is a VQ-VAE architecture, proposed by Oord et al. [52], Razavi et al. [53]. One change we introduce is to train a transformer, similar to Ramesh et al. [54], to model the prior of the model. Here, we first train $q(z|x)$ and $p(x|z)$ by setting $z = \mathrm{VQ}(q(z|x))$, with the vector quantizer block introduced in Oord et al. [52]. Then, we train a transformer prior $p_\theta(z) = \Pi_{i=0}^{L} p_\theta(z_i|z_{<i})$, where L is the length of the prior.

VQ-VAE-BB and **HVAE-BB** are versions that learn backbone dihedral angles instead of interresidue angles. We follow the work of [55] in learning a joint phi-psi backbone angle, which then we convert into a von Mises distribution as a part of $E(x; z)$.

$E(x; z)$ is a piecewise linear energy function constructed from the distogram and binned angle outputs of $p(x|z)$ as detailed in Appendix A.1.2. We use Rosetta to minimize the energy according to Equation (2.1) and obtain a structure. Similar to Yang et al. [25], we input $E(x; z)$ as an additive energy term with `ref2015`, and find a conformation of a poly-alanine chain that minimizes the energy.

Analysis of generated structures can be found in Section 2.2. Further detail on architectural and training choices we explored is in Appendix A.1.1.

2.1.3 Evaluation of Models

Across experiments we observe that validation loss does not reflect the quality of generated structures. For example HVAE models with a large number of latent variables show better fitting and generalization when measured by loss, but are unable to generate secondary

Table 2.1: Comparison of models. Metrics computed across generated structures. $R(x) < 0$ is the percent of all generations with negative Rosetta energy. $R(x)$ is the average absolute value of Rosetta energies. "MR/LR contacts" is the average number of medium or long range contacts, divided by number of residues. "MR/LR polar" is the average number medium or long range hydrogen bonds, divided by number of residues. % helix and % sheet measures the average proportion of α-helices and β-sheets. VQ-VAE generates structures with low Rosetta energies, having long range contacts and hydrogen bonds, and containing β-sheets.

	% $R(x) <$ 0	$R(x)$ Avg	MR/LR contacts	MR/LR polar	% α-helix	% β-sheets
MLP-VAE	0.0043	0.7301	1.5033	0.0329	0.0487	0.0138
Conv-VAE	0.0223	0.3408	1.5426	0.0277	0.1273	0.0154
HVAE	0.0817	0.3526	1.1066	0.0291	0.3229	0.0169
HVAE-BB	**0.3127**	**0.0612**	1.4830	0.0312	**0.4194**	0.0161
VQ-VAE	0.2140	0.2066	1.6849	0.0602	0.3387	0.0720
VQ-VAE-BB	0.1955	0.1671	**1.9890**	**0.0879**	0.2677	**0.1080**

structure elements. As a result we use the Rosetta score function for model selection. We evaluate models using a variety of metrics, including the Rosetta score function values, medium and long-range contacts, hydrogen bonds, and secondary structure elements.

We use AlphaFold [3] to predict whether the generated structures can be realized as the endpoint of folding. We design sequences for the structures using Rosetta `FastDesign`. Since `FastDesign` is a monte-carlo procedure, we generate up to 200 designed sequences, and use the one with lowest `ref2015` score. We use the `reduce_db` option provided by Alphafold to fold the generated sequences.

2.2 Structure Generation Experiments

Here we focus on evaluating structures sampled from the model according to Figure 2.1. We will study the ability to design sequences that fold to those structures in Section 2.3.

Qualitative inspection Figure A.2 presents sampled distograms for each model. Samples from the single latent variable models are not sharp. HVAE and VQ-VAE generate

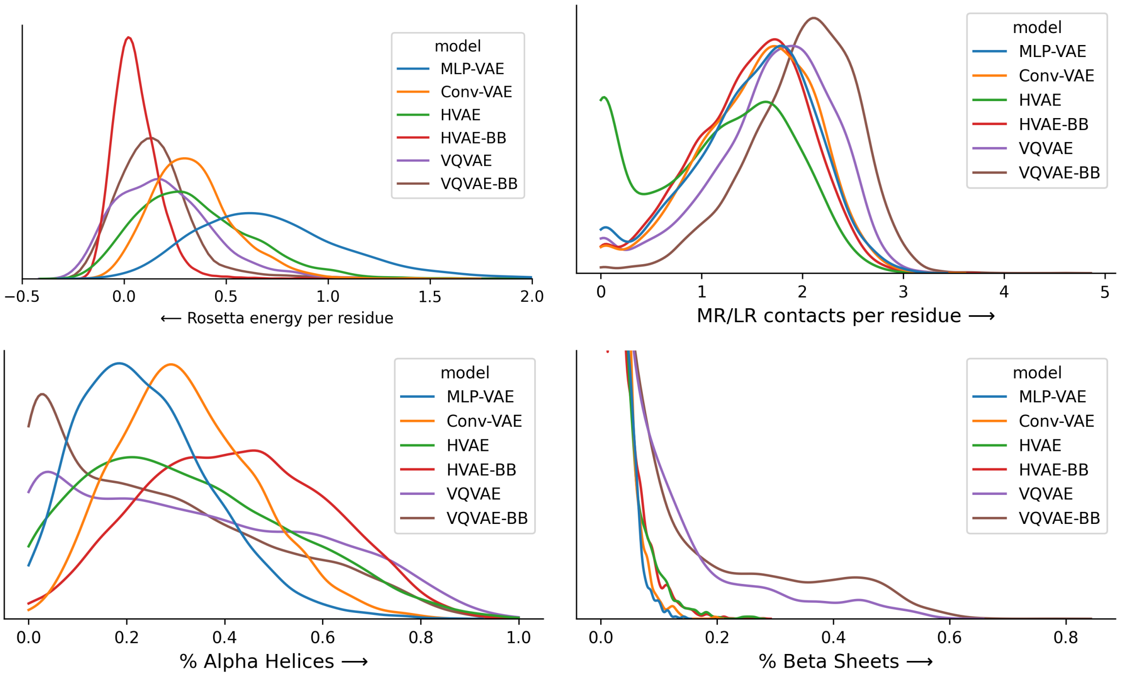

Figure 2.2: Gaussian kernel density estimate plot of the proposed metrics, for structures generated with different models.

qualitatively sharper and more distinct distograms. To benchmark model fidelity, we generate 3000 samples from $E(x;z)$, with z sampled from the model's respective priors. Figure A.3 shows the structures resulting from energy minimization using the generated distrograms. We observe that the baseline models only generate α-helix secondary structure elements, and do not have realistic backbone angles. HVAE appears to mostly generate coils and trivial α-helix bundles. VQ-VAE is the only model that successfully generates β-sheets with high-fidelity.

Quantitative metrics of Rosetta-folded structures Table 2.1 summarizes data on the quality of generated structures. The metrics measured are as follows. $R(x)$ is the value of Rosetta's `ref2015` energy after the constrained folding procedure. $\%R(x) < 0$ is the percent of all generations with negative Rosetta energy. In general, the more negative the energy, the better. Structures with positive energy tend to not be well formed, exhibiting pathologies such as physically intersecting amino acids and disordered secondary structure.

We define a Medium or Long Range (MR/LR) contact to be when two residues σ_i, σ_j, have sequence separation in the amino acid chain $|i - j| > 12$ and a Cβ-Cβ distance of less than 8Å. We then normalize this by the length of the protein and average to obtain the MR/LR contact score. The MR/LR polar contacts similar to Cβ-Cβ contacts, except that we use Kunzmann and Hamacher [56] to find hydrogen bonds via the Baker-Hubbaard algorithm [57] on the generated polyalanine chain. Both of these metrics are a measure of how compact and well-structured the generated proteins are.

Finally, % α-helix and % β-sheet measures the average proportion formed in generated structures. The categories are based on secondary structure assignment with DSSP [58].

Table 2.1 and Figure 2.2 compare the models. Large differences in the quality of generations are observed favoring the VQ-VAE models. Although HVAE-BB seem to be best in terms of energy favorable structures, a large proportion of its generated structures are α-helices. For VQ-VAE-BB, we see a large increase in the number of MR/LR contacts and long range hydrogen bonds. The results suggest that VQ-VAE produces more realistic generations.

Best structures The above metrics can also be used to filter the generations to identify candidates that are most likely to be designable. Figure A.1 and Figure 2.3 present filtered results. We filter with $R(x) < 0$ and MR/LR contacts > 1 and show random generations that pass this filter. Even after filtering for the highest quality generations, each of the baselines retrieve mostly only α-helices as the best configurations. VQ-VAE is the only model that can generate β-sheets.

Novelty and diversity Figure 2.4 evaluates the diversity and novelty of structures generated by VQ-VAE-BB. First we compare the generated structures to each other. We compute pairwise TM-scores between 180 generated structures. The generated structures cluster by secondary structure into Mainly Alpha, Mainly Beta, and Alpha/Beta folds. We classify a structure into Mainly Alpha and Mainly Beta if the proportion of β-sheets and

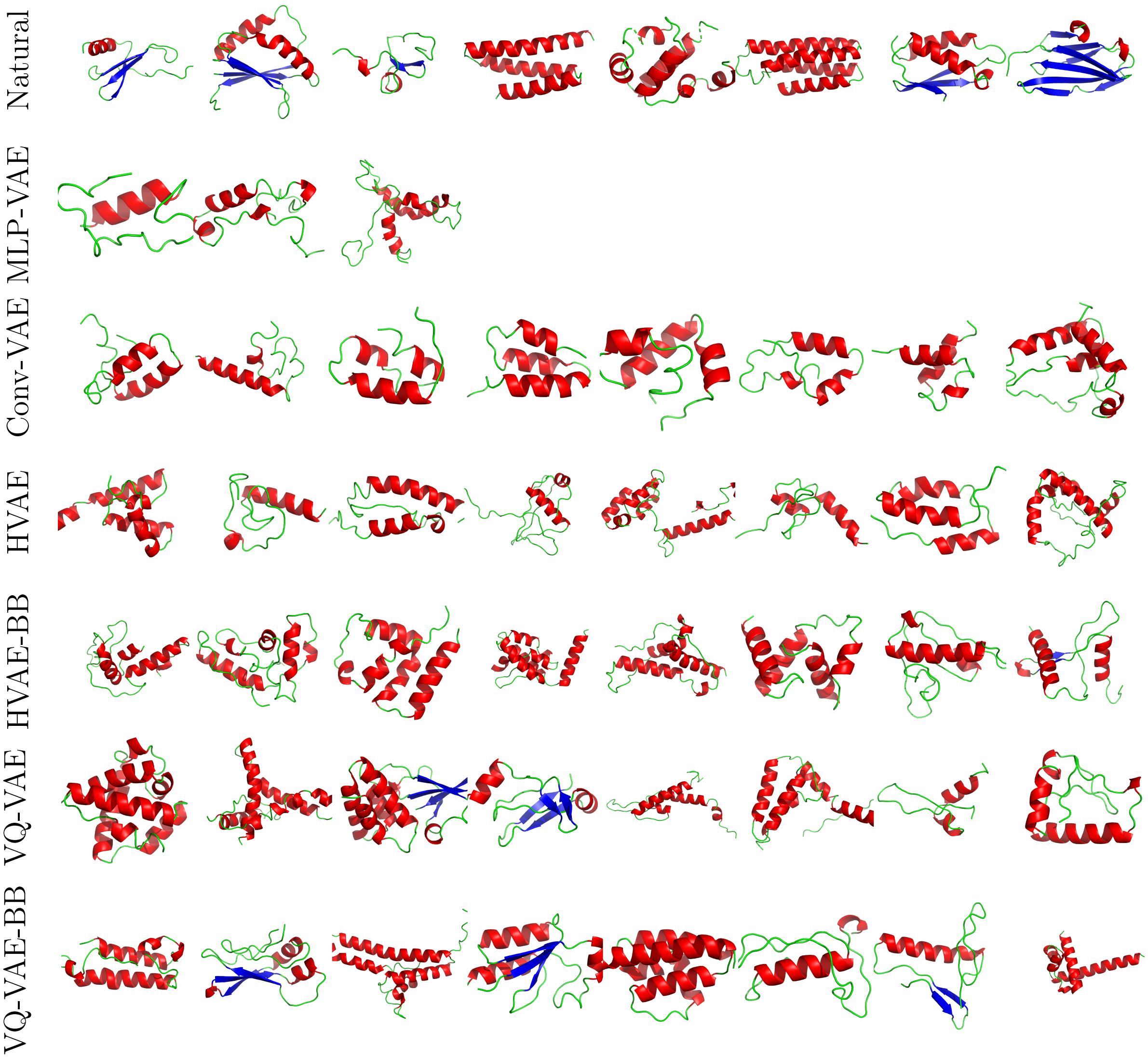

Figure 2.3: Top Structures: Corresponding to Figure A.1, we show how each of the distograms fold under constrained folding. These structures are selected to have negative Rosetta energy and > 1 MR/LR contacts per residue We color α-helices red, β-sheets blue, and coils green. Only 3 such structures out of 3000 passed such a filter for MLP-VAE. Only VQ-VAE style models are able to generate many β-sheets.

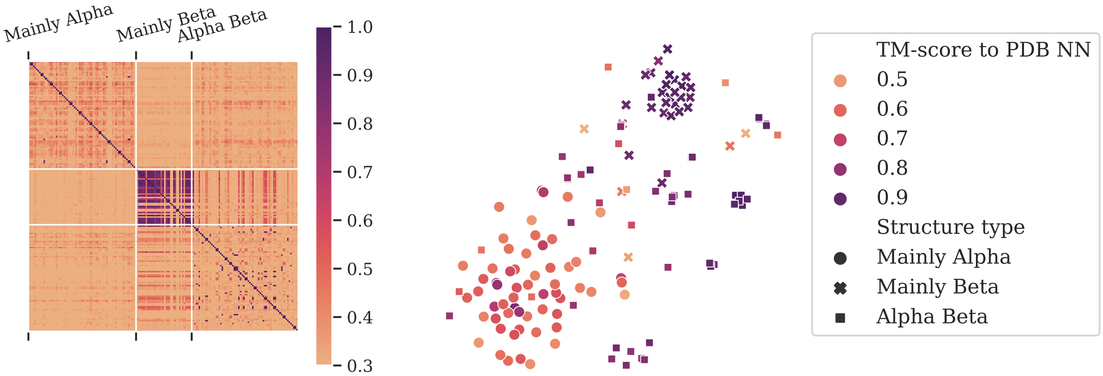

Figure 2.4: Diversity of structures. (Left) Pairwise TM-scores between 180 generated structures. The majority of pairwise TM-scores are lower than the value 0.5 which roughly corresponds to the same fold [1]. (Right) t-SNE [2] plot of the structural space covered by the same proteins. The proteins clearly cluster by category. Each category has structures with higher and lower TM-score to its nearest neighbor in PDB (color-code), indicating the generated structures cover a range of similarity with natural structures.

α-helices is below 5% respectively, and Alpha/Beta otherwise. The majority of pairwise TM-scores are below 0.5 indicating that the model produces a diverse set of folds. We also perform a structural comparison for each of the generated structures against the entire PDB. We find that the model generates diverse protein structures with a range of structural similarity to natural proteins, except in the Mainly Beta category where the generations have higher similarity.

2.3 Protein Design Experiments

We ask whether generated structures can be realized by an amino acid sequence as the endpoint of folding. We design sequences for 40 structures and use AlphaFold to predict their structures.

We select 40 random structures for sequence design after filtering the raw generations. Approximately 5% of generations have $R(x) < -0.1$, MR/LR contacts more than 1.5 per residue, and coils consisting of less than half the protein. 40 random structures are sampled that meet this criteria. 200 sequences for each structure are sampled using `FastDesign` and the lowest energy sequence is selected. We run the default AlphaFold pipeline, and select the model with highest predicted LDDT.

AlphaFold produces high confidence models (> 0.7 LDDT) for 9 of the designed sequences. These proteins are shown in Figure 2.5. Out of this set, 8 of AlphaFold predictions had low (maximum 2.6A) RMSD to their generated structure. The last row is the design where AlphaFold disagrees with the generated structure. Several designs are β-barrels having similar distograms. We note that although the argmax distograms look similar, the structures encoded at their minima are different enough to produce different nearest neighbors in PDB for each of the structures.

Most of the designed sequences do not have homologous sequences in UniRef90 or MGnify [17, 59]. We run JackHMMER [60] against Uniref90 and MGnify and find no significant sequence matches for 5 designed structures, matches with very low sequence identity (15%) for 2 designed structures, and 75 matches with maximum sequence identity of 26% for the final structure. To confirm this finding, we also run `FastDesign` against 40 randomly selected natural protein chains between length 64-128. Sequence designs for natural structures result in many more JackHMMER hits against UniRef90, with a higher degree of sequence similarity, shown in Figure 2.6. These results suggest that the model generates novel proteins.

2.4 Related Work

Deep generative models One of the major goals in deep learning has been to represent complex distributions over high dimensional data with a rich hierarchical models [61, 62, 63]. Notable successes have been achieved using deep generative models with latent variables such as Generative Adversarial Networks (GANs) [64] and Variational Auto-Encoders (VAEs) [47]. Vector Quantized VAE (VQ-VAE) [52, 53] finds that discretizing the latent variables

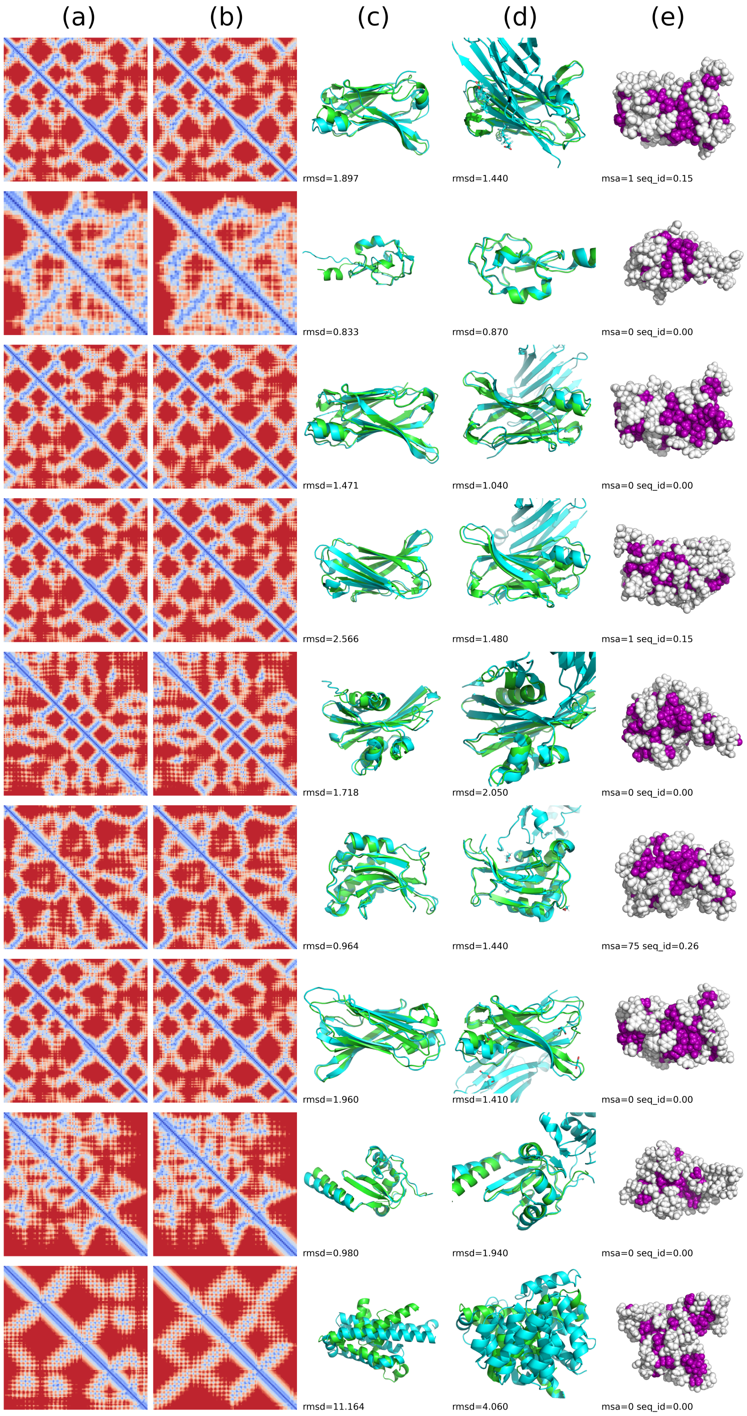

Figure 2.5: Each row is an individual generation, selected for AlphaFold modeling confidence. (a) $\arg\max$ of the distogram proposed by our model. (b) Distogram of the AlphaFold folded structure. (c) Superposition of designed structure (green) with AlphaFold prediction. (d) Superposition of designed structure (green) with nearest match in PDB by TM-score (teal), using the transformation matrix from TMalign. (e) Hydrophobic residues in purple, hydrophilic in white, proteins exhibit hydrophobic core. Most generations have few homologs, and low sequence identity with closest homolog.

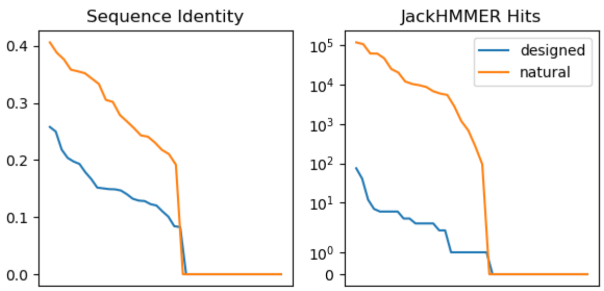

Figure 2.6: Comparison of `FastDesign` sequences for natural proteins vs generations. (Left) Sequence identity of nearest match; (Right) Log scale number of matches for each protein. Sequence identity is computed with respect to the closest sequence match. Approximately half of the natural and generated proteins do not have any nearby sequences in UniRef and MGnify. However `FastDesign` sequences for natural proteins have many more sequence matches. Additionally, the sequence identity of the nearest matches are much higher for natural proteins than generated proteins.

enables higher fidelity generations. Vahdat and Kautz [50] and Sønderby et al. [48], Child [51] suggest that hierarchies are an important part of latent variable models.

Generative models for protein structures While there has been breakthrough progress in deep learning for prediction of structure from sequence [65, 24, 55, 25, 3], in comparison, little work has been done on generative modeling of protein structures.

Initial work explored using latent variable generative models such as GANs and VAEs [20, 35, 36, 21]. In an alternative approach, Anishchenko et al. [34] proposes inverting a supervised structure prediction model to generate structures and sequences jointly. Previous latent variable generative models have not been able to produce globally coherent structures when trained to model highly diverse sets of protein folds [20], and have been most effective when trained on a focused set of highly related structures such as immonoglobulins [21]. In this work we develop latent variable models that are able to generate diverse and coherent protein structures.

Additional Generative models have also been explored for protein sequences. VAEs on amino acid sequences have been applied in several problem settings; VAEs and other deep generative models can capture the sequence diversity of specific protein families without explicitly conditioning on structure information [66, 67, 68, 69, 70, 33]. A related problem setting is designing protein sequences explicitly conditioned on a backbone structure as in [71, 22, 72, 73].

2.5 Conclusions

We perform a systematic study of deep generative models for designing protein structures. We find that generative models are able to capture local secondary structure as well as globally coherent fold topology, when trained on a diverse set of protein domains. We find that generated structures can be realized by sequence designs that are predicted to correctly fold with high confidence by AlphaFold. Although the generations are structurally similar to existing folds, sequences designed to realize them are often novel. This suggests that neural generative models are capable of generalizing beyond simply reproducing natural proteins.

Acknowledgements

Anonymized for the reviewing process.

3 | Learning inverse folding from millions of predicted structures

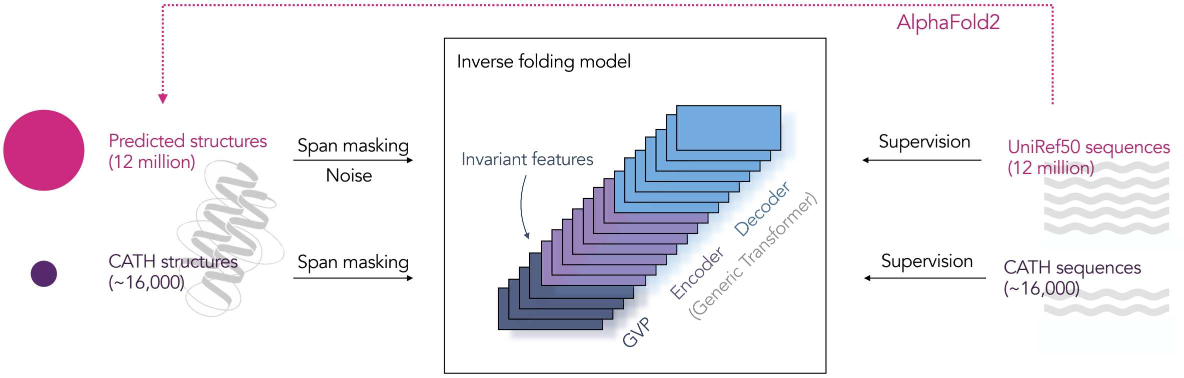

Figure 3.1: Augmenting inverse folding with predicted structures. To evaluate the potential for training protein design models with predicted structures, we predict structures for 12 million UniRef50 protein sequences using AlphaFold2 [3]. An autoregressive inverse folding model is trained to perform fixed-backbone protein sequence design. Train and test sets are partitioned at the topology level, so that the model is evaluated on structurally held-out backbones. We compare transformer models having invariant geometric input processing layers, with fully geometric models used in prior work. Span masking and noise is applied to the input coordinates.

3.1 Introduction

Designing novel amino acid sequences that encode proteins with desired properties, known as *de novo protein design*, is a central challenge in bioengineering [74]. The most well-established approaches to this problem use an energy function which directly models the physical basis of a protein's folded state [19].

Recently a new class of deep learning based approaches has been proposed, using generative models to predict sequences for structures [22, 72, 75, 23], generate backbone structures [20, 21], jointly generate structures and sequences [34, 76], or model sequences directly [16, 77, 78, 79, 80, 14]. The potential to learn the rules of protein design directly from data makes deep generative models a promising alternative to current physics-based energy functions.

However, the relatively small number of experimentally determined protein structures places a limit on deep learning approaches. Experimentally determined structures cover less than 0.1% of the known space of protein sequences. While the UniRef sequence database [17] has over 50 million clusters at 50% sequence identity; as of January 2022, the Protein Data Bank (PDB) [81] contains structures for fewer than 53,000 unique sequences clustered at the same level of identity.

Here we explore whether predicted structures can be used to overcome the limitation of experimental data. With progress in protein structure prediction [3, 82, 41], it is now possible to consider learning from predicted structures at scale. Predicting structures for the sequences in large databases can expand the structural coverage of protein sequences by orders of magnitude. To train an inverse model for protein design, we predict structures for 12 million sequences in UniRef50 using AlphaFold2.

We focus on the problem of predicting sequences from backbone structures, known as *inverse folding* or fixed backbone design. We approach inverse folding as a sequence-to-sequence problem [22], using an autoregressive encoder-decoder architecture, where the model is tasked with recovering the native sequence of a protein from the coordinates of its backbone atoms.

We make use of the large number of sequences with unknown structures by adding them as additional training data, conditioning the model on predicted structures when the experimental structures are unknown (Figure 3.1). This approach parallels back-translation [83, 84] in machine translation, where predicted translations in one direction are used to improve a model in the opposite direction. Back-translation has been found to effectively learn from extra target data (i.e. sequences) even when the predicted inputs (i.e. structures) are of low quality.

We find that existing approaches have been limited by data. While current state-of-the-art inverse folding models degrade when training is augmented with predicted structures, much larger models and different model architectures can effectively learn from the additional data, leading to an improvement of nearly 10 percentage points in the recovery of sequences for

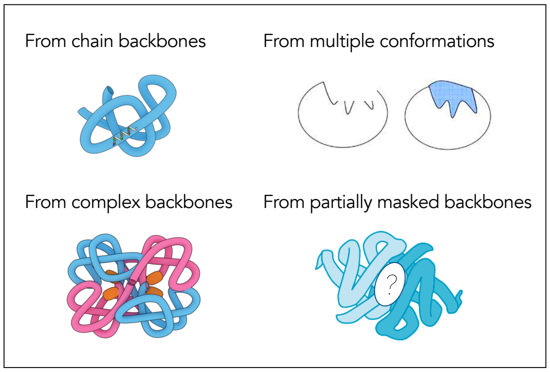

Figure 3.2: Illustration of the protein design tasks considered.

structurally held out native backbones.

We evaluate models on fixed backbone design benchmarks from prior work, and assess the generalization capabilities across a series of tasks including design of complexes and binding sites, partially masked backbones, and multiple conformations. We further consider the use of the models for zero-shot prediction of mutational effects on protein function and stability, complex stability, and binding affinity.

3.2 Learning inverse folding from predicted structures

The goal of inverse folding is to design sequences that fold to a desired structure. In this work, we focus on the backbone structure without considering side chains. While each of the 20 amino acid has a specific side chain, they share a common set of atoms that make up the amino acid backbone. Among the backbone atoms, we choose the N, Cα (alpha Carbon), and C atom coordinates to represent the backbone.

Using the structures of naturally existing proteins we can train a model for this task by supervising it to predict the protein's native sequence from the coordinates of its backbone atoms in three-dimensional space. Formally we represent this problem as one of learning the conditional distribution $p(Y|X)$, where for a protein of length n, given a sequence X of spatial coordinates $(x_1, \dots, x_i, \dots, x_{3n})$ for each of the backbone atoms *N, Cα, C* in the structure, the objective is to predict Y the native sequence $(y_1, \dots, y_i, \dots, y_n)$ of amino acids. This density is modeled autoregressively through a sequence-to-sequence encoder-decoder:

$$p(Y|X) = \prod_{i=1}^{n} p(y_i|y_{i-1}, \dots, y_1; X) \tag{3.1}$$

We train a model by minimizing the negative log likelihood of the data. We can design sequences by sampling, or by finding sequences that maximize the conditional probability given the desired structure.

3.2.1 Data

Predicted structures We generate 12 million structures for sequences in UniRef50 to explore how predicted structures can improve inverse folding models. To select sequences for structure prediction we first use MSA Transformer [85] to predict distograms for MSAs of all UniRef50 sequences. We rank the sequences by distogram LDDT scores [55] as a proxy for the quality of the predictions. We take the top 12 million sequences not longer than five hundred amino acids and forward fold them using the AlphaFold2 model with a final Amber [86] relaxation. This results in a predicted dataset approximately 750 times the size of the training set of experimental structures (Appendix A.2.1.1).

Training and evaluation data We evaluate models on a structurally held-out subset of CATH [87]. We partition CATH at the topology level with an 80/10/10 split resulting in 16153 structures assigned to the training set, 1457 to the validation set, and 1797 to the test set. Particular care is required to prevent leakage of information in the test set via the predicted structures. We use Gene3D topology classification [88] to filter both the sequences used for supervision in training, as well as the MSAs used as inputs for AlphaFold2 predictions (Appendix A.2.1.1). We also perform evaluations on a smaller subset of the CATH test set that has been additionally filtered by TM-score using Foldseek [5] to exclude any structures with similarity to those in the training set (Appendix A.2.2).

3.2.2 Model architectures

We study model architectures using Geometric Vector Perceptron (GVP) layers [23] that learn rotation-equivariant transformations of vector features and rotation-invariant transformations of scalar features.

We present results for three model architectures: (1) GVP-GNN from Jing et al. [23] which is currently state-of-the-art on inverse folding; (2) a GVP-GNN with increased width and depth (GVP-GNN-large); and (3) a hybrid model consisting of a GVP-GNN structural encoder followed by a generic transformer (GVP-Transformer). All models used in evaluations are trained to convergence, with detailed hyperparameters listed in Table A.1.

In inverse folding, the predicted sequence should be independent of the reference frame of the structural coordinates. For any rotation and translation T of the input coordinates, we would like for the model's output to be invariant under these transformations, i.e., $p(Y|X) = p(Y|TX)$. Both the GVP-GNN and GVP-Transformer inverse folding models studied in this work are invariant (Appendix A.2.1.3).

GVP-GNN We start with the GVP-GNN architecture with 3 encoder layers and 3 decoder layers as described in [23], with the vector gates described in [13] (GVP-GNN, 1M parameters). As inputs to GVP-GNN, protein structures are represented as proximity graphs where each amino acid corresponds to a node in the graph. The node features are a combination of scalar node features derived from dihedral angles and vector node features derived from the relative positions of the backbone atoms, while the edge features capture the relative positions of nearby amino acids.

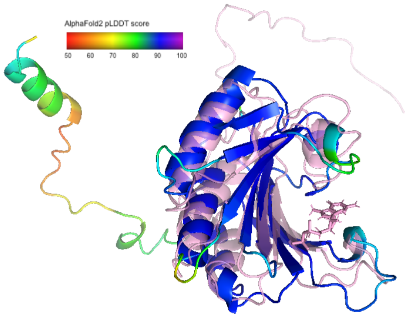

Figure 3.3: Example AlphaFold prediction compared with experimental structure for a UniRef50 sequence (UniRef50: P07260; PDB: 1AP8). The experimental structure is shown as pink with transparency. The prediction is coloured by the pLDDT confidence score, with blue in high-confidence regions.

When trained on predicted structures, we find a deeper and wider version of GVP-GNN with 8 encoder layers and 8 decoder layers (GVP-GNN-large, 21M parameters) performs better. Scaling GVP-GNN further did not improve model performance in preliminary experiments (Figure 3.6c).

GVP-Transformer We use GVP-GNN encoder layers to extract geometric features, followed by a generic autoregressive encoder-decoder Transformer [89]. In GVP-GNN, the input features are translation-invariant and each layer is rotation-equivariant. We perform a change of basis on the vector features from GVP-GNN into local reference frames defined for each amino acid to derive rotation-invariant features (Appendix A.2.1.3). In ablation studies increasing the number of GVP-GNN encoder layers improves the overall model performance (Figure A.6), indicating that the geometric reasoning capability in GVP-GNN is complementary to the Transformer layers. Scaling improves performance up to a 142M-parameter GVP-Transformer model with 4 GVP-GNN encoder layers, 8 generic Transformer encoder layers, and 8 generic Transformer decoder layers (Figure 3.6c).

3.2.3 Training

Combining experimental and predicted data During training, in each epoch we mix the training set of experimentally derived structures (∼16K structures) with a 10% random sample of the AlphaFold2-predicted training set (10% of 12M), resulting in a 1:80 experimental:predicted data ratio. For larger models, a high ratio of predicted data during training helps prevent overfitting on the smaller experimental train set (Figure 3.6b). While adding predicted data improves performance, training only on predicted data leads to substantially worse performance (Table A.4).

The loss is equally weighted for each amino acid in target sequences. We mask out predicted input coordinates with AlphaFold2 confidence score (pLDDT) below 90, around 25% of the predicted coordinates. See Figure 3.3 for visualization of the pLDDT confidence score. Most often these low confidence regions are at the start and the end of sequences and may correspond to disordered regions. We prepend one token at the beginning of each sequence to indicate whether the structure is experimental or predicted. For each residue we provide the pLDDT confidence score from AlphaFold2 as a feature encoded by Gaussian radial basis functions.

Adding Gaussian noise at the scale of 0.1 angstroms to the predicted structures during training slightly improves performance (Table A.3). This finding is consistent with Edunov et al. [84], who observe that backtranslation with sampled or noisy synthetic data provides a stronger training signal than maximum a posteriori (MAP) predictions.

Span masking To enable sequence design for partially masked backbones, we introduce backbone masking during training. We experiment with both independent random masking and span masking. In natural language processing, span masking improves performance over random masking [90]. We randomly select continuous spans of up to 30 amino acids until 15% of input backbone coordinates are masked. The communication patterns in the geometric

		Perplexity			Recovery %		
Model	Data	Short	Single-chain	All	Short	Single-chain	All
Natural frequencies		18.12	18.03	17.97	9.6%	9.0%	9.5%
Structured GNN	CATH	7.91	6.48	6.49	31.5%	37.1%	37.1%
GVP-GNN	CATH	7.14	5.36	5.43	34.0%	42.7%	42.2%
	+ AlphaFold2	8.55	6.17	6.06	29.5%	38.2%	38.6%
GVP-GNN-large	CATH	7.68	6.12	6.17	32.6%	39.4%	39.2%
	+ AlphaFold2	6.11	4.09	4.08	**38.3%**	50.8%	50.8%
GVP-Transformer	CATH	8.18	6.33	6.44	31.3%	38.5%	38.3%
	+ AlphaFold2	**6.05**	**4.00**	**4.01**	38.1%	**51.5%**	**51.6%**

Table 3.1: Fixed backbone sequence design. Evaluation on the CATH 4.3 topology split test set. Models are compared on the basis of per-residue perplexity (lower is better; lowest perplexity bolded) and sequence recovery (higher is better; highest sequence recovery bolded). Large models can make better use of the predicted UniRef50 structures. The best model trained with predicted structures (GVP-Transformer) improves sequence recovery by 8.9 percentage points over the best model (GVP-GNN) trained on CATH only.

layers are adapted to account for masking with details in Appendix A.2.1.2. Span masking improves the performance of GVP-Transformer both on unmasked backbones (Table A.3) and on masked regions (Figure 3.4).

3.3 Results

We evaluate models across a variety of benchmarks in two overall settings: fixed backbone sequence design and zero-shot prediction of mutation effects. For fixed backbone design, we start with evaluation in the standard setting [22, 23] of sequence design given all backbone coordinates. Then, we make the sequence design task more challenging along three dimensions: (1) introducing masking on coordinates; (2) generalization to protein complexes; and (3) conditioning on multiple conformations. Additionally, we show that inverse folding models are effective zero-shot predictors for protein complex stability, binding affinity, and insertion effects.

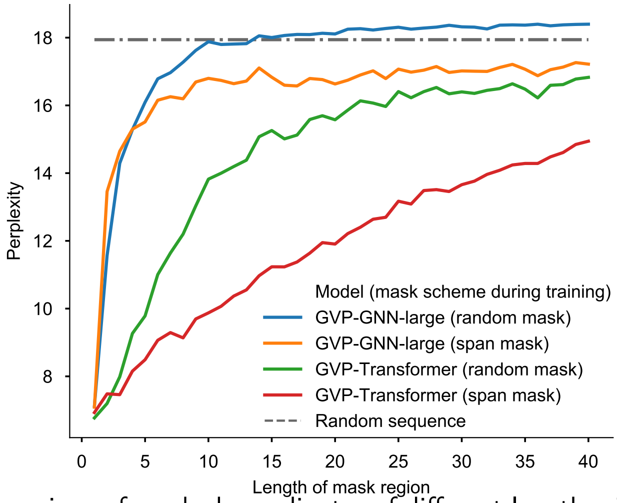

Figure 3.4: Perplexity on regions of masked coordinates of different lengths. The GVP-GNN architecture degrades to the perplexity of the background distribution for masked regions of more than a few tokens, while GVP-Transformer maintains moderate accuracy on long masked spans, especially when trained on masked spans.

3.3.1 Fixed backbone protein design

We begin with the task of predicting the native protein sequence given its backbone atom (N, Cα, C) coordinates. Perplexity and sequence recovery on held-out native sequences are two commonly used metrics for this task. Perplexity measures the inverse likelihood of native sequences in the predicted sequence distribution (low perplexity for high likelihood). Sequence recovery (accuracy) measures how often sampled sequences match the native sequence at each position. To maximize sequence recovery, the predicted sequences are sampled with low temperature $T = 1\text{e}{-6}$ from the model. While the model is calibrated (Figure A.10), a lower temperature results in sequences with higher likelihoods (and hence typically higher sequence recovery) and lower diversity. Empirically at temperature as low as $1\text{e}{-6}$ the sampling is almost deterministic. Table 3.1 compares models using the perplexity and sequence recovery metrics on the structurally held-out backbones.

We observe that current state-of-the-art inverse folding models are limited by the CATH training set. Scaling the current 1M parameter model (GVP-GNN) to 21M parameters

(GVP-GNN-large) on the CATH dataset results in overfitting with a degradation of sequence recovery from 42.2% to 39.2% (Table 3.1). On the other hand, the current model at the 1M parameter scale cannot make use of the predicted structures: training GVP-GNN with predicted structures results in a degradation to 38.6% sequence recovery (Table 3.1), with performance worsening with increasing numbers of predicted structures in training (Figure 3.6a).

Larger models benefit from training on the AlphaFold2-predicted UniRef50 structures. Training with predicted structures increases sequence recovery from 39.2% to 50.8% for GVP-GNN-large and from 38.3% to 51.6% for GVP-Transformer over training only on the experimentally derived structures. The improvements are also reflected in perplexity. Similar improvements are observed on the test subset filtered by TM-score (Table A.2). The best model trained with UniRef50 predicted stuctures, GVP-Transformer, improves sequence recovery by 9.4 percentage points over the best model, GVP-GNN, trained on CATH alone.

As there are many sequences that can fold to approximately the same structure, even an ideal protein design model will not have 100% native sequence recovery. We observe that the GVP-GNN-large and GVP-Transformer models are well-calibrated (Figure A.10). The substitution matrix between native sequences and model-designed sequences resembles the BLOSUM62 substitution matrix (Figure A.9), albeit noticeably sparser for the amino acid Proline.

When we break down performance on core residues and surface residues, as expected, core residues are more constrained and have a high native sequence recovery rate of 72%, while surface residues are not as constrained and have a lower sequence recovery of 39% (Figure 3.5; top). Generally perplexity increases with the solvent accessible surface area (Figure 3.5; bottom). Despite the lower sequence recovery on the surface, sampled sequences do tend not to have hydrophobic residues on the surface (Figure A.11).

As an example of inverse folding of a structurally-remote protein, we re-design the receptor

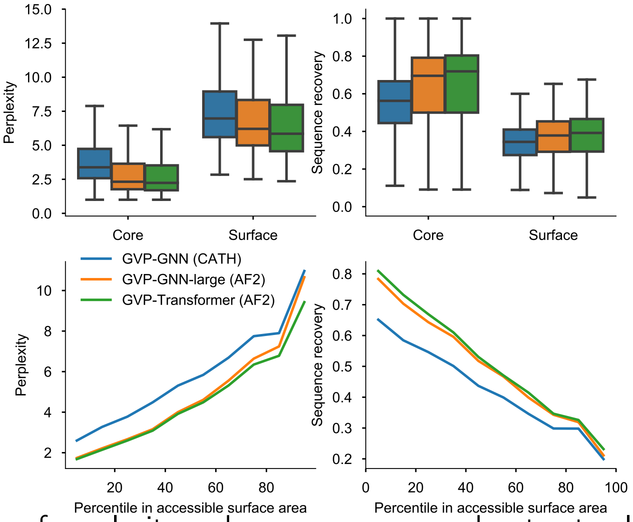

Figure 3.5: Comparison of perplexity and sequence recovery by structural context according to two different measures: number of neighbors (top) and solvent accessible surface area (bottom). Top: Breakdown for core and surface residues. Residues are categorized by density of neighboring Cα atoms within 10A of the central residue Cα atom (core: ≥ 24 neighbors; surface: < 16 neighbors). Each box shows the distribution of perplexities for the core or surface residues across different sequences. Bottom: Perplexity and sequence recovery as a function of solvent accessible surface area. Increased sequence recovery for buried residues suggests the model learns dense hydrophobic packing constraints in the core.

binding domain (RBD) sequence of the SARS-CoV-2 spike protein (PDB: 6XRA and 6VXX; illustrated in Figure A.8) with the two models. The SARS-CoV-2 spike protein has no match to the training data with TM-score above 0.5. Both GVP-GNN and GVP-Transformer achieve high sequence recovery (49.7% and 53.6%) for the native RBD sequence (Table A.6). See Table A.10 for a list of randomly sampled sequence designs.

While perplexity and sequence recovery are informative metrics, low perplexity and high sequence recovery do not necessarily guarantee structural similarity. One empirically observed failure mode in sampled sequences is repetition of the same amino acid, e.g. EEEEEEE. It would be interesting to further identify more failure modes by studying the experimental or predicted structures of sampled sequences.

Partially-masked backbones We evaluate the models on partial backbones. While masking during training does not significantly change test performance on unmasked back-

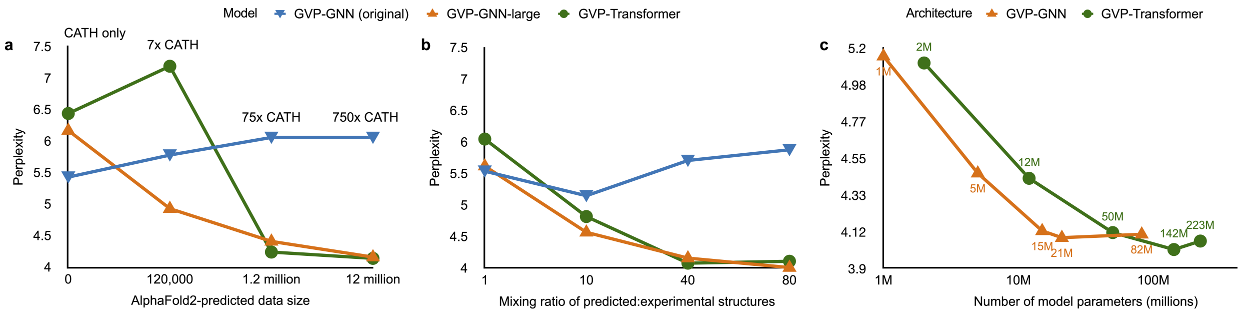

Figure 3.6: Ablation studies on training data. (a) Effect of increasing the number of predicted structures. The original GVP-GNN degrades with training on additional data, but GVP-GNN-large and GVP-Transformer improve with increasing numbers of predicted structures. (b) Effect of increasing the mixing ratio during training between predicted and experimental structures. A higher ratio of predicted structures improves performance for both GVP-GNN-large and GVP-Transformer. (c) GVP-GNN and GVP-Transformer model size.

	Perplexity	
Model	Chain	Complex
Natural frequencies	17.93	
GVP-GNN	7.80	5.37
GVP-GNN-large+AF2	**6.32**	3.90
GVP-Transformer+AF2	**6.32**	**3.81**

Table 3.2: Sequence design performance on complexes in the CATH topology test split when given the backbone coordinates of only a chain ("Chain" column) and when given all backbone coordinates of the complex ("Complex" column). The perplexity is evaluated on the same chain in the complex for both columns.

bones (Table A.3), masking does enable models to non-trivially predict sequences for mask regions. Although GVP-GNN-large has low perplexity on short-length masks, its performance quickly degrades to the perplexity of the background distribution on masks longer than 5 amino acids (Figure 3.4). By contrast, the GVP-Transformer model maintains moderate performance even on longer masked regions, with less degradation if trained with span masking instead of independent random masking (Figure 3.4).

Protein complexes Although the training data only consists of single chains, we find that models generalize to multi-chain protein complexes. We represent complexes by concatenating the chains together with 10 mask tokens between chains, and place the target chain for

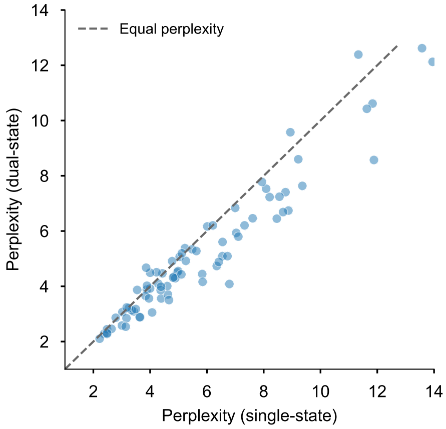

Figure 3.7: Dual-state design. GVP-Transformer conditioned on two conformations results in lower sequence perplexity at locally flexible residues than single-conformation conditioning for structurally held-out proteins in PDBFlex (see Appendix A.2.3 for details).

sequence design at the beginning during concatenation. We include all complexes in the CATH 4.3 test set up to 1000 amino acids in length. For chains that are part of a protein complex, there is a substantial improvement in perplexity of both models when given the full complex coordinates as input, versus only the single chain (Table 3.2 and Figure A.7), suggesting that both GVP-GNN and GVP-Transformer can make use of inter-chain information from amino acids that are close in 3D structure but far apart in sequence.

Multiple conformations Multi-state design is of interest for engineering enzymes and biosensors [91, 92]. Some proteins exist in multiple distinct folded forms in equilibrium, while other proteins may exhibit distinct conformations when binding to partner molecules.

For a backbone X, the inverse folding model predicts a conditional distribution $p(Y|X)$ over possible sequences Y for the backbone. To design a protein sequence compatible with two states A and B, we would like find sequences with high likelihoods in the conditional distributions $p(Y|A)$ and $p(Y|B)$ for each state. We use the geometric average of the two conditional likelihoods as a proxy for the desired distribution $p(Y|A, B)$ conditioned on the sequence being compatible with both states.

We compare single-state and multi-state sequence design performance on 87 test split

	Spearman correlation			
Model	No coords	No RBM coords	No ACE2 coords	All coords
ESM-1v	0.03			
ESM-1b	0.02			
ESM-MSA-1b (few-shot)	**0.51**			
GVP-GNN		-0.10	0.50	0.60
GVP-GNN-large+AF2		-0.05	0.52	**0.69**
GVP-Transformer+AF2		-0.06	**0.53**	0.64

Table 3.3: Zero-shot performance on binding affinity prediction for the receptor binding domain (RBD) of SARS-CoV-2 Spike, evaluated on ACE2-RBD mutational scan data [9]. The zero-shot predictions are based on the sequence log-likelihood for the receptor binding motif (RBM), which is the portion of the RBD in direct contact with ACE2 [10]. We evaluate in four settings: 1) Given sequence data alone ("No coords"); 2) Given backbone coordinates for both ACE2 and the RBD but excluding the RBM and without sequence ("No RBM coords"); 3) Given the full backbone for the RBD but no information for ACE2 ("No ACE2 coords"); and 4) Given all coordinates for the RBD and ACE2.

proteins with multiple conformations in the PDBFlex dataset [93]. On locally flexible residues, multi-state design results in lower sequence perplexity than single-state design (Figure 3.7). See Appendix A.2.3 for more details on the PDBFlex data.

3.3.2 Zero-shot predictions

We next show that inverse folding models are effective zero-shot predictors of mutational effects across practical design applications, including prediction of complex stability, binding affinity, and insertion effects. To score the effect of a mutation on a particular sequence, we use the ratio between likelihoods of the mutated and wildtype sequences according to the inverse folding model, given the experimentally determined wildtype structure. Exact likelihood evaluations are possible from both GVP-GNN and GVP-Transformer as they are both based on autoregressive decoders. We then compare these likelihood ratio scores to experimentally-determined fitness values measured on the same set of sequences.

De novo mini-proteins Rocklin et al. [11] performed deep mutational scans across a set of *de novo* designed mini-proteins with 10 different folds measuring the stability in response to point mutations. The likelihoods of inverse folding models have been shown to correlate with experimentally measured stability using this dataset [22, 23]. We evaluate the GVP-Transformer and GVP-GNN-large models on the same mutational scans, and observe improvements in stability predictions from using predicted structures as training data for 8 out of 10 folds in the dataset (Table A.5). Further details are in Appendix A.2.3.

Complex stability We evaluate models on zero-shot prediction of mutational effects on protein complex interfaces, using the Atom3D benchmark [12] which incorporates binding free energy changes in the SKEMPI database [94] as a binary classification task. We find that sequence log-likelihoods from GVP-GNN are effective zero-shot predictors of stability changes of protein complexes even without predicted structures as training data (Table A.7), performing comparably to the best supervised method which uses transfer learning. While we observe a substantial improvement in perplexity when predicted structures are added to training (Table 3.2), this does not further improve complex stability prediction for the single-point mutations in SKEMPI (Table A.7), indicating potential limitations of evaluating models only on single-point mutations.

Binding affinity While the SKEMPI dataset features one mutation entry per protein, we also want to evaluate whether inverse folding models can rank different mutations on the same protein, potentially enabling binding-affinity optimization, which is an important task in therapeutic design. We assess whether inverse folding models can predict mutational effects on binding by leveraging a dataset generated by Starr et al. [9] in which all single amino acid substitutions to the SARS-CoV-2 receptor binding domain (RBD) were experimentally measured for binding affinity to human ACE2. Given potential applications to interface optimization or design, we focus on mutations within the receptor binding motif (RBM), the portion of the RBD in direct contact with ACE2 [10]. When given all RBD and ACE2 coordinates, the best inverse folding model produces RBD-sequence log-likelihoods that have a Spearman correlation of 0.69 with experimental binding affinity measurements (Table 3.3). We observe weaker correlations when not providing the model with ACE2 coordinates, indicating that inverse folding models take advantage of structural information in the binding partner. When masking RBM coordinates (69 of 195 residues, a longer span than masked during model training), we no longer observe correlation between RBD log-likelihood and binding affinity, indicating that the model relies on structural information at the interface to identify interface designs that preserve binding. Zero-shot prediction via inverse folding outperforms methods for sequence-based variant effect prediction, which use the likelihood ratio between the mutant and wildtype amino acids at each position to predict the impact of a mutation on binding affinity. These likelihoods are inferred by masked language models, ESM-1b, ESM-1v, and ESM-MSA-1b, as described by Meier et al. [95] (Table 3.3); additional details are given in Appendix A.2.3.

Sequence insertions Using masked coordinate tokens at insertion regions, inverse folding models can also predict insertion effects. On adeno-associated virus (AAV) capsid variants, we show that relative differences in sequence log-likelihoods correlate with the experimentally measured insertion effects from Bryant et al. [80]. As shown in Table A.8, both GVP-GNN and GVP-Transformer outperform the sequence-only zero-shot prediction baseline ESM-1v [95]. When evaluating on subsets of sequences increasingly further away from the wildtype (≥ 2, ≥ 3, and ≥ 8 mutations), the GVP-GNN-large and GVP-Transformer models trained with predicted structures have increasing advantages compared to GVP-GNN trained without predicted structures.

3.4 Related work

Structure-based protein sequence design Early work on design of protein sequences studied the packing of amino acid side chains to fill the interior space of predetermined backbone structures, either for a fixed backbone conformation [96, 97, 98], or with flexibility in the backbone conformation [26]. Since then, the Rosetta energy function [19] has become an established approach for structure-based sequence design. An alternative non-parametric approach involves decomposing the library of known structures into common sequence-structure motifs [99].

Early machine learning approaches in structure-based protein sequence design used fragment-based and energy-based global features derived from structures [100, 101]. More recently, convolution-based deep learning methods have also been applied to predict amino acid propensities given the surrounding local structural environments [75, 102, 103, 104, 105, 106, 107, 71]. Another recent machine learning approach is to leverage structure prediction networks for sequence design. Anishchenko et al. [34] carried out Monte Carlo sampling in the sequence space to invert the trRosetta [25] structure prediction network for sequence design, while Norn et al. [108] backpropagated gradients through the trRosetta network.

Generative models of proteins The literature on structure-based generative models of protein sequences is the closest to our work. Ingraham et al. [22] introduced the formulation of fixed-backbone design as a conditional sequence generation problem, using invariant features with graph neural networks, modeling each amino acid as a node in the graph with edges connecting spatially adjacent amino acids. Jing et al. [23, 13] further improved graph neural networks for this task by developing architectures with translation- and rotation-equivariance to enable geometric reasoning, showing that GVP-GNN achieves higher native sequence recovery rates than Rosetta on TS50, a benchmark set of 50 protein chains. Strokach et al.

[72] trained graph neural networks for conditional generation with the masked language modeling objective, adding homologous sequences as data augmentation to training. Most recently, contemporary with our work, Dauparas et al. [109] improved an existing graph neural network [22] by combining additional input features and edge updates, and validated designed protein sequences through X-ray crystallography, cryoEM and functional studies. Also contemporary with our work, Yang et al. [110] showed that pretrained sequence-only protein masked language models can be combined with structure-based graph neural networks to improve inverse folding performance.

Recently models have been proposed to jointly generate structures and sequences. Anishchenko et al. [34] generate structures by optimizing sequences through the trRosetta structure prediction network to maximize their difference from a background distribution. The joint generation approach is also being explored in the setting of infilling partial structures. Contemporary to this work, Wang et al. [76] apply span masking to fine-tune the RosettaFold model [41] to perform infilling, although conditioning on both coordinates and amino acid identities instead of considering the inverse folding task. Also contemporary to this work, Jin et al. [111] develop a conditional generation model for jointly generating sequences and structures for antibody complementarity determining regions (CDRs), conditioned on framework region structures. Anand and Achim [112] introduced an equivariant denoising diffusion approach for jointly generating protein structures and sequences for infilling, loop completion, and beyond.

So far there has been little work on generative models of structures directly. Interesting examples include Anand and Huang [20] who model fixed-length protein backbones with generative adversarial networks (GANs) via pairwise distance matrices, and Eguchi et al. [21] who generate antibody structures with variational autoencoders (VAEs).

Language models A large body of work has focused on modeling the sequences in individual protein families. Shin et al. [78] show that protein-specific autoregressive sequence models trained on related proteins can predict point mutation and indel effects and design functional nanobodies. Trinquier et al. [113] also studied protein-specific autoregressive models for sequence generation.

Recently language models have been proposed for modeling large scale databases of protein sequences rather than families of related sequences. Examples include [114, 69, 115, 116, 70, 117, 16, 85]. Meier et al. [95] found that the log-likelihoods of large protein language models predict mutational effects. Madani et al. [77] study an autoregressive sequence model conditioned on functional annotations and show it can generate functional proteins.

Structure-agnostic protein sequence design We point the reader to Wu et al. [118] for a review of the many machine learning-based sequence design approaches that do not explicitly model protein structures. Additionally, as an alternative to sequence generation models, model-guided algorithms design sequences based on predictive models as oracles [119, 120, 121, 122].

Back-translation For machine translation (MT) in NLP, Sennrich et al. [83] studied how to leverage large amounts of monolingual data in the target language, a setting that parallels the situation we consider with protein sequences (the target language in our case). Sennrich et al. found it most effective to generate synthetic source sentences by performing the backwards translation from the target sentence, i.e. back-translation. This parallels the approach we take of predicting structures for sequence targets that have unknown structures. Edunov et al. [84] further investigated back-translation for large-scale language models.

3.5 Conclusions

While there are billions of protein sequences in the largest sequence databases, the number of available experimentally determined structures is on the order of hundreds of thousands, imposing a limit on generative methods that learn from protein structure data. In this work, we explored whether predicted structures from recent deep learning methods can be used in tandem with experimental structures to train models for protein design.

To this end, we generated structures for 12 million UniRef50 sequences using AlphaFold2. As a result of training with this data we observe improvements in perplexity and sequence recovery by substantial margins, and demonstrate generalization to longer protein complexes, to proteins in multiple conformations, and to zero-shot prediction for mutation effects on binding affinity and AAV packaging. These results highlight that in addition to the geometric inductive biases which have been the major focus for work on inverse-folding to date, finding ways to leverage more sources of training data is an equally important path to improved modeling capabilities.

Contemporary with our work, the AlphaFold Protein Structure Database [123] is rapidly growing, featuring 1 million predicted structures as of June 2022. Structure-based protein design models will likely continue to benefit from this new data source as the coverage expands to encompass more of the structural universe. Additionally, with the recent progress in structure prediction for multi-chain protein complexes [82, 124], predicted complex structures could be another valuable source of data for learning protein-protein interactions.

We also take initial steps toward more general structure-conditional protein design tasks. By integrating backbone span masking into the inverse folding task and using a sequence-to-sequence transformer, reasonable sequence predictions can be achieved for short masked spans.

If ways can be found to continue to leverage predicted structures for generative models of proteins, it may be possible to create models that learn to design proteins from an expanded universe of the billions of natural sequences whose structures are currently unknown.

4 | Evolutionary-scale prediction of atomic level protein structure with a language model

4.1 Introduction

The sequences of proteins at the scale of evolution contain an image of biological structure and function. This is because the biological properties of a protein constrain the mutations to its sequence that are selected through evolution, recording biology into evolutionary patterns [125, 126, 127]. Protein structure and function can therefore be inferred from the patterns in sequences [128, 129]. This insight has been central to progress in computational structure prediction starting from classical methods [130, 131], through the introduction of deep learning [65, 132, 42, 25], up to present high accuracy structure prediction [3, 133].

Language models have the potential to learn patterns in protein sequences across evolution. This idea motivates a new line of research on evolutionary scale language models [16], where basic models [134, 69, 135] learn representations that reflect aspects of the underlying biology, and with greater representational capacity capture secondary structure [16, 117] and tertiary structure [136, 16, 137, 138] at a low resolution.

Beginning with Shannon's model for the entropy of text [139], language models of increasing complexity have been developed, culminating in modern large-scale attention based architectures [89, 140, 141]. Despite the simplicity of their training objectives, such as filling in missing words or predicting the next word, language models of text are shown to develop emergent capabilities that develop as a function of scale in increasing computate, data, and number of parameters. Modern language models containing tens to hundreds of billions of parameters develop abilities such as few-shot language translation, commonsense reasoning, and mathematical problem solving, all without explicit supervision [142, 143, 144, 145].

We posit that the task of filling in missing amino acids in protein sequences across evolution will require a language model to understand the underlying structure that creates the patterns in the sequences. As the representational capacity of the language model and the diversity of protein sequences seen in its training increase, we expect deep information about the biological properties of the protein sequences to emerge, since those properties give rise to the patterns that are observed in the sequences. To study this kind of emergence, we scale language models from 8 million parameters up to 15 billion parameters. We discover that atomic resolution structure prediction emerges and continues to improve in language models over the four orders of magnitude in parameter scale. Strong correlations between the language model's understanding of the protein sequence (perplexity) and the accuracy of the structure prediction reveal a close link between language modeling and the learning of structure.

We show that language models enable fast end-to-end atomic resolution structure prediction directly from sequence. Our new approach leverages the evolutionary patterns captured by the language model to produce accurate atomic level predictions. This removes costly aspects of the current state-of-the-art structure prediction pipeline, eliminating the need for a multiple sequence alignment, while at the same time greatly simplifying the neural architecture used for inference. This results in an improvement in speed of up to 60x on the inference forward pass alone, while also removing the search process for related proteins entirely, which can take over 10 minutes with the high-sensitivity pipelines used by AlphaFold [3] and RosettaFold [133], and which is a significant part of the computational cost even with new lower sensitivity fast pipelines [8]. In practice this means the speedup over the state-of-the-art prediction pipelines is up to one to two orders of magnitude.

This makes it possible to expand structure prediction to metagenomic scale datasets. The last decade has seen efforts to expand knowledge of protein sequences to the immense microbial natural diversity of the earth through metagenomic sampling. These efforts have contributed to an exponential growth in the size of protein sequence databases, which now contain billions of proteins [146, 59, 147]. While computational structural characterizations have recently been completed for ∼20K proteins in the human proteome [148], and the ∼200M cataloged proteins of Uniprot [149], the vast scale of metagenomic proteins represents a far greater challenge for structural characterization. The extent and diversity of metagenomic structures is unknown and is a frontier for biological knowledge, and a potential source of new discoveries for medicine and biotechnology [150, 151, 152].

We present the first evolutionary scale structural characterization of metagenomic proteins, folding practically all sequences in MGnify90 [59], over 617M proteins. We are able to complete this characterization in 2 weeks on a heterogeneous cluster of 2,000 GPUs, demonstrating scalability to far larger databases. High confidence predictions are made for over 225M structures, revealing and characterizing regions of metagenomic space distant from existing knowledge with the vast majority (76.8%) of high confidence predictions being separate from UniRef90 [17] by at least 90% sequence identity, and tens of millions of predictions (12.6%) without a match to experimentally determined structures. These results give the first large-scale view into the vast extent and diversity of metagenomic protein structures.

All predictions can be accessed in the ESM Metagenomic Atlas (https://esmatlas.com) open science resource.

4.2 Atomic resolution structure emerges in language models trained on protein sequences

We begin with a study of the emergence of high resolution protein structure. We train a new family of transformer protein language models, ESM-2, at scales from 8 million parameters up to 15 billion parameters. Relative to our previous generation model ESM-1b, ESM-2 introduces improvements in architecture, training parameters, and increases computational resources and data (Appendix A.3.1.1 and Appendix A.3.2). The resulting ESM-2 model family significantly outperforms previously state-of-the-art ESM-1b (a ∼650 million parameter model) at a comparable number of parameters, and on structure prediction benchmarks it also outperforms other recent protein language models (Table A.11).

ESM-2 is trained to predict the identity of amino acids that have been randomly masked out of protein sequences:

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in M} \log p(x_i | x_{\backslash M}) \tag{4.1}$$

Where for a randomly generated mask M that includes 15% of positions i in the sequence, the model is tasked with predicting the identity of the amino acids x_i in the mask from the surrounding context $x_{\backslash M}$ excluding the masked positions. This masked language modeling objective [141] causes the model to learn dependencies between the amino acids. Although the training objective itself is simple and unsupervised, solving it over millions of evolutionarily diverse protein sequences requires the model to internalize sequence patterns across evolution. We expect that this training will cause biological structure to materialize in the language model since it is linked to the sequence patterns. ESM-2 is trained over sequences in the

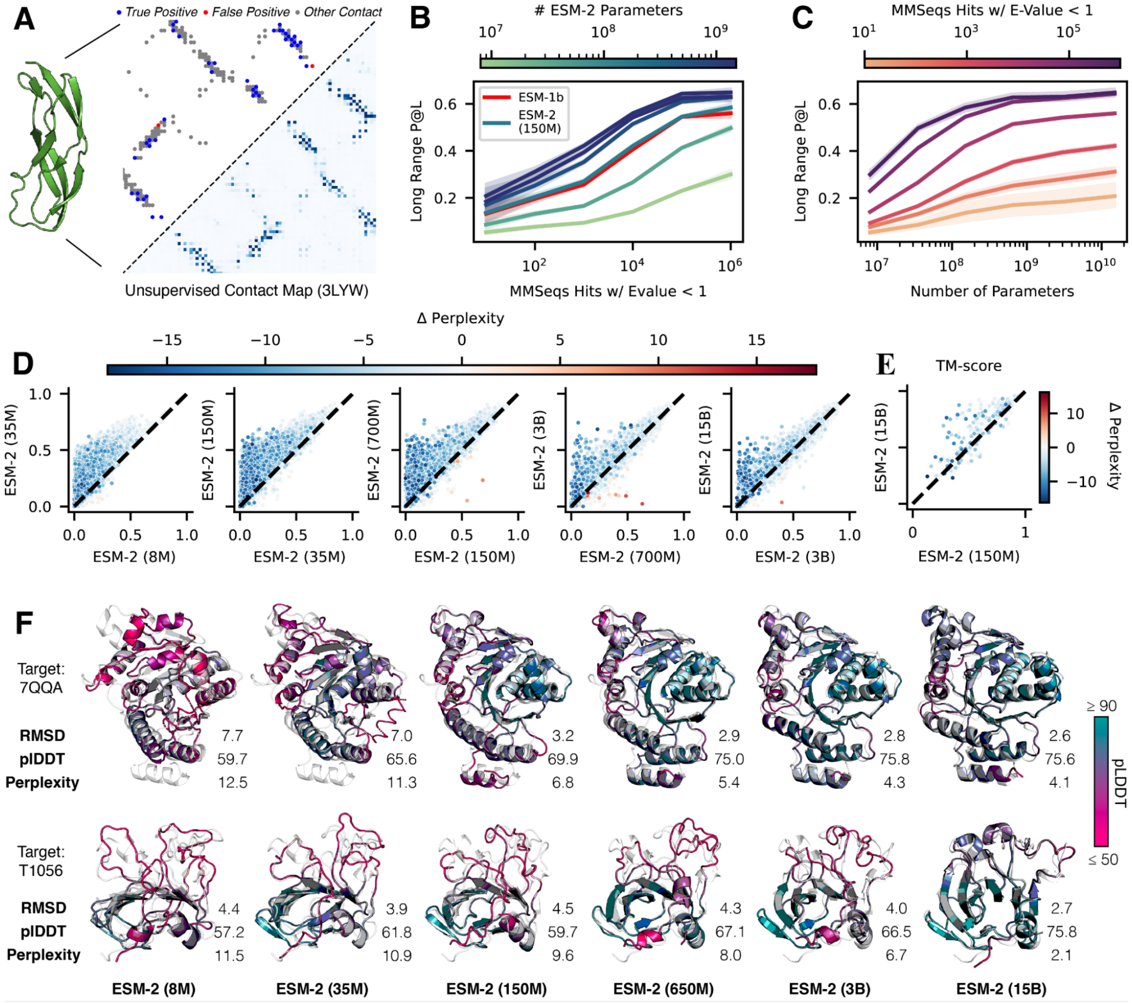

Figure 4.1: *Emergence of structure when scaling language models to 15 billion parameters.* (A) Predicted contact probabilities (bottom right) and actual contact precision (top left) for 3LYW. A contact is a positive prediction if it is within the top-L most likely contacts for a sequence of length L. (B, C, D) Unsupervised contact prediction performance (long range P@L, see Appendix A.3.2.1) for all scales of the ESM-2 model. (B) Performance binned by the number of MMseqs hits when searching the training set. Larger ESM-2 models perform better at all levels; the 150M parameter ESM-2 model is comparable to the 650M parameter ESM-1b model. (C) Trajectory of improvement as model scale increases for sequences with different numbers of MMseqs hits. (D) Left-to-right shows models from 8M to 15B parameters, comparing the smaller model (x-axis) against the next larger model (y-axis) via unsupervised contact precision. Points are PDB proteins colored by change in perplexity for the sequence between the smaller and larger model. Sequences with large changes in contact prediction performance also exhibit large changes in language model understanding measured by perplexity. (E) TM-score on combined CASP14 and CAMEO test sets. Predictions are made using structure module-only head on top of language models. Points are colored by the change in perplexity between the models. (F) Structure predictions on CAMEO structure 7QQA and CASP target 1056 at all ESM-2 model scales, colored by pLDDT (pink = low, teal = high). For 7QQA, prediction accuracy improves at the 150M parameter threshold. For T1056, prediction accuracy improves at the 15B parameter threshold.

UniRef [17] protein sequence database. During training, sequences are sampled with even weighting across ∼43 million UniRef50 training clusters from ∼138 million UniRef90 sequences so that over the course of training the model sees ∼65 million unique sequences.

As we increase the scale of ESM-2 from 8 million to 15 billion parameters, we observe large improvements in the fidelity of its modeling of protein sequences. This fidelity can be measured using perplexity, which ranges from 1 for a perfect model to 20 for a model that makes predictions at random. Intuitively, the perplexity describes the average number of amino acids the model is choosing between for each position in the prediction. Mathematically, perplexity is defined as the exponential of the negative log-likelihood of the sequence (Appendix A.3.2.2). Figure A.12 shows perplexity for the ESM-2 family as a function of the number of training updates, evaluated on a set of ∼500K UniRef50 clusters that have been held out from training. Comparisons are performed at 270k training steps for all models in this section. The fidelity continues to improve as the parameters increase up to the largest model. The 8M parameter model has a perplexity of 10.45, and the 15B model reaches a perplexity of 6.37, indicating a large improvement in the understanding of protein sequences with scale.

This training also results in the emergence of structure in the models. Since ESM-2's training is only on sequences, any information about structure that develops must be the result of representing the patterns in sequences. Transformer models trained with masked language modeling, are known to develop attention patterns that correspond to the residue-residue contact map of the protein [136, 137]. We examine how this low resolution picture of protein structure emerges as a function of scale. We use a linear projection to extract the contact map from the attention patterns of the language model (Appendix A.3.2.1). The precision of the top L (length of the protein) predicted contacts (long range contact precision) measures the correspondence of the attention pattern with the structure of the protein. Attention patterns develop in ESM-2 that correspond to tertiary structure (Figure 4.1a), and scaling leads to large improvements in the understanding of structure (Figure 4.1b). The accuracy of the predicted contacts varies as a function of the number of evolutionarily related sequences in the training set. Proteins with more related sequences in the training set have steeper learning trajectories with respect to model scale (Figure 4.1c). This means that improvement on sequences with high evolutionary depth saturates at lower model scales, and improvement on sequences with low evolutionary depth continues as models increase in size.

For individual proteins, we often observe non-linear improvements in the accuracy of the contact prediction as a function of scale. Figure 4.1d plots the change in the distribution of long range contact precision at each transition to a higher level of scale. At each step there is an overall shift in the distribution toward better performance. Also at each transition, there is a subset of proteins that undergo significant improvement. In Figure 4.1d these are in the upper left of each plot, far from the diagonal. The accuracy of the contact map prediction and perplexity are linked, with proteins undergoing large changes in contact map accuracy also undergoing large changes in perplexity (NDCG = 0.87, Appendix A.3.2.6). This link indicates that the language modeling objective is directly correlated with the materialization of the folded structure in the attention maps.

We investigate whether high resolution structure at an atomic level also develops. To identify atomic resolution information in the model, we project out spatial coordinates for each of the atoms from the internal representations of the language model using an equivariant transformer (Appendix A.3.3.3). This projection is fit using experimentally determined protein structures from PDB [153], and evaluated on 194 CAMEO proteins [154] and 51 CASP14 proteins [155]. TM-score, which ranges from 0 to 1, measures the accuracy of the projection in comparison to the ground truth structure, with a value of 0.5 corresponding to the threshold for correctly predicting the fold [156]. The evaluation uses a temporal cutoff, ensuring that the proteins used for testing are held out from those used in fitting the projection. This makes it possible to measure how atomic level information emerges in the representations as a function of the parameter scale.

We discover that an atomic resolution structure prediction can be projected from the representations of the ESM-2 language models. The accuracy of this projection improves with the scale of the language model. The 15 billion parameter model reaches a TM-score of 0.72 on the CAMEO test set and 0.55 on the CASP14 test set, a gain of 14% and 17% respectively relative to the the 150 million parameter ESM-2 model (Figure 4.1e). At each increase in scale a subset of proteins undergo large changes in accuracy. For example, the protein 7QQA improves in RMSD from 7.0 to 3.2 when scale is increased from 35M to 150M parameters, and the CASP target T1056 improves in RMSD from 4.0 to 2.6 when scale is increased from 3B to 15B parameters (Figure 4.1f). Before and after these jumps, changes in RMSD are much smaller. Across all models (Table A.11) there is a correlation of -0.99 between validation perplexity and CASP14 TM-score, and -1.00 between validation perplexity and CAMEO TM-score indicating a strong connection between the understanding of the sequence measured by perplexity and the atomic resolution structure prediction. Additionally there are strong correlations between the low resolution picture of the structure that can be extracted from the attention maps and the atomic resolution prediction (0.96 between long range contact precision and CASP14 TM-score, and 0.99 between long range contact precision and CAMEO TM-score). These findings connect improvements in language modeling with the increases in low resolution (contact map) and high resolution (atomic level) structural information.

4.3 Accelerating accurate atomic resolution structure prediction with a language model

Language models greatly accelerate state-of-the-art high resolution structure prediction. The language model internalizes evolutionary patterns linked to structure, eliminating the need for external evolutionary databases, multiple sequence alignments, and templates. We find that

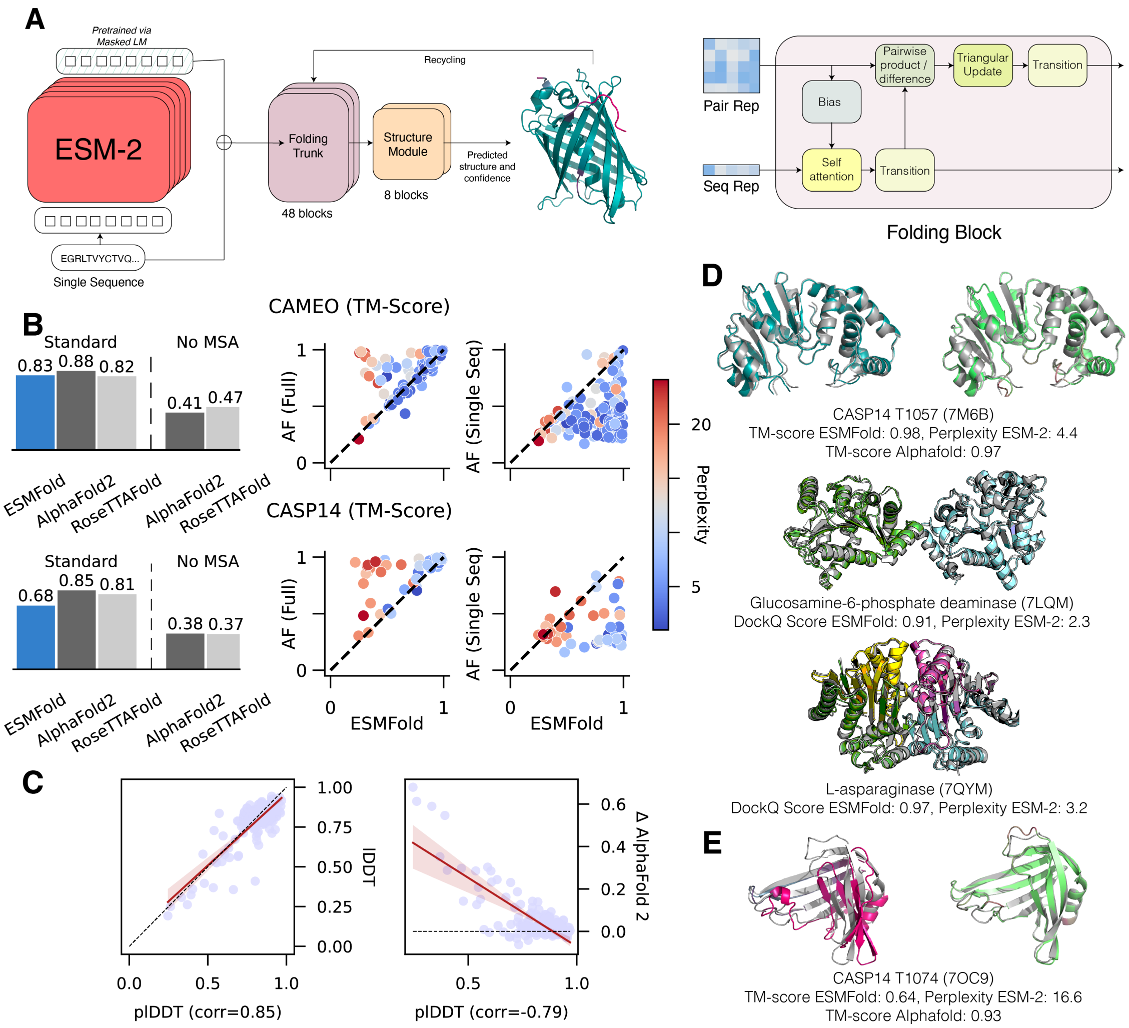

Figure 4.2: *Single sequence structure prediction with ESMFold.* (A) ESMFold model architecture. Arrows show the information flow in the network from the language model to the folding trunk to the structure module which outputs 3D coordinates and confidences. (B) ESMFold produces accurate atomic resolution predictions, with similar accuracy to RosettaFold on CAMEO. When MSAs are ablated for AlphaFold and RosettaFold, performance of the models degrades. Scatter-plots compare ESMFold (x-axis) predictions with AlphaFold2 (y-axis), colored by language model perplexity. Proteins with low perplexity score similarly to AlphaFold2. (C) Model pLDDT vs. true LDDT (left) and relative performance against AlphaFold (right) on CAMEO. pLDDT is a well calibrated estimate of prediction accuracy. (D) Successful examples: Top shows test-set predictions of T1057, with ESMFold (left) and AlphaFold2 (right). Coloring shows predicted LDDT for both models (ESMFold high-confidence=teal, AlphaFold2 high-confidence=green, both low-confidence=pink). Ground truth is shown in grey. Bottom two show complex predictions on a dimer (7LQM) and a tetramer (7QYM); ESMFold predictions are colored by chain ID and overlaid on ground truth (gray). DockQ [4] scores are reported for the interactions; in the case of the tetramer 7QYM, the score is the average of scores over interacting chain-pairs. (E) Unsuccessful example: Test-set predictions of T1074, with ESMFold (left) and AlphaFold2 (right). Coloring shows predicted LDDT for both models (ESMFold high-confidence=teal, AlphaFold2 high-confidence=green, both low-confidence=pink). Ground truth is shown in grey. ESMFold TM-score is significantly below AlphaFold2 TM-score. The perplexity of the unsuccessful sequence is 16.6, meaning the language model does not understand the input sequence.

the ESM-2 language model generates state-of-the-art three-dimensional structure predictions directly from the primary protein sequence. This results in a speed improvement for structure prediction of more than an order of magnitude while maintaining high resolution accuracy.

We develop ESMFold, a fully end-to-end single sequence structure predictor, by training a folding head for ESM-2 (Figure 4.2a). At prediction time the sequence of a protein is input to ESM-2. The sequence is processed through the feedforward layers of the language model, and the model's internal states (representations) are passed to the folding head. The head begins with a series of folding blocks. Each folding block alternates between updating a sequence representation and a pairwise representation. The output of these blocks is passed to an equivariant transformer structure module, and three steps of recycling are performed before outputting a final atomic-level structure and predicted confidences (Appendix A.3.3.1). This architecture represents a major simplification in comparison to current state-of-the-art structure prediction models which deeply integrate the multiple sequence alignment into the neural network architecture through an attention mechanism operating across the rows and columns of the MSA [3, 85].

Our approach results in a significant improvement in prediction speed. On a single NVIDIA V100 GPU, ESMFold makes a prediction on a protein with 384 residues in 14.2 seconds, 6x faster than a single AlphaFold2 model. On shorter sequences the improvement increases up to ∼60x (Figure A.13). The search process for related sequences, required to construct the MSA, can take over 10 minutes with the high sensitivity protocols used by the published versions of AlphaFold and RosettaFold; this can be reduced to less than 1 minute, although with reduced sensitivity [8].

We train the folding head on ∼25K clusters covering a total of ∼325K experimentally determined structures from the PDB, further augmented with a dataset of ∼12M structures we predicted with AlphaFold2 (Appendix A.3.1.2). The model is trained with the same losses that are used for AlphaFold [157]. To evaluate the accuracy of structure predictions we use test sets that are held out from the training data by a May 2020 cutoff date; as a result all structures that are used in evaluation are held out from the training, and the evaluation is representative of the performance that would be expected in regular usage as a predictive model on the kinds of structures that are selected by experimentalists for characterization. This also makes it possible to compare with AlphaFold and RosettaFold since these models also have not been trained on structures deposited after May 2020. We use two test sets: the CAMEO test set consists of 194 structures used in the ongoing CAMEO assessment (between April 2022 to June 2022); the CASP14 test set consists of 51 publicly released structures that have been selected for their difficulty for the biannual structure prediction competition.

We compare results on these evaluation sets to AlphaFold2 and RosettaFold (Figure 4.2b). ESMFold achieves an average TM-score of 0.83 on CAMEO and 0.68 on CASP14. Using the search protocols released with AlphaFold2, including MSAs and templates, AlphaFold2 achieves 0.88 and 0.85 on CAMEO and CASP14 respectively. ESMFold achieves competitive accuracy with RosettaFold on CAMEO, which averages a TM-score of 0.82. When evaluating AlphaFold2 and RosettaFold on single sequences by ablating the multiple sequence alignment, performance degrades substantially, and falls well below that of ESMFold. Note that this is an artificial setting as AlphaFold2 has not been explicitly trained for single sequences, however it has recently emerged as important in protein design, where these models have been used with single sequence inputs for *de novo* protein design [158, 159, 160].

While average performance on the test sets is below AlphaFold2, the performance gaps are explained by the language model perplexity. On proteins where perplexity is low, ESMFold results match AlphaFold2. On the CAMEO test set, the 3B parameter ESM-2 model used in ESMFold achieves an average perplexity of 5.7. On the CASP14 test set, the same model only has an average perplexity of 10.0. Performance within each set is also well correlated with perplexity. On the CAMEO test set, language model perplexity has a Pearson correlation of -0.52 with the TM-score between the predicted and experimental structures; on CASP14, the correlation is -0.71 (Figure 4.2b). On the subset of 18 CASP14 proteins where ESM-2 achieves perplexity < 7, ESMFold matches AlphaFold in performance (average TM-score difference < 0.03, and no TM-score differences > 0.1). The relationship between perplexity and structure prediction suggests that improvements in the language model will translate into improvements in single-sequence structure prediction accuracy, consistent with observations from the scaling analysis (Figures 4.1d and 4.1e). Additionally, this means the language model's perplexity for a sequence can be used to predict the quality of the ESMFold structure prediction.

Ablation studies indicate that the language model representations are critical to ESMFold performance (Figure A.14). With a folding trunk of 8 blocks, performance on the CAMEO test set is 0.74 LDDT (baseline). Without the language model, this degrades substantially, to 0.58 LDDT. When removing the folding trunk entirely (i.e. only using the language model and the structure module), performance degrades to 0.66 LDDT. Other ablations: only 1 block of a structure module, turning off recycling, not using AlphaFold2 predicted structures as distillation targets, or not using triangular updates, result in small performance degradations (change in LDDT of -0.01 to -0.04).

ESMFold provides state-of-the-art structure prediction accuracy, matching AlphaFold2 performance (< 0.05 LDDT difference) on more than half the proteins (Figure 4.2b). We find that this is true even on some large proteins—T1076 is an example with 0.98 TM-score and 540 residues (Figure 4.2d). Parts of structure with low accuracy do not differ significantly between ESMFold and AlphaFold, suggesting that language models are learning information similar to that contained in MSAs. We also observe that ESMFold is able to make good predictions for components of homo- and heterodimeric protein-protein complexes (Figure 4.2d). In a comparison with AlphaFold-Multimer [82] on a dataset of 2,978 recent multimeric complexes deposited in the PDB, ESMFold achieves the same qualitative DockQ [4] categorization for 53.2% of chain pairs, despite not being trained on protein complexes (Figure A.15).

Confidence is well calibrated with accuracy. ESMFold reports confidence in the form of predicted-LDDT. This confidence correlates well with the accuracy of the prediction, and for high-confidence predictions (pLDDT > 0.7) accuracy is comparable to AlphaFold2 (ESMFold LDDT=0.83, AlphaFold2 LDDT=0.85 on CAMEO) (Figures 4.2c and A.16). High-confidence predictions approach experimental-level accuracy. On the CAMEO test set, ESMFold predictions have a median all-atom RMSD_{95} (root-mean-squared deviation at 95% residue coverage) of 1.91Å and backbone RMSD_{95} of 1.33Å. When confidence is very high (pLDDT > 0.9), predictions have median all-atom RMSD_{95} of 1.42Å and backbone RMSD_{95} of 0.94Å. This means the confidence can be used to predict how likely it is that a given structure prediction will match the true structure if it were to be experimentally determined.

4.4 Evolutionary-scale structural characterization of metagenomics

This fast and high resolution structure prediction capability enables the first large-scale structural characterization of metagenomic proteins. We fold over 617 million sequences from the MGnify90 database [59]. This is the entirety of the sequences of length 20 to 1024, and covers 99% of all the sequences in MGnify90. Overall, the characterization produces ∼365 million predictions with good confidence (mean pLDDT > 0.5 and pTM > 0.5) corresponding to ∼59% of the database, and ∼225 million predictions with high confidence (mean pLDDT > 0.7 and pTM > 0.7) corresponding to ∼36% of total structures folded (Figure 4.3). We were able to complete the predictions in 2 weeks on a cluster of approximately 2,000 GPUs (Appendix A.3.4.1).

For structure prediction at scale, it is critical to distinguish well predicted proteins from those that are poorly predicted. In the previous section, we evaluated calibration against experimentally determined structures on held out test sets, finding that the model confidence is predictive of the agreement with experimentally determined structures. We also assess calibration against AlphaFold predictions on metagenomic proteins. On a random subset of ∼4K metagenomic sequences, there is a high correlation (Pearson r = 0.79) between ESMFold pLDDT and the LDDT to AlphaFold2 predictions (Figure 4.3a). Combined with results on CAMEO showing that when confidence is very high (pLDDT $>$ 0.9), ESMFold predictions often approach experimental accuracy, these findings mean that ESMFold's confidence scores provide a good indication of the agreement with experimental structures and with the predictions that can be obtained from AlphaFold2. Across the ∼617 million predicted structures, ∼113 million structures meet the very high confidence threshold.

Many of the metagenomic structure predictions have high confidence (Figure 4.3b) and are novel with respect to existing sequence and structure databases (Figures 4.3c to 4.3e). On a random sample of 1 million high confidence structures, 76.8% (767,580) of the proteins have a sequence identity below 90% to any sequence in UniRef90, indicating that these proteins are distinct from existing UniRef90 sequences (Figure 4.3d). For 3.4% (33,521 proteins), no significant match is found in UniRef90 at all (Appendix A.3.4.2). We use Foldseek [5] to compare the predicted structures to known experimentally determined structures in the PDB database. At a threshold of 0.7 TM-score, Foldseek reports 25.4% (253,905 proteins) without a match. At a threshold of 0.5 TM-score, Foldseek reports 12.6% (125,765 proteins) without a match in the PDB database (Figures 4.3c and 4.3e). For 2.6% (25,664) there is both low structural similarity (TM-score $\leq$ 0.5) and no close sequence homolog ($>$ 30% identity) (Figure 4.4a and Table A.12). Based on these subsampled estimates, there are approximately 28 million proteins (12.6% of 225 million) with both high confidence predictions and TMScore $<$ 0.5 to known protein structures. These results indicate that there are millions of structures in the atlas that are novel in comparison to known structures, and that ESMFold can effectively characterize regions of protein space that are distant from existing knowledge.

Large scale structural characterization also makes it possible to identify structural similarities in the absence of sequence similarity. Many high-confidence structures with low similarity to UniRef90 sequences do have similar structures in the PDB. This remote homology often extends beyond the limit detectable by sequence similarity. For example, MGnify sequence MGYP000936678158 has no significant sequence matches to any entry in UniRef90, nor any significant matches via a jackhmmer [161] reference proteome search, but has a predicted structure conserved across many nucleases (PDB 5YET_B, TM-score 0.68; PDB 3HR4_A, TM-score 0.67) (Figure 4.4b and Table A.12); similarly, MGnify sequence MGYP004000959047 has no significant UniRef90 or jackhmmer reference proteome matches but its predicted structure has high similarity to experimental structures of lipid binding domains (PDB 6BYM_A, TM-score 0.80; PDB 5YQP_B, TM-score 0.78) (Figure 4.4c and Table A.12). The ability to detect remote similarities in structure enables insight into function that cannot be obtained from the sequence.

All predicted structures are available in the ESM Metagenomic Atlas (https://esmatlas.com) as an open science resource. Structures are available for bulk download, via a programmatic API, and through a web resource which provides search by sequence and by structure [162, 5]. These tools facilitate both large scale and focused analysis of the full scope of the hundreds of millions of predicted structures.

4.5 Conclusions

Fast and accurate computational structure prediction has the potential to accelerate progress toward an era where it is possible to understand the structure of all proteins discovered in gene sequencing experiments. This promises new insights into the vast natural diversity of proteins, most of which is being newly discovered in metagenomic sequencing. To this end we have completed the first large-scale structural characterization of metagenomic proteins.

This characterization reveals the structures of hundreds of millions proteins that have been previously unknown, millions of which are potentially novel in comparison to experimentally determined structures.

As structure prediction continues to scale to larger numbers of proteins, calibration becomes critical, since when the throughput of prediction is limiting, the accuracy and speed of the prediction form a joint frontier in the number of accurate predictions that can be generated. Very high confidence predictions in the metagenomic atlas are expected to often be reliable at a resolution sufficient for insight similar to experimentally determined structures, such as into the biochemistry of active sites [163]. For many more proteins where the topology is predicted reliably, insight can be obtained into function via remote structural relationships that could not be otherwise detected with sequence.

The emergence of atomic level structure in language models shows a high resolution picture of protein structure encoded by evolution into protein sequences that can be captured with unsupervised learning. ESM-2 is the result of our work over several years focusing on emergence of biological properties, and is the first time a language model has been shown to learn structure at an atomic resolution. Our current models are very far from the limit of scale in parameters, sequence data, and compute that can in principle be applied. We are optimistic that as we continue to scale there will be further emergence. Our results showing the improvement in the modeling of low depth proteins point in this direction.

ESM-2 results in an advance in speed that in practical terms is up to one to two orders of magnitude, which puts far larger numbers of sequences within reach of accurate atomic level prediction. Structure prediction at the scale of evolution can open a deep view into the natural diversity of proteins, and accelerate the discovery of new protein structures and functions.

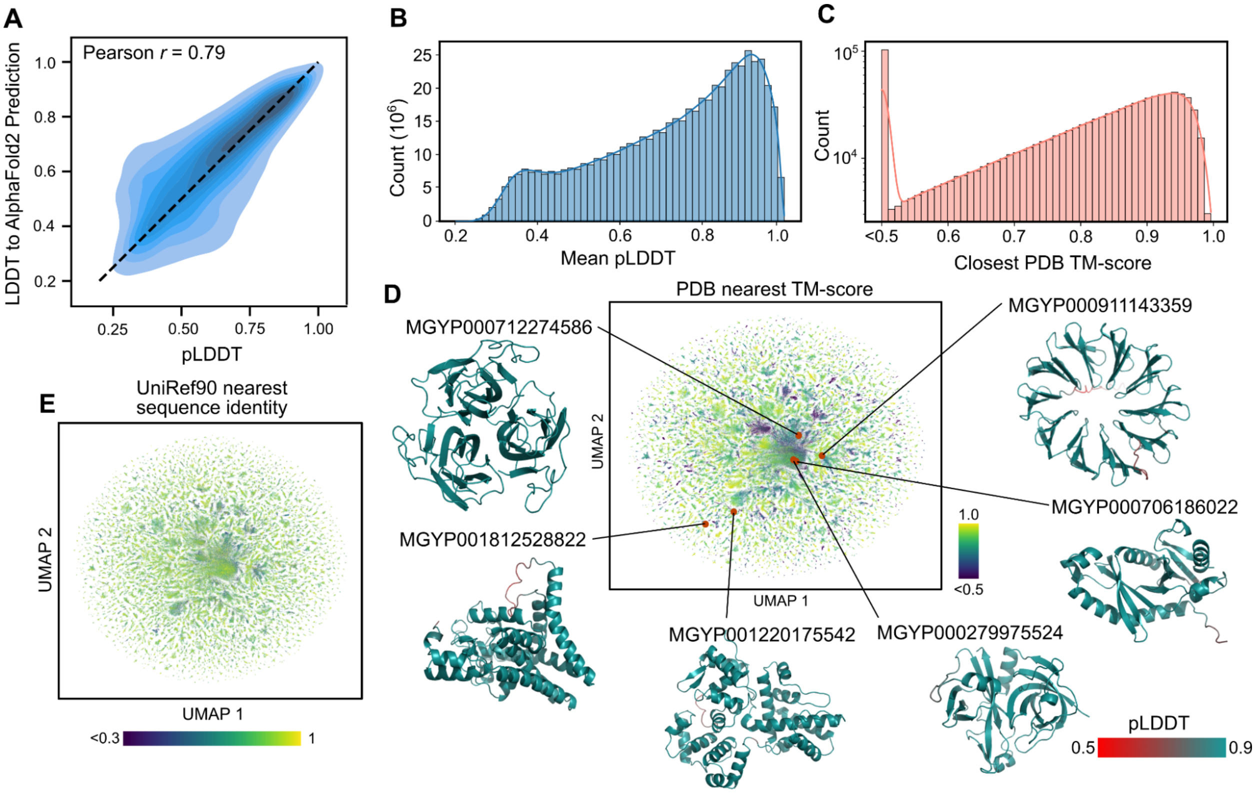

Figure 4.3: *Mapping metagenomic structural space.* (A) ESMFold calibration with AlphaFold2 for metagenomic sequences. Mean pLDDT is shown on the x-axis, and LDDT to the corresponding AlphaFold2 prediction is shown on the y-axis. Distribution is shown as a density estimate across a subsample of ∼4K sequences from the MGnify database. (B) The distribution of mean pLDDT values computed for each of ∼617 million ESMFold-predicted structures from the MGnify database. (C) The distribution of the TM-score to the most similar PDB structure for each of 1 million randomly sampled high confidence (mean pLDDT > 0.7 and pTM > 0.7) structures. Values were obtained by a Foldseek search, which does not report values under 0.5 TM-score [5]. (D) This sample of 1 million high-confidence protein structures is visualized in two dimensions using the UMAP algorithm and colored according to distance from nearest PDB structure, where regions with low similarity to known structures are colored in dark blue. Example protein structures and their locations within the sequence landscape are provided; see also Figure 4.4 and Table A.12. (E) Additional UMAP plot in which the 1 million sequences are plotted according to the same coordinates as in (D) but colored by the sequence identity to the most similar entry in UniRef90 according to a blastp search.

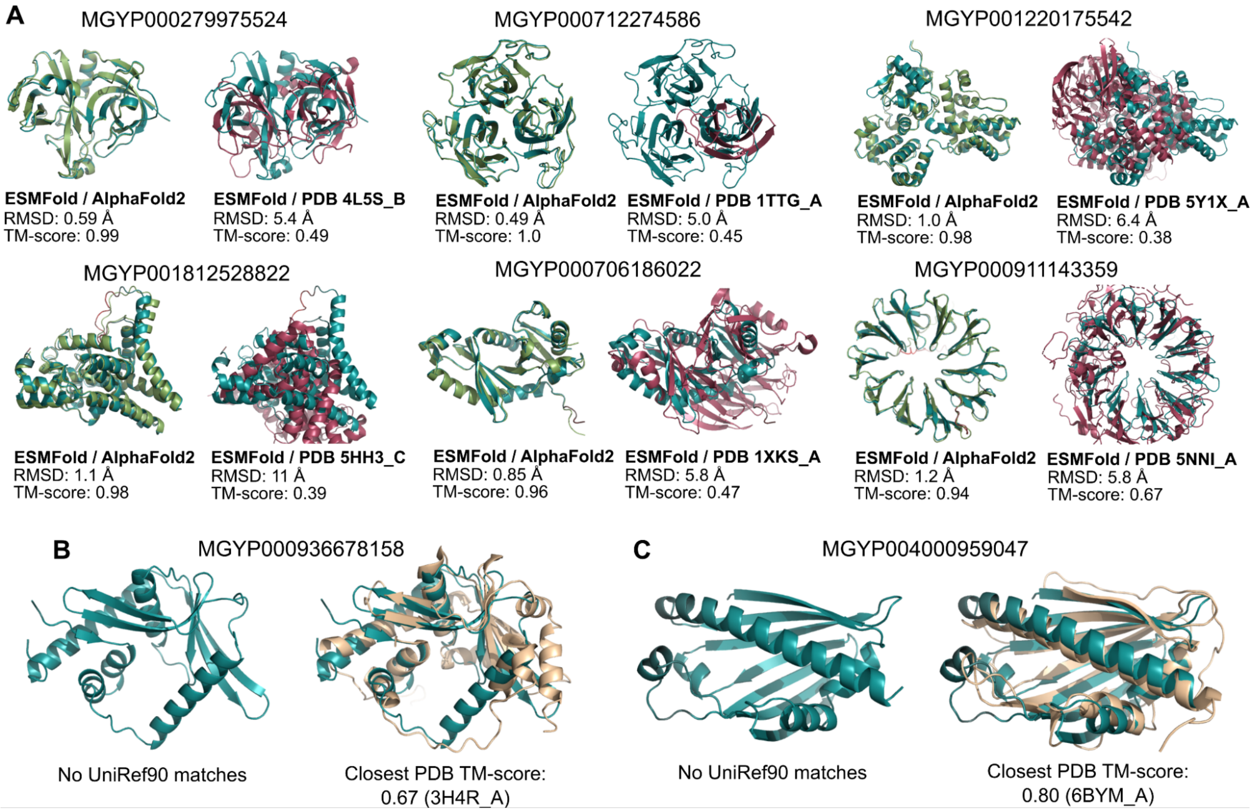

Figure 4.4: *Example ESMFold structure predictions of metagenomic sequences.* (A) Example predicted structures from six different metagenomic sequences; also see Table A.12. Left of each subfigure: The prediction is displayed with the AlphaFold2 prediction (light green). Right of each subfigure: The prediction is displayed with the Foldseek-determined nearest PDB structure according to TM-score. (B, C) Examples of two ESMFold-predicted structures that have good agreement with experimental structures in the PDB but that have low sequence identity to any sequence in UniRef90. (B) The predicted structure of MGYP000936678158 aligns to an experimental structure from a bacterial nuclease (light brown, PDB: 3H4R), while (C) the predicted structure of MGYP004000959047 aligns to an experimental structure from a bacterial sterol binding domain (light brown, PDB: 6BYM).

5 | A HIGH-LEVEL PROGRAMMING LANGUAGE FOR GENERATIVE PROTEIN DESIGN

5.1 Introduction

Protein design would benefit from the regularity, simplicity, and programmability provided by a basic set of abstractions [164, 165, 166, 167] like those used in the engineering of buildings, machines, circuits, and computer software. But unlike these artificial creations, proteins cannot be decomposed into easily recombinable parts because the local structure of the sequence is entangled in its global context [168, 169]. Classical de novo protein design has attempted to determine a fundamental set of structural building blocks, which could then be assembled into higher-order structures [26, 27, 28, 29, 170]. Likewise, traditional protein engineering often recombines segments or domains of natural protein sequences into hybrid chimeras [30, 31, 32]. However, existing approaches have not been able to achieve the high combinatorial complexity that is necessary for true programmability.

We show modern generative models realize these classical goals of modularity and programmability at a new level of combinatorial complexity. Our idea is to place the modularity and programmability at a higher level of abstraction, where a generative model bridges the

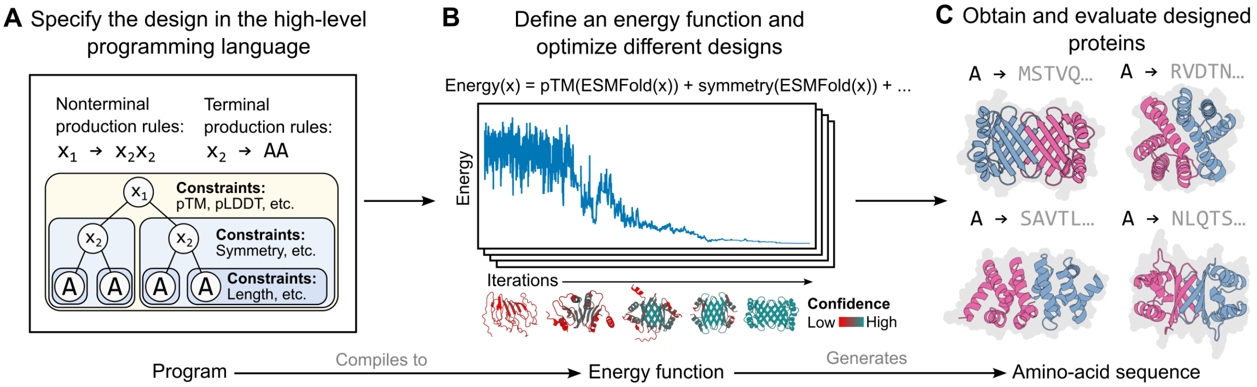
A Specify the design in the high-level programming language
Nonterminal production rules:
x1 → x2x2
Terminal production rules:
x2 → AA
x1
Constraints: pTM, pLDDT, etc.
x2
x2
Constraints: Symmetry, etc.
A
A
A
A
Constraints: Length, etc.
B Define an energy function and optimize different designs
Energy(x) = pTM(ESMFold(x)) + symmetry(ESMFold(x)) + ...
Energy
Iterations
Confidence
Low
High
C Obtain and evaluate designed proteins
A → MSTVQ...
A → RVDTN...
A → SAVTL...
A → NLQTS...
Program
Compiles to
Energy function
Generates
Amino-acid sequence

Figure 5.1: *Overview of the high-level programming language and the optimization algorithm.* (**A**) We propose a high-level programming language in which each program consists of (1) a syntax tree (corresponding to a set of nonterminal and terminal production rules) that enables modular and hierarchical organization of protein subunits and (2) a set of constraint functions that can be defined at each node of the syntax tree, where a given constraint is applied to the entire subtree rooted at the corresponding node. (**B**) This program is then compiled to a single energy function, which in our study is a simple linear combination of the specified constraint functions. The energy function is used to guide an optimization procedure based on simulated annealing, of which a key component is the use of an accurate and efficient structure predictor to evaluate the energy function at each step of the optimization. The same energy function can guide multiple optimization trajectories. (**C**) Each of these trajectories produces a protein sequence design and an associated predicted structure. These sequences and predicted structures can then be evaluated downstream using in silico and experimental metrics.

gap between human intuition and the production of specific sequences and structures. In this setting, the protein designer needs only to recombine high-level directives, while the task of obtaining a protein that fulfills those directives is placed on the generative model.

We propose a programming language for generative protein design, which allows a designer to specify intuitive, modular, and hierarchical programs. We show that high-level programs can be translated into low-level sequences and structures by a generative model. Our approach leverages advances in protein language models, which learn structural information [16, 171] and the design principles of proteins (see accompanying paper by Verkuil et al.).

In this study, our specific implementation is based on an energy-based generative model. First, a protein designer specifies a high-level program consisting of a set of hierarchically organized constraints (Figure 5.1a). Then, this program compiles to an energy function that evaluates compatibility with the constraints, which can be arbitrary and non-differentiable (Figure 5.1b). We apply constraints on structure by incorporating atomic-level structure predictions, enabled by a language model, into the energy function. This approach enables the generation of a wide set of complex designs (Figure 5.1c).

The use of a high-level language allows the protein designer to systematically reason about the design space and specify very general, modular, and composable programs. To demonstrate this, we generate proteins that realize a variety of constraints that include secondary structure, symmetry, multimerization, and atomic-level coordination in the predicted structures.We apply these constraints in complex, hierarchical settings, where we can enumerate a space of highly idealized forms that have low similarity to natural structures. As de novo design progresses to more complex proteins and protein assemblies, high-level abstractions such as the programming language described in this study should facilitate the systematic exploration and design of complex artificial proteins.

5.2 A generative programming language for protein design

We introduce a high-level programming language for generative protein design. This language first requires a syntax tree (Figure 5.1a) consisting of terminal symbols (i.e., the leaves of the tree) that each corresponds to a unique protein sequence (which is potentially repeated within the protein) and nonterminal symbols (i.e., the internal nodes of the tree) that enable hierarchical organization. Second, the language requires a set of constraints: at each node in the tree, a protein designer can specify any number of constraints, which are applied to the entire subtree. The syntax tree and its constraints fully specify a program in our high-level language. We provide a more extended description of this language in the Methods section.

Each program is compiled into an energy function that specifies a generative model for that program in the form of a distribution

$$p_\theta(\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}))}{Z(\theta)}$$

over protein sequences $\mathbf{x}$ conforming to the program. The constraints are encoded as weighted terms that are additively combined into the total energy. Since the partition function Z, which is a function of the parameters θ, is intractable, low temperature samples can be taken with MCMC and simulated annealing (Figure 5.1b). The generative capacity of this approach is built on recent developments in deep learning for protein biology. Specifically, each step in the optimization loop has access to a fast and accurate atomic-level structure prediction enabled by the ESM-2 protein language model.

Given a single program, the generative model can create potentially diverse designs that fulfill the user-specified constraints (Figure 5.1c). These constraints can be arbitary and nondifferentiable, and can span multiple scales of biological complexity, from atomic-level coordinates to abstract plans of the protein including the overall topology and symmetry. This approach allows the model to propose diverse solutions where many potential designs may satisfy the program. By leveraging an expressive model of structure in a generative capacity, the resulting designs respect the various constraints individually and are also globally coherent.

5.3 Results

5.3.1 Full-protein constraints

We first demonstrate that our approach can design proteins where the constraints are simply applied to the entire sequence and structure without any hierarchical organization. An especially valuable constraint, which we apply generally across all our design efforts, steers the optimization toward predicted structures with higher model confidence, i.e., high pTM and mean pLDDT. For proteins that we desire to have soluble, monomeric expression, we also steer the optimization to minimize hydrophobic residues that are solvent-exposed (Methods). Using only these constraints on structural confidence and hydrophobic residue placement, our model is able to generate or "freely hallucinate" [34] high-confidence structures (Figures 5.2a and 5.2b); across 200 seeds, all optimization loops produced predicted structures with an ESMFold mean pLDDT greater than 0.7 (Figure 5.2B). Of these, a large portion (44, or 22%) also had high predicted confidence (pLDDT > 0.7) by single-sequence AlphaFold2 [3] (Figure 5.2c), a separate structure prediction model that was not used in our optimization procedure.

Our objective function also enables other full-protein constraints, such as specifying the positions of the backbone atoms while allowing the algorithm to design the corresponding sequence, a design task referred to as fixed backbone design [74]. To achieve this, we can add a term to the energy function that minimizes the root-mean-square deviation (RMSD)

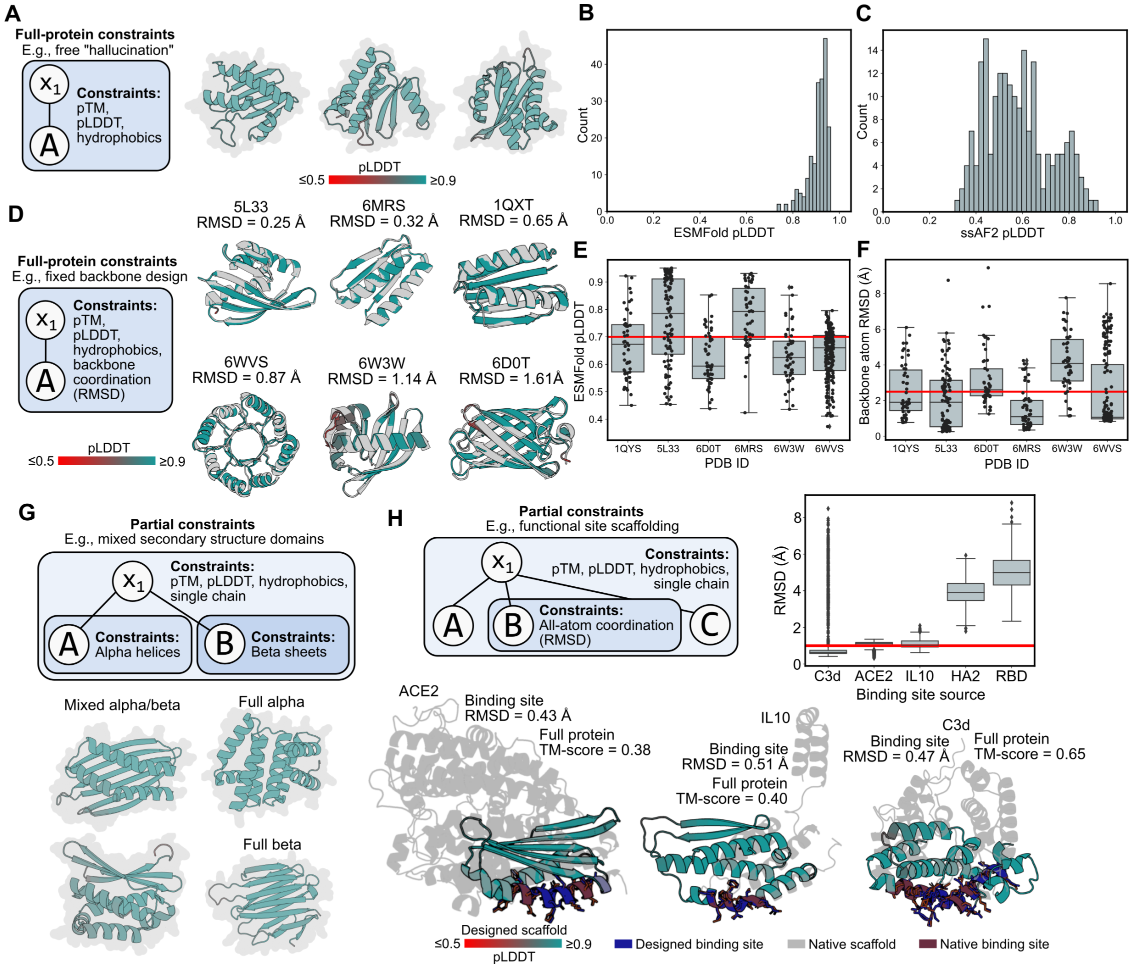

Figure 5.2: *Programming full-protein or partial constraints.* (**A**) A graphical representation of a program for protein "free hallucination" (left) along with three example designed structures (right). (**B**) The distribution of ESMFold pLDDT values over 200 free-hallucinated structures. Of these, 100% have good confidence (ESMFold pLDDT $>$ 0.7). (**C**) The distribution of single-sequence AlphaFold2 (ssAF2) over the same 200 structures; note that ssAF2 was not used in the design procedure. Of these, 22% have good confidence (ssAF2 pLDDT $>$ 0.7). (**D**) A graphical representation of a program for fixed backbone design (left) along with example designs for six de novo target backbones. The experimental backbone is colored gray; the designed backbone is colored by ESMFold pLDDT. (**E**) For each target backbone, the distribution of the ESMFold pLDDT values of the final designs from 50 or more fixed backbone design seeds is plotted as a boxplot (for all boxplots in this figure, the box extends from first to third quartile, black line indicates the median, and whiskers indicate 2.5 times the interquartile range) with each seed also plotted as a black circle. A horizontal red line indicates pLDDT = 0.7. (**F**) For each target backbone, the distribution of RMSD values between the target and design backbone atoms from 50 or more fixed backbone design seeds is plotted as a boxplot with each seed also plotted as a black circle. A horizontal red line indicates RMSD = 2.5 Å. (**G**) A graphical representation of a program for designing a protein with mixed secondary structure (top) along with example designs in which secondary structure was explicitly specified (bottom). (**H**) Top-left: A graphical representation of a program for functional site scaffolding. Top-right: For each scaffolded binding site, the distribution of RMSD between the native and designed binding site atoms (including side chains) from 2,000 seeds is plotted as a boxplot. A horizontal red line indicates RMSD = 2 Å. Bottom: Example designs that achieve sub-angstrom atomic coordination in the scaffolded binding site atoms, high model confidence in the associated scaffold, and low similarity (quantified by TM-score) to the natural protein.

between the corresponding designed and target backbone atoms (Methods). Our simulated annealing procedure successfully produces high-confidence designs with low RMSD (< 1.6 Å) across diverse de novo backbones (Figures 5.2d and 5.2e), and can do so reproducibly over different optimization runs (Figure 5.2f).

5.3.2 Partial constraints

We next sought to increase the complexity of our designable space by varying the constraints enforced on different parts of a protein. For example, a simple mixed-constraint setting is to specify a two-domain protein with different combinations of secondary structure composition (Figures 5.2g and A.20a–A.20c). In our programs, we can represent this setting by a syntax tree containing two or more subtrees, where different constraints are only applied within the discrete subtrees.

A more complex mixed-constraint setting is to design functional proteins by constraining one region of the protein design to have the same all-atom positions (including protein side chains) as a functional site from nature, while allowing the design procedure to freely generate the remainder of the protein; this design setting is sometimes referred to as functional site "scaffolding" [159]. Importantly, in contrast to fixed backbone design, in which constraints are only placed on backbone atomic coordinates, functional site scaffolding requires constraints on side-chain atoms as well, since these are critical to achieving function. Because our optimization procedure produces an all-atom structure prediction at each step of the optimization, we can readily incorporate this constraint as part of the energy function by minimizing the all-atom RMSD between the natural and designed atomic coordinates of the functional site (Methods).

Across functional sites involving sequence-contiguous or -discontiguous residues from a variety of natural proteins, our algorithm is able to produce designs that scaffold the site with sub-angstrom RMSD between the experimental and predicted structure in three out of five functional sites attempted (Figures 5.2h and A.20d). Moreover, the algorithm produces designed scaffolds that depart from the native protein (Figure 5.2h). The ability to move natural functional sites onto designed backbones has many practical applications, including the design of functional proteins that are smaller or stabler than their natural counterparts.

5.3.3 Symmetric and multimeric group constraints

Beyond proteins containing partial constraints, we next increase the complexity of our protein designs by generating structures that contain constraints over multiple subunits. A foundational design task for the generation of idealized, de novo proteins is to constrain structural symmetry [26, 160]. To generate symmetric proteins, we first enforce the notion of a repeated unit that is repeated K times when designing a K-fold symmetry (where we can control the value of K). To guide the optimization toward symmetric structures, we add various constraints on the distances among the centroids of each repeated unit as part of the energy function (Methods). In our high-level language, a symmetric protein would be encoded by repeating the same non-terminal symbol K times (corresponding to the repeated unit); the symmetry constraint is then placed at the level of the syntax tree containing these repeated non-terminals (Figure 5.3a).

Using these symmetric constraints, we show that we can program the level of symmetry within a protein design. When directed to design 3- to 8-fold symmetry, the generative model produces a diverse set of high-confidence structures (Figures 5.3b, A.21a, and A.21b), including folds that have common analogs in nature (including coiled-coils, beta propellers, beta sandwiches, beta barrels, and TIM barrels) as well as highly-idealized designs that are different from natural structures, including a pentagonal star-shaped protein (with a TM-score of 0.48 to the nearest PDB structure 3S38; row 1 and column 3 in Figure 5.3b) and a cube-shape protein (nearest-PDB TM-score of 0.51 to PDB 7DEG; row 2 and column

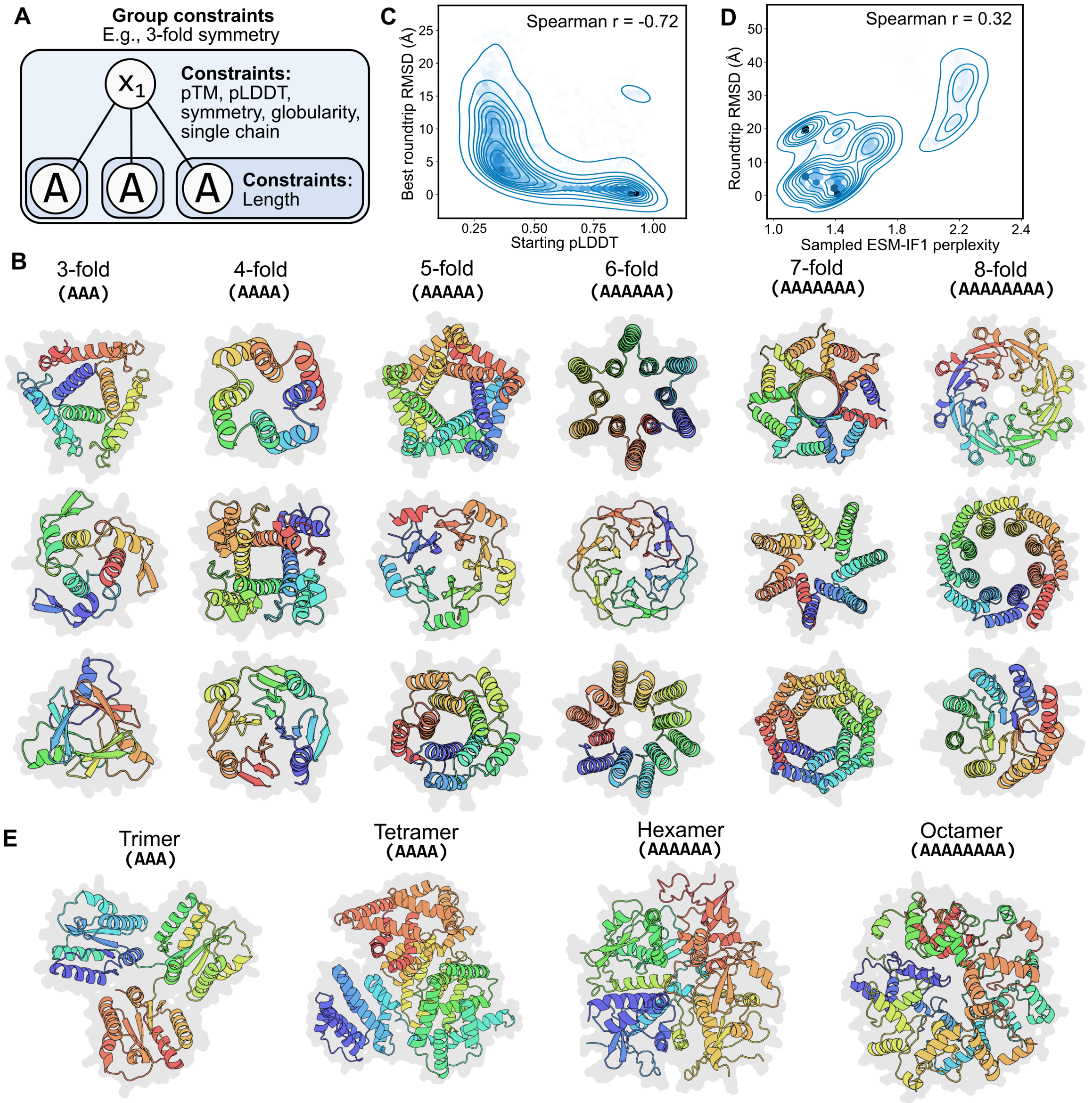

Figure 5.3: *Programming symmetry and homo-oligomerization.* (**A**) A graphical representation of a program for designing a single protein chain with 3-fold symmetry based on a repeated subsequence. (**B**) Example designs varying fold symmetry from 3- to 8-fold. (**C**) 1000 randomly sampled symmetric protein designs were "roundtripped" by sampling ten sequences via ESM-IF1 inverse folding [6] of their backbones followed by ESMFold structure prediction. The ESMFold pLDDT of the starting backbone is indicated on the horizontal axis. The lowest of the 10 RMSDs comparing the starting and roundtripped backbone atoms is indicated on the vertical axis. Blue lines indicate density contours and hexagonal bins are darker with greater density. We observed that a more confident design is associated with roundtrip success. (**D**) 1,000 randomly sampled inverse folding samples are plotted according to their ESM-IF1 perplexity on the horizontal axis and their roundtrip RMSD on the vertical axis. We observed that a lower perplexity sequence is associated with roundtrip success. (**E**) Example homo-oligomers with increasing numbers of individual protomers. The tetrameric, hexameric, and octameric oligomers depicted here form globular polyhedral shapes rather than the rotational symmetry of designs in (**B**).

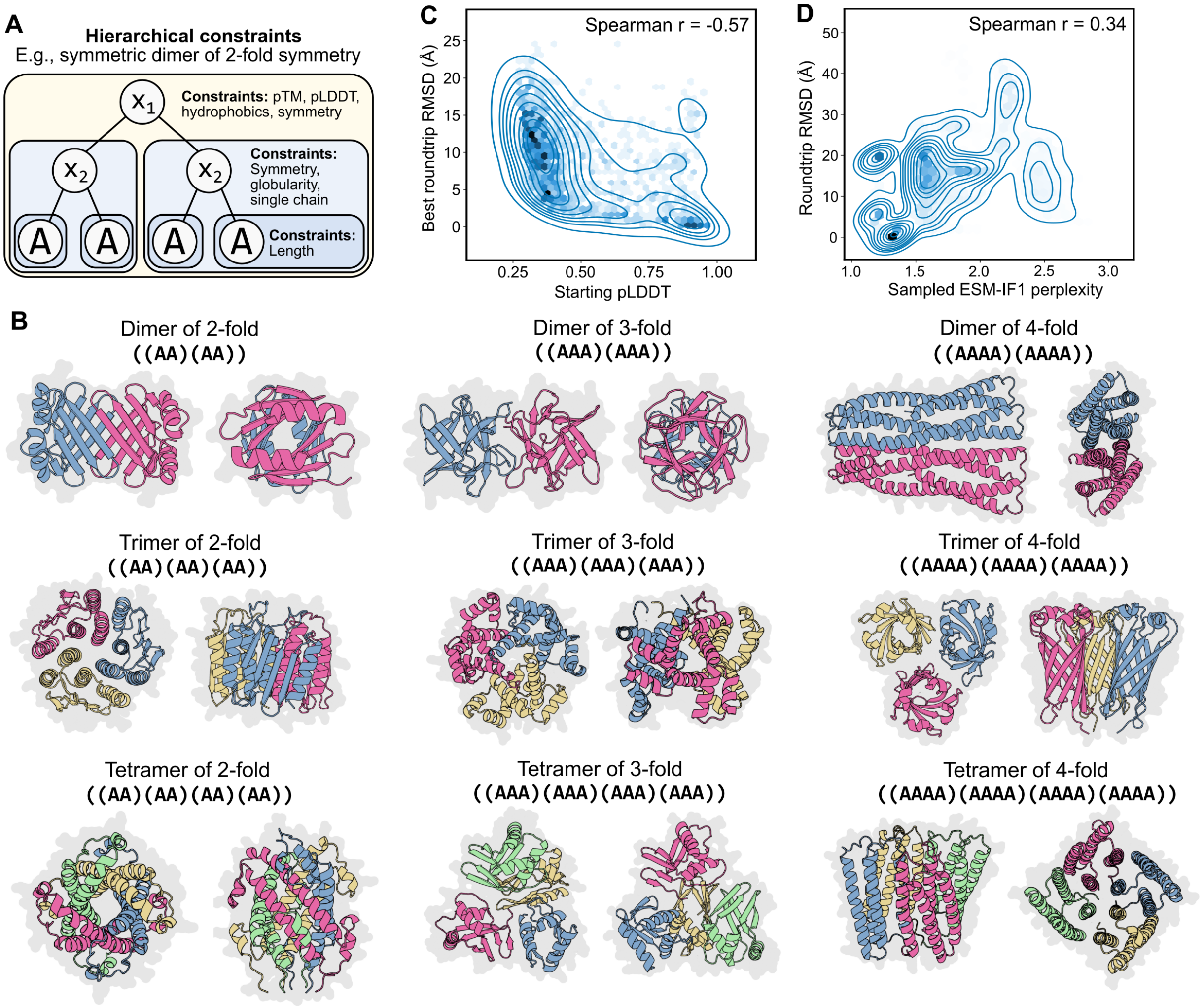

Figure 5.4: *Programming two levels of symmetry.* (**A**) A graphical representation of a program for designing two levels of symmetry in which a homo-oligomeric symmetric dimer represents the top level of symmetry and each unit within the dimer also has two-fold symmetry. (**B**) Example oligomers with two levels of symmetry, in which we procedurally enumerate across a grid in which we vary the top-level symmetry across the rows and the bottom-level symmetry across the columns. Discrete chains are indicated by different colors. (**C**) 1,000 randomly sampled two-level symmetric protein oligomer designs were "roundtripped" by sampling ten sequences via ESM-IF1 inverse folding [6] of their backbones followed by ESMFold structure prediction (Methods). The ESMFold pLDDT of the starting backbone is indicated on the horizontal axis. The lowest of the 10 RMSDs comparing the starting and roundtripped backbone atoms is indicated on the vertical axis. Blue lines indicate density contours and hexagonal bins are darker with greater density. We observed that a more confident design is associated with roundtrip success. (**D**) 1,000 randomly sampled inverse folding samples are plotted according to their ESM-IF1 perplexity on the horizontal axis and their roundtrip RMSD on the vertical axis. We observed that a lower perplexity sequence is associated with roundtrip success.

2 in Figure 5.3b). The highlighted symmetric proteins in Figure 5.3b have nearest-PDB TM-scores ranging from 0.47 to 0.86, with a median TM-score of 0.64; TM-scores for all seeds are plotted in Figure A.21c.

To increase our confidence that these idealized structures correspond to valid and designable backbones, we observe that sampling sequences via inverse folding with the ESM-IF1 and ProteinMPNN models [6, 158] followed by structure prediction of the sequence samples can reproducibly recover the original backbone geometry (Methods), in many instances with sub-angstrom backbone-atom RMSD. To increase our confidence that a successful "roundtrip" through inverse folding indicates designable backbones, we observe that high confidence designs indicates better roundtrip success (Figures 5.3c and A.21d) and that the ability to sample a low perplexity sequence also indicates roundtrip success (Figure 5.3d).

Beyond single-chain symmetries, we can also design multimeric proteins similarly. We enforce the notion of single or multiple chains in our programming language with a "single chain" constraint that dictates that all terminal elements in a subtree belong to the same chain (Methods); to design multimeric proteins, we need only remove this constraint. Example multimeric symmetric proteins involving 4- to 8-mers are provided in Figure 5.3e.

5.3.4 Hierarchical constraints

Formalizing our constraints into a syntax tree naturally enables the specification of hierarchical constraints, which enables more complex protein designs. As an initial demonstration, guided by our high-level language's formalization, we design different levels of symmetry at two levels of hierarchy, where the lower level of symmetry is specified within a chain and the upper level of symmetry is specified among protomers in a homo-oligomer (Figure 5.4a). We procedurally enumerate over examples that range from a dimer of units with 2-fold symmetry to a tetramer of units with 4-fold symmetry (Figures 5.4b, A.22a, and A.22b). As in the single-chain symmetric design setting, we observe successful structure prediction roundtrips

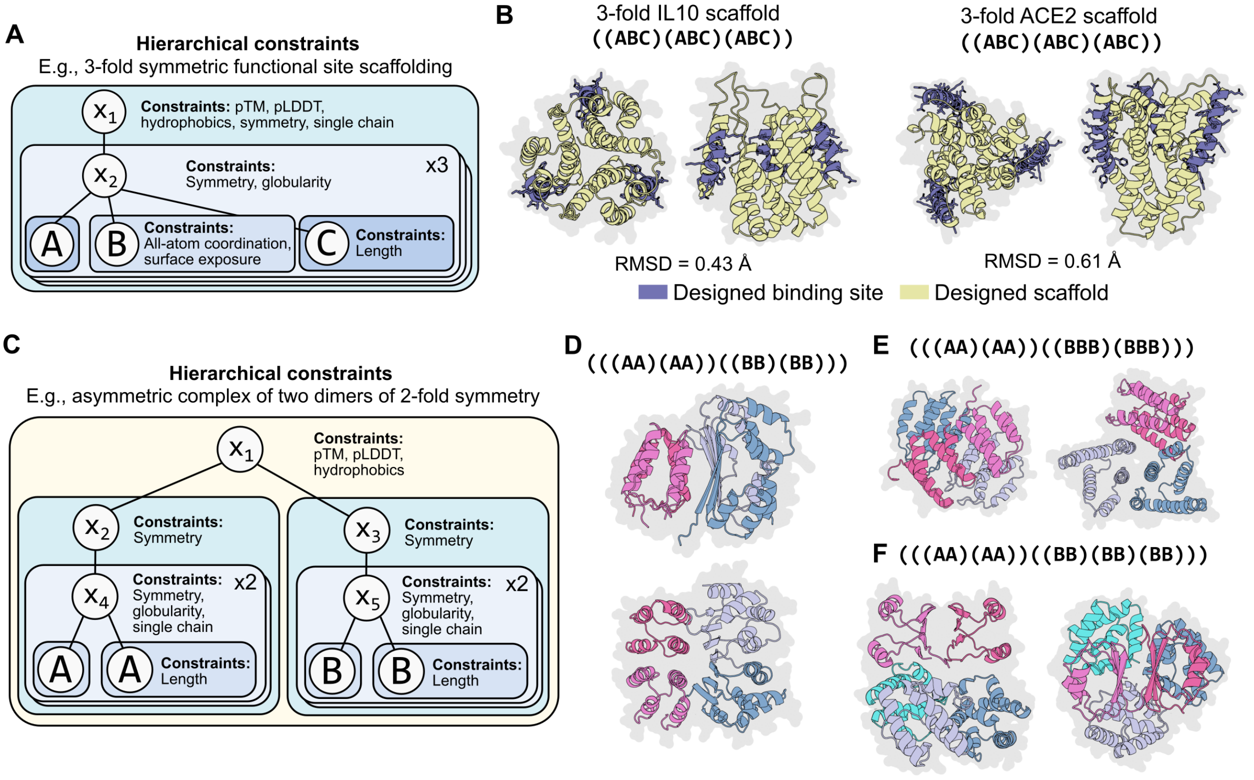
A
Hierarchical constraints
E.g., 3-fold symmetric functional site scaffolding
x_1
Constraints: pTM, pLDDT, hydrophobics, symmetry, single chain
x_2
Constraints: Symmetry, globularity
x3
A
B
Constraints: All-atom coordination, surface exposure
C
Constraints: Length
B
3-fold IL10 scaffold
((ABC)(ABC)(ABC))
3-fold ACE2 scaffold
((ABC)(ABC)(ABC))
RMSD = 0.43 Å
RMSD = 0.61 Å
Designed binding site
Designed scaffold
C
Hierarchical constraints
E.g., asymmetric complex of two dimers of 2-fold symmetry
x_1
Constraints: pTM, pLDDT, hydrophobics
x_2
Constraints: Symmetry
x_3
Constraints: Symmetry
x_4
Constraints: Symmetry, globularity, single chain
x2
x_5
Constraints: Symmetry, globularity, single chain
x2
A
A
Constraints: Length
B
B
Constraints: Length
D
(((AA)(AA))((BB)(BB)))
E
(((AA)(AA))((BBB)(BBB)))
F
(((AA)(AA))((BB)(BB)(BB)))

Figure 5.5: *Programming complex hierarchical constraints.* (**A**) A graphical representation of a program for scaffolding three functional sites in which those sites have a 3-fold symmetry. (**B**) Example 3-fold symmetric scaffolds for the IL10 and ACE2 binding sites that achieve sub-angstrom RMSD averaged across the three sites. (**C**) A graphical representation of a program that specifies an asymmetric protein complex consisting of two pairs of chains. Each pair is constrained to have 2-fold symmetry between the constituent chains. Furthermore, each constituent chain itself has 2-fold symmetry. (**D**) A generated protein structure as specified by the program depicted in (**C**). Discrete chains are indicated by different colors. (**E**) A generated protein structure as specified by the program depicted in (**C**) except where one of the pairs has constituent chains that have three-fold symmetry. Discrete chains are indicated by different colors. (**F**) A generated protein structure as specified by the program depicted in (**C**) except where one of the pairs is replaced with a symmetric trimer (where each constituent chain in the trimer has two-fold symmetry). Discrete chains are indicated by different colors.

through inverse folding, and that both high-confidence predicted structures and low inverse folding perplexity indicate roundtrip success (Figures 5.4c and 5.4d). Many of the designs with two levels of symmetry have low overall similarity to structures in the PDB; for example, a dimer of 2-fold symmetry in which opposing beta sheets form a regular checkerboard pattern (nearest-PDB TM-score of 0.49 to PDB 3W38; row 1 and column 1 in Figure 5.4b). The highlighted homo-oligomers of two-level symmetry in Figure 5.4b have nearest-PDB TM-scores ranging from 0.25 to 0.52, with a median TM-score of 0.48; TM-scores for all seeds are plotted in Figure A.22c.

Another hierarchical design setting is to combine the function-scaffolding and the symmetric design tasks described above, as some functions are enhanced by repetition of a functional site; for example, when improving the strength of a binding interaction, multiple binding sites on a protein could synergize such that the overall binding avidity is greater than the sum of the individual affinities [172]. This task requires two levels of hierarchy: the top level specifies symmetry while the bottom level specifies the side-chain atomic coordination constraint (Figure 5.5a). With this corresponding program, we can generate designs in which an atomic-level constraint is enforced on multiple functional sites over the protein, the overall protein organization is constrained to be symmetric, and we can control the level of designed symmetry (Figures 5.5b and A.23a–A.23c).

We lastly show that we can specify protein designs that have even deeper levels of hierarchy in their constraints (Figure 5.5c) by designing protein assemblies that combine both symmetry and asymmetry. For example, we designed a protein complex composed of four units in which a pair of the chains are symmetric to each other (and each unit internally has two-fold symmetry) and where another pair of chains are symmetric to each other (and each unit also internally has two-fold symmetry), but the two pairs are asymmetric to each other (Figures 5.5c, 5.5d, A.23d, and A.23e). Our high-level programming language readily enables us to control the complexity of the generated complexes such that, for example, one of the pairs consists of chains with three-fold symmetry (Figure 5.5e) or that the complex consists of five chains (a pair of symmetric chains of two-fold symmetry asymmetrically complexed with a triple of symmetric chains of two-fold symmetry) (Figure 5.5f). We find that our optimization procedure can produce designed structures consistent with all of these hierarchical specifications.

5.4 Related work

This paper is related to classical work that attempts to (i) classify a set of common sequence or structure motifs [166, 167] and (ii) combine these motifs to generate new proteins [26, 27, 28, 29, 30, 31, 32, 170]. More recently, deep-learning-based methods have increased the complexity of designable structures [34, 159, 160] and machine-learning-based generative models have shown increasingly sophisticated design capabilities. These include sequence-based Potts models and autoregressive language models for designing sequences [173, 77, 174], Markov Chain Monte Carlo algorithms combined with structure prediction for jointly designing sequences and structures [34, 159, 160], inverse folding models that use structural backbone coordinates to design sequences [6, 158], and concurrent work using diffusion models for designing protein backbones [175, 176]. A key contribution of this study is to combine the modularity aspired to by classical methods with the power of modern generative models, in particular leveraging improvements in the accuracy and efficiency of language-model-based protein structure prediction [171].

5.5 Discussion

In this study, we show that generative artificial intelligence enables high-level programmability at a new level of combinatorial complexity. We propose a programming language that can express high-level programs for the design of proteins at diverse biological scales, including atomic-level coordinates, secondary structure, and high-level symmetries within single chains and the units of self-assembling multi-chain complexes. We show that programs written in the abstract language can be compiled into an energy function and that the corresponding generative model is capable of fulfilling complex constraints within an overall coherent structure.

We demonstrate programs of increasing levels of complexity, including the design of homo-oligomers with two levels of symmetry, symmetric functional scaffolds, and asymmetric complexes of subunits that themselves have two levels of symmetry. The approach reveals a large space of idealized protein designs created from top-down design principles. Especially as the complexity of the constraints increases, many of the corresponding designs are highly idealized, analogous to the regularity of artificially created machines and systems.

Our computational results using two independent inverse folding methods suggest that the generated structures are designable, since inverse folding models have demonstrated high experimental success rates [158]. We are also obtaining data to experimentally validate the designs.

More broadly, the formalization offered by a high-level programming language enables logical design principles to be applied to protein design as in other fields of engineering. This has been especially challenging in biology due to the way that the amino acid sequence opaquely encodes the structure and function. As protein design moves toward the engineering of more complex functions, we anticipate that such a system will become increasingly useful.

6 | Conclusion

This dissertation started with the idea of modeling protein sequences and structures via four topics. To reiterate, these are (a) learning distributions of protein sequences, (b) learning distributions of protein structures, (c) predicting protein sequence from a given protein structure backbone, and (d) predicting the protein structure directly from the protein sequence alone.

We showed that neural models are capable of modeling proteins holistically. By using a VQ-VAE, we are able to model the distribution of protein structures. Generating synthetic data allows us to build models of protein sequences given a structure and achieve state of the art fixed backbone protein design results. Scaling language models allow us to build distributions of protein sequences, and building a structure prediction model on top allows us to make predictions of protein structures given a sequence. These models are shown to be capable of generalizing outside of natural proteins, to make predictions that cannot be found in genetic or structural databases. Finally, we jointly design proteins via inverting our structure prediction model.

Throughout this work, we released a set of open sourced tools to be used by biologists in downstream applications. ESM-IF1 is an inverse folding model that takes advantage of additional data to improve on existing models. ESM2 and ESMFold are respectively an open sourced protein language model and structure prediction model. ESM2 is a capable representation learning model that can be finetuned on a multitude of tasks such as property prediction, mutation effect prediction, binding, and so on. ESMFold has been similarly integrated into various tools such as HuggingFace and ChimeraX. The speed of the model enables webservers to provide protein structure predictions almost instantaneously, giving scientists the unprecedented option of examining a high quality guess at a protein structure the moment the sequence is known.

ESM2 and ESMFold was very well received by the academic community. To name a few examples, Zheng et al. [177] incorporates ESM2 to improve fixed-backbone protein design, and Fang et al. [178] improves protein binding site prediction with ESMFold. ESMFold has been used in the processes of designing a PD-L1 binder [179], in studying malignant melanoma [180], identifying the likelihood of a de novo protein design in folding properly [181], and in studying a RNA virus [182] . ESM embeddings have also been used in B-cell epitote prediction [183], improving RNA-seq cell embeddings [184], and automated model building in CryoEM maps [185]. We hope our models will continue to contribute to biology and eventually lead to improvements in human health and well being.

6.0.1 Future Work

There has been enormous progress on modeling proteins since the start of this dissertation. We are now able to generate arbitrary protein chains, condition the generation on arbitrary constraints, and design proteins to perform any function we choose. Using protein language models, in silico design can progress significantly faster. It would seem to some that protein modeling is largely a solved science, but in actuality, we still have a lot of work ahead of us.

Protein chains are often the building blocks of large molecular machines. Being capable of modeling single chains does not allow us to build the complex machines that transcribe DNA into RNA (DNA Polymerase) or break apart proteins (proteosome). As said in the introduction, current data sources depend on crystallography, which by nature imposes that the protein structures must be static. Working on large complexes will necessitate improvements in data collection and molecular imaging techniques. Along with complexes, it is also difficult to produce data for small molecules, DNA, and RNA. Again, better data is key to learning good distributions over these other molecules that proteins interact with.

Additionally, even though learning sequence distributions over natural sequences correlate to function, it's unclear how much this extends to adversarial systems in biology and de novo proteins. For example, viruses and antibodies are in constant co-evolution with each other, continually evolving to bypass each other's attacks and defenses. A fixed point of "higher function" might not exist for such systems. Additionally, we might want to design proteins that really break correlation with evolutionary fitness by creating proteins that work in environments not known to nature. Feedback through wet-lab experimentation will be important to understand these proteins.

Finally, the explosion in biological data in single cell RNA-seq, cryo EM, and epigenetic measurements means that there are other modalities to consider in order to better understand biology. The interactions between genes, proteins, and cellular environments is a web that cannot be understood through studying proteins alone. Many important diseases are only understood from interactions between multiple protein targets, and drugs can only be designed by considering the full system. Studying these other modalities along with protein modeling will be key to engineering future biological systems.

6.0.2 Retrospective

Work on protein understanding has been continuous since the start of this dissertation. When I started working on this topic in early 2019, state of the art methods often used the dynamic programming techniques of MSAs, and protein understanding was restricted to the family of proteins around a sequence. Structure prediction was still a "grand challenge," and it was hard to imagine that in a few years, designing proteins with arbitrary constraints can be done in an afternoon. A PhD student can now create structure prediction models, and generative models of protein structures have become easy to build.

However, there's still a big gap between what can be done on a computer and wet-lab experimentation. It's still hard to obtain protein structures for sequences, to perform assays on arbitrary sequences, and to fully design one that can work as a drug. We are really just entering into the age where biology is becoming a field of engineering, and I am excited to see what continued work in this field can bring.

A | Appendix

A.1 Deep generative models of protein structure create new and diverse proteins

A.1.1 Training and model details

We train an encoder-decoder convolutional network in almost all cases. In all cases we use a residual network (resnet) for the encoder. The encoder resnet is a stack of several bottleneck residual blocks [45] followed by a downsample, repeated 7 times until we obtain a 1×1 map, which is fed into the decoder as the initial latent variable. The HVAE and VQ-VAE have shortcut connections at coarser resolutions from the encoder to the decoder, where each shortcut connection is either a Vector Quantization layer or stochastic sampling layer via the reparameterization trick. These intermediate latent variables are output at up to a 32×32 resolution.

Model selection was done in all cases by a randomized hyperparameter sweep, generating 30 structures, and selecting the model with the lowest Rosetta energy of that model class.

The output of the decoder trunk is always routed to an individual 2 layer MLP to output the final predictions for each of the distogram and (interresidue or backbone) angle predictions. Backbone angles are predicted via averaging across the heights and width of the distogram, and then averaging the logits obtained from each of the height and width predictions.

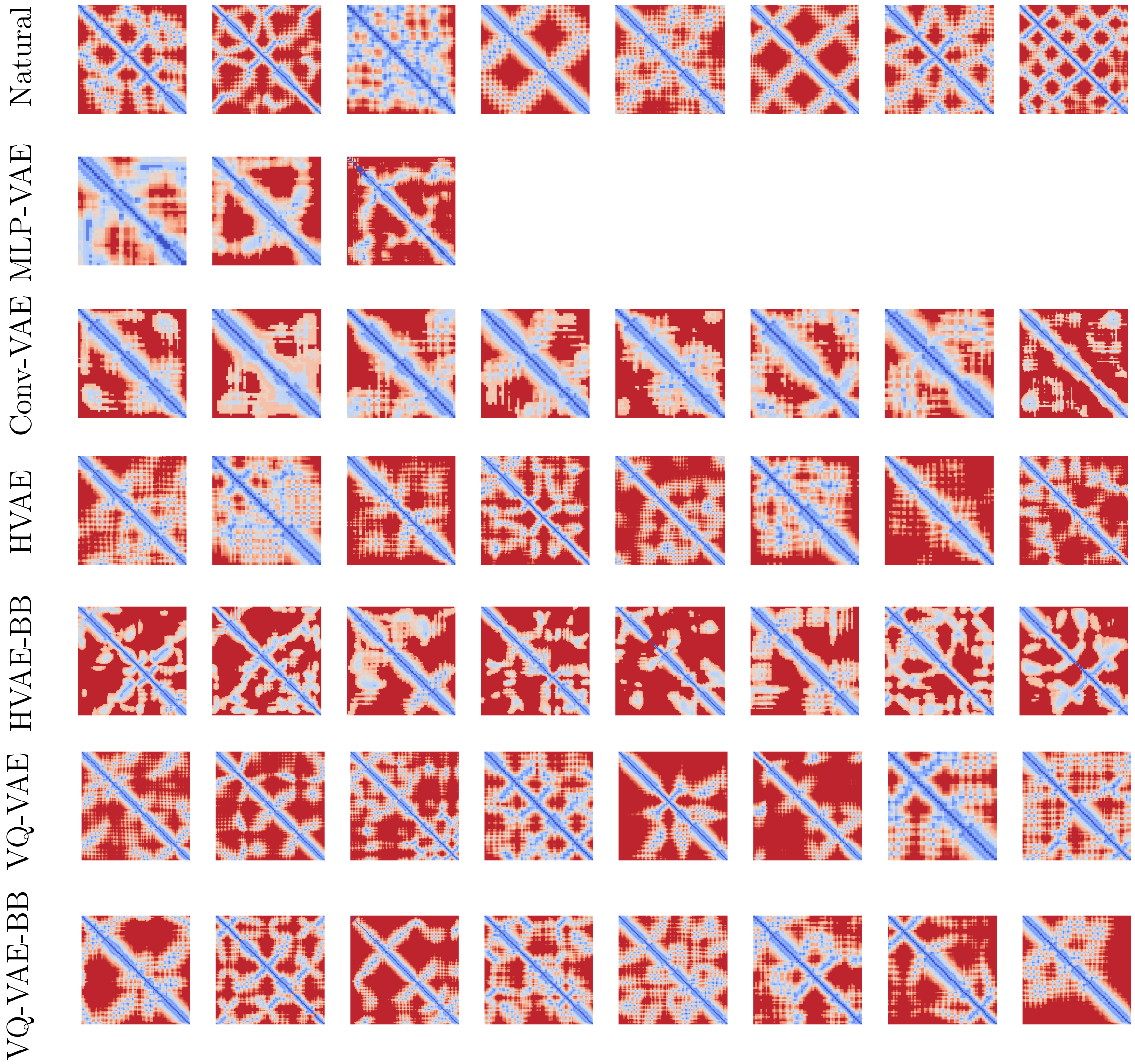

Figure A.1: Top Distograms: we filter generations to negative rosetta energy and greater than 1 MR/LR contacts per residue. These are a random sample of such distograms

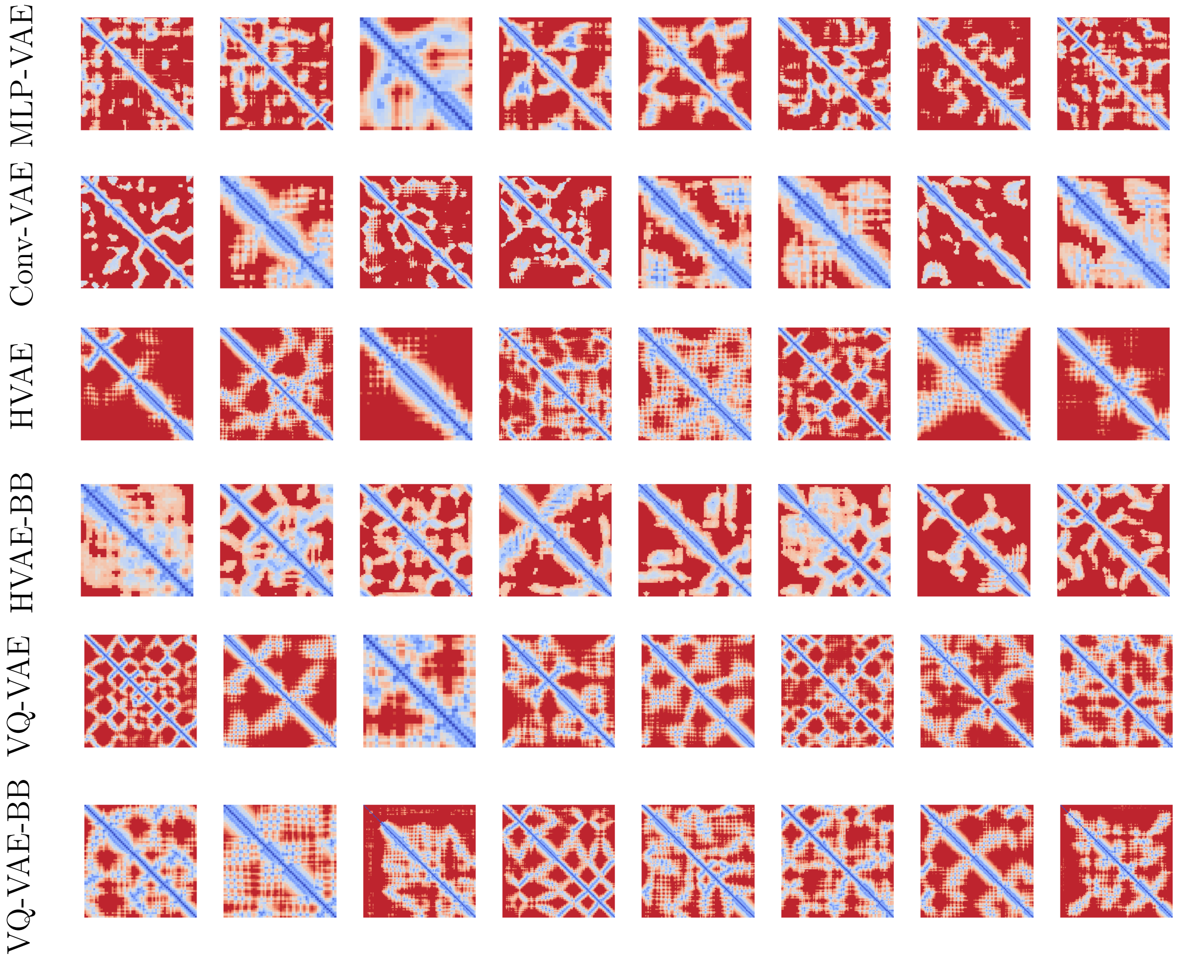

Figure A.2: Random distograms: We show a random sample of all generated distograms by each model

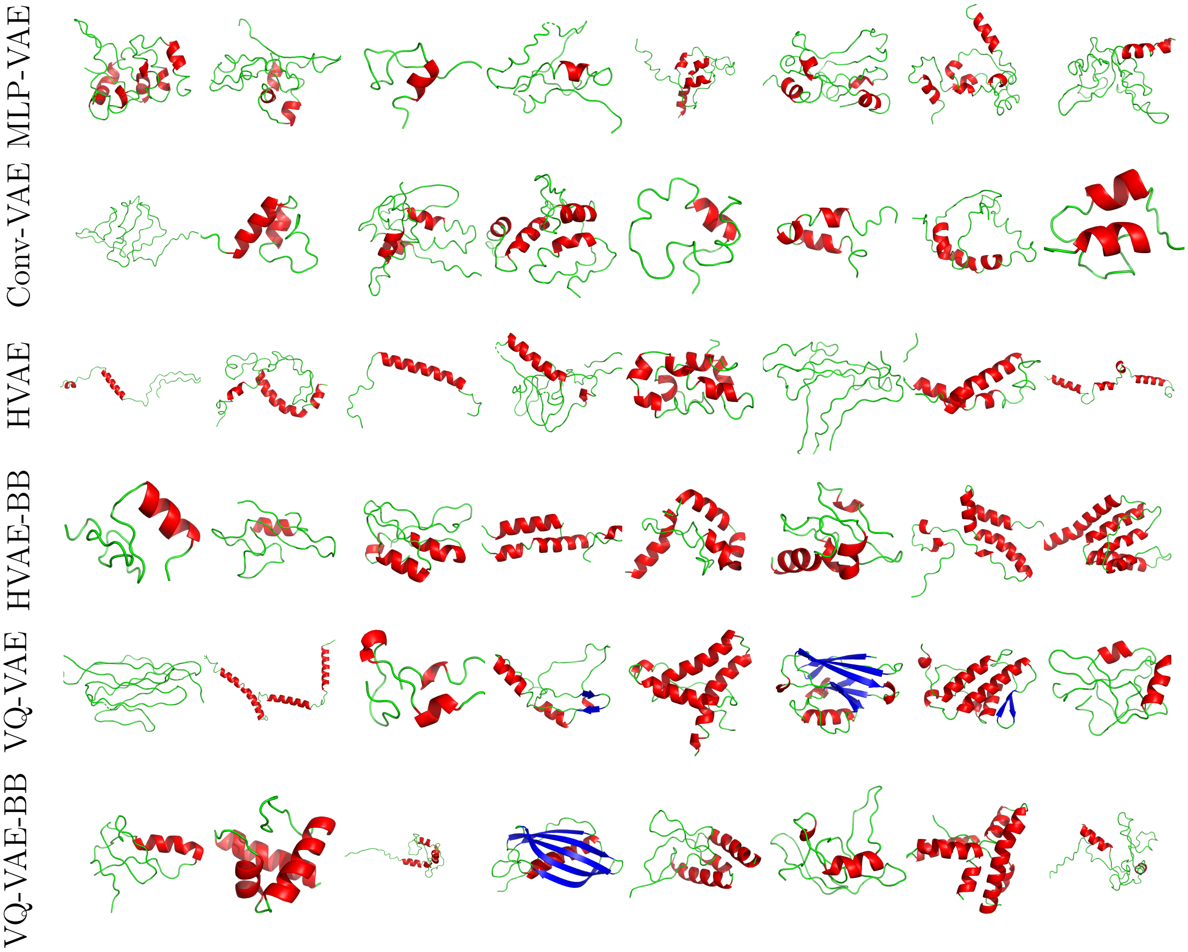

Figure A.3: Random structures: We show, correspondingly to Figure A.2, the results of constrained folding on those distograms. We color α-helices red, β-sheets blue, and coils green. Most baseline generations are unstructed coils.

We also found that for spatial latent variables, it was important to symmetrize the latent variables post-hoc, as well as symmetrizing the predictions via a simple average.

We explored the following hyperparameters for all models:

- Architecture (see next paragraphs). This is defined by the number of residual blocks per resolution ("stack" of residual blocks, between each of the 7 downsampling steps).
- Initial channel dimension (64, 128, 196, 256).
- Most resnets increase in the channel dimension as the encoder decreases in resolution, and vice versa for the decoder. We varied with which layer to start increasing the channel dimension. We start doubling the channel dimension at resolution (32×32, 16×16,8×8)
- Parameters in distogram, angle, and sequence prediction branches. We tried a simple linear model, a MLP, and a few layers of dilated convolutions, as well as varying widths. We chose the MLP head in all cases.
- Representation of the output space. We also implemented the discrete logistic mixture distribution from [186], but we choose the typical categorical distribution.
- We use Adam in all cases but sweep across different learning rates. We selected $e * 10^{-4}$ in all cases, as it seemed to train stably for all models.

Architecture specification In this section, we will specify the encoder and decoder with a list of 8 numbers, corresponding to the number of bottleneck residual blocks [187] between the 7 downsampling steps. The first number always refers to the blocks that work at a 128×128 resolution, and the last number always refers to the block that works at a 1×1 resolution. There is a downsampling operation between the stacks of blocks. We mainly varied whether the majority of the compute is at the coarse or fine resolutions. The final choices are specified below.

Conv-VAE The encoder is the same as MLP-VAE. The decoder consists of 61 convolutional blocks of size 3×3. We upsample and replicate the single latent variable to the 128×128 sized output concatenated with a Sinusoidal positional embedding. This model is 89M parameters.

For MLP-VAE and Conv-VAE baselines, we experimented with annealing the coefficient on the KL divergence term from 0 at the start of training. We found that annealing was critical to stable training - the cyclic annealing schedule of [188] was helpful in stable model training across a range of hyperparameters, so this was used in all models reported, with 4 cycles through the first half of the training. We experimented with large models, batch sizes, learning rates, and latent variable sizes. We also experimented with VAE variants like β-VAE [189] and Deterministic Regularized Autoencoders [190], but found little effect on the generation quality.

MLP-VAE The encoder is a stack of convolutional blocks: `[3, 3, 3, 3, 3, 3]`, with an intial hidden size of 64 and doubling every time a downsample.

For MLP-VAE, we found it impossible to train with a flattened structure. Taking inspiration from [46], we run three residual MLP layers, across the height, width, and channel dimensions. We found that this was able to converge more consistently.

Define an residual MLP block as a sequence of (LayerNorm, Linear, GeLU, Linear) operations. A MLP stack is 3 MLP blocks, the first over the height dimension, the second over the width dimension, and the third over the channel dimension.

We upsample and replicate the single latent variable to the 128×128 sized output concatenated with a Sinusoidal positional embedding. The decoder is then 21 MLP stacks with a hidden dimension of 128. This model is 88M parameters.

HVAE The HVAE model, results in a step function in generation quality over previous models, generating a much larger proportion of α-helices.

In order to implement the hierachical VAE as specified in [50, 51], we used an upsampling convolutional decoder. Between each stack, there is an upsampling operation. In both HVAE and HVAE-BB, we output a 1×1 latent variable, and 2 sets of latent variables at a 4×4 resolution, one at the start of the stack and one at the end.

The encoder for HVAE is `[4, 4, 4, 8, 8, 32, 4, 1]`. The decoder is `[4, 4, 4, 8, 12, 32, 4, 1]`. The model is 101M parameters.

The encoder for HVAE-BB is `[1, 1, 4, 8, 8, 24, 20, 1]`. The decoder is `[1, 4, 8, 12, 12, 28, 20, 1]`. The model is 115M parameters.

The model architectures were determined via random hyperparameter sweeps as specified above. We also tried to vary the number of output latent variables, trying up to 20. The reconstruction loss and ELBO did indeed improve with the number of latent variable scales, but the generation quality did not.

VQ-VAE Similar to the HVAE, we output a latent variable at specific scales. This latent variable is quantized through the Vector Quantization layer of [52].

For the decoder, we stack all encoded latent variables, upsampled to the finest resolution output by the encoder, and run it through the same upsampling convolutional blocks as described above. We use 512 quantized latent variables with dimensionality 64 in all cases.

For VQ-VAE, the encoder is `[2, 2, 2, 2, 2, 2, 2, 1]`, outputting latent variables at the 1×1 and 32×32 scales. The decoder is `[4, 4, 4, 4]`, since the input starts at the 32×32 scale. The model is 114M parameters.

For VQ-VAE-BB, the encoder is `[2, 2, 2, 2, 2, 2, 2, 1]`, outputting latent variables at the 1×1, 4×4 and 16×16 scales. The decoder is `[4, 4, 4, 4]`, since the input starts at the 32×32 scale. The model is 115M parameters.

In both cases, for $p_\theta(x)$, we use an autoregressive, 6 layer vanilla transformer decoder [89] to learn the prior. We use 6 layers with a channel dimension of 512 over 8 heads. This worked robustly, so we only searched through dropout in order to better regularize our models. However, we found that models overfit to the training set provided better generations, so we report only results with the overfit priors in this work. More exploration to explain this phenomenon is needed. The learned priors are 20M parameters in all cases.

Training procedure We use stratified sampling at the family level during training. Stratified sampling has been shown to be critical to training large protein language models, though we did not explore uniform sampling [16]. We also tested results on fold and superfamily-level structural heldout sets, though we found that there was too much noise given the small dataset. Furthermore, since their losses were not correlated to Rosetta folding energies, so we decided not to report on their exact numerical values.

We fix models to work on 128 length proteins. Larger proteins are ignored, as we theorized that sequence-level cropping would compromise structural integrity. Smaller proteins are centered and padded.

Variable length generation We use a heuristic to generate structures of varying lengths. We fit a head on the amino acid identity of each position, and we train with a "out-of-bounds" label for padding tokens. Then, during generation, we noticed that the model consistently generated centered distograms. Therefore, we used the heuristic that the generated protein starts at the first position when $p(\text{out-of-bounds}) < .2$, and ends at the last position when $p(\text{out-of-bounds}) < .2$.

Compute Costs To find optimal hyperparameters, we used tens of GPUs for 2 days at a time. Most models can be trained on a single V100 GPU in a few days. Each `FastDesign` decoy takes from 3-6 hours to run on a single core. Therefore, one round of designs for 40 generations takes approximately 1,500 CPU days.

A.1.2 Structure generation details

Constraints on interresidue distances and orientations We follow the procedure of Yang, et al. [25] to incorporate constraints into Rosetta folding, with some variations:

1. Generate constraints given the distogram and interresidue angles. (Renormalize distrogram such that the minimum logit is -10).
2. Coarse grained folding (9 independent trajectories using centroid energy function). Followed by fine grained all-atom fold of all 9 trajectories.
3. Best trajectory selected by Rosetta energy.

Constraints on backbone dihedral angles Some models incorporate constraints on the backbone dihedral angles rather than the interresidue orientations. We discretize the phi/psi dihedral angles of the backbone into a 25 by 25 grid. Then 10000 samples are drawn from the grid, and a Von Mises distribution is fit independently to the marginals for phi and psi. The constraints are upweighted by a factor of 1000 to balance them with the distance contraints.

A.1.3 Design Verification

Given a generated structure, first we run `FastDesign` to generate up to 200 sequences. We run it with RosettaScripts [191], allowing all amino acids and extra rotamer angles with ALLAA EX 1 EX_CUTOFF 3. We also use the linear memory interaction graph to conserve memory, and default databases provided by Rosseta.

We run AlphaFold with default parameters. We randomly choose the predicted LDDT $>$ 0.7 threshold - on average it means the model should be correct until up to 2Å. Only one structure that passes the predicted LDDT > 0.7 threshold did not agree with Alpafold. For many of the structures that Alphafold was not confident on, we discover close matches in PDB, so it may be the case that `FastDesign` was not able to find a suitable sequence for our generated structure.

Figure 2.6 shows many of the designed structures have very low sequence identity with Uniref90. These are unfiltered designs, so it is unclear if they are all realizable, although in Figure 2.5 we show that most structures agreed on by our design and AlphaFold have no MSAs in common.

A.2 Learning inverse folding from millions of predicted structures

A.2.1 Additional details on datasets, training procedures, and model architectures

A.2.1.1 Details on dataset of predicted structures

We used training data from two sources: 1) experimental protein structures from the CATH 40% non-redundant chain set, and 2) AlphaFold2-predicted structures from UniRef50 sequences. To evaluate the generalization performance across different protein folds, we split the train, validation, and test data based on the CATH hierarchical classification of protein structures [87] for both data sources. To achieve that a rigorous structural hold-out, we additionally use foldseek [5] for pairwise TMalign between the test set the train set.

CATH topology split. Following the structural split methodology in previous work [22, 23, 72], we randomly split the CATH v4.3 (latest version) topology classification codes into train, validation, and test sets at a 80/10/10 ratio. The CATH [87] structural hierarchy, classifies domains in four levels: Class (C), Architecture (A), Topology/fold (T), and Homologous superfamily (H). The topology/fold (T) level roughly corresponds to the SCOP fold classification.

Experimental structures. We collected full chains up to length 500 for all domains in the CATH v4.3 40% sequence identity non-redundant set. The experimental structure data contained only stand-alone chains and no multichain complexes. As each chain may be classified with more than one topology codes, we further removed chains with topology codes spanning different splits, so that there is no overlap in topology codes between train, validation, and test. This results in 16,153 chains in the train split, 1457 chains in the validation split, and 1797 chains in the test split.

Predicted structures. We curated a new data set of AlphaFold2 [3]-predicted structures for a selective subset of UniRef50 (202001) sequences. To prevent information leakage about the test set from the predicted structures, we proceeded in the following steps.

First, we annotated UniRef50 sequences with CATH classification according to the Gene3D [88] database, also used by Strokach [72] for data curation. Gene3D represents each CATH classification code as a library of representative profile HMMs. We searched all HMMs associated with the validation and test splits against the UniRef50 sequences using default parameters in hmmsearch [192] and excluded all hits.

Additionally, as AlphaFold2 predictions use multiple sequence alignments (MSAs) as inputs, we also took precaution to avoid information leakage from sequences in the MSAs. We created a filtered version of UniRef100 by searching all the validation-split and test-split Gene3D HMMs against UniRef100 (202001) and excluding all hits. Then, we constructed our MSAs using hhblits [193] on this filtered version of UniRef100. While this filtering step was out of precaution, in retrospect it was perhaps unlikely for the MSA inputs to AlphaFold2 to leak information as the MSAs themselves were not seen during training. The filtering step may have negatively impacted the quality of the resulting predicted structures, although empirically only a very small percentage of MSA sequences were filtered out.

As AlphaFold2 predictions are computationally costly, our budget only allowed for predicting structures for a subset of the UniRef50 sequences. We ranked UniRef50 sequences based on the distogram lDDT score, based on distogram predictions from MSATransformer [85], as a proxy for the quality of predicted structures. While the original distogram lDDT score (Supplementary Equation 6 in [55]) is based on pairwise distances from native protein structures, in the absence of native structures we use the argmax of pairwise distances instead, effectively measuring the "sharpness" of distograms and prioritizing sharper predictions. In this order, using AlphaFold2 Model 1 on the filtered UniRef100 MSAs described above, we obtained predicted structures for the top 12 million UniRef50 sequences under length 500, roughly 750 times the CATH train set size.

We used the publicly released model weights from AlphaFold2 Model 1 for CASP14 as a single model, as opposed the 5-model ensemble in [3], to cover more sequences with the same amount of computing resources. We curated the input MSAs from UniRef100 with hhblits, with an additional filtering step as described above. To reduce computational costs, compared to the standard AlphaFold2 protocol, we did not include the UniRef90 jackhmmer MSAs, or the MGnify and BFD metagenomics MSAs, nor the pdb70 templates. Other than a reduced inputs, we followed the default settings in AlphaFold2 open source code, using 3 recycling iterations and the default Amber relaxation protocol. Despite the reduced inputs, the resulting 12 million predicted structures still have high pLDDT scores from AlphaFold, with 75% of residues having pLDDT above 90 (highly confident).

We found that increasing the predicted data size to up to 1 million structures (75 times the CATH experimental data size) substantially improves model performance. Beyond 1 million structures, models still benefit from more data but with diminished marginal returns (Figure 3.6a).

	GVP-GNN	GVP-GNN-large	GVP-Transformer
GVP-GNN embedding dim (node)	(100, 16)	(256, 64)	(1024, 256)
GVP-GNN embedding dim (edge)	(32, 1)	(32, 1)	(32, 1)
Top K neighbors in GVP-GNN	30	30	30
GVP-GNN encoder layers	3	8	4
GVP-GNN decoder layers	3	8	
Transformer embedding dim			512
Feedforward embedding dim			2048
Attention heads			8
Transformer encoder layers			8
Transformer decoder layers			8
Total number of parameters	**1M**	**21M**	**142M**
Batch size (tokens per GPU)	3072	4096	4096
GPUs	1	32	32
CATH:AF2 mixing ratio	1:0	40:1	80:1
Epochs until convergence	84	368	178
Train time per epoch (GPU hours)	0.07	24	88
Total train time (GPU days)	**0.2**	**368**	**653**
Optimizer	Adam	Adam	Adam
Learning rate schedule	Constant	Inverse square root	Inverse square root
Peak learning rate	1.0E-03	1.0E-03	1.0E-03
Initial learning rate		1.0E-07	1.0E-07
Warm-up updates		5000	5000
Gradient clipping	4.0	1	

Table A.1: Details on model hyperparameters and training.

Noise on AlphaFold2-predicted backbone coordinates. Even after Amber relaxation, the backbone coordinates predicted by AlphaFold2 contain artifacts in the sub-Angstrom scale that may give away amino acid identities. Without adding noise on predicted structures, there is a substantial gap between held-out set performance on predicted structures and on experimental structures. To prevent the model from learning non-generalizable AlphaFold2-specific rules, we added Gaussian noise at the 0.1A scale on predicted backbone coordinates. The Gaussian noise improves the invariant Transformer performance but not the GVP-GNN performance (Supplementary Figure A.3).

A.2.1.2 Details on span masking

We add a binary feature indicating whether each coordinate is masked or not. In GVP-Transformer, we exclude the masked nodes in the GVP-GNN encoder layers, and then impute zeros when passing the GVP-GNN outputs into the main Transformer. Imputing zeros for missing vector features ensure the rotation- and translation- invariance of the model. In GVP-GNN, we impute zeros for the input vector features, and in the input graph connect the masked nodes to the k sequence nearest-neighbors ($k = 30$) in lieu of the k nearest nodes by spatial distance.

For span masking, we randomly select continuous spans of up to 30 amino acids until 15% of input backbone coordinates are masked. Such a span masking scheme has shown to improve performance on natural language processing benchmarks [90]. The span lengths are sampled from a geometric distribution $\text{Geo}(p)$ where $p = 0.05$ (corresponding to an average span length of $1/p = 20$). The starting points for the spans are uniformly randomly sampled. Compared to independent random masking, span masking is better for GVP-Transformer but not for GVP-GNN (Table A.3).

For the amino acids with masked coordinates, we exclude the corresponding nodes from the input graph to the pre-processing GVP message passing layers, and then impute zeros for the geometric features when passing the GVP outputs into the main Transformer. Imputing zeros for missing vector features ensure the rotation- and translation- invariance of the model.

A.2.1.3 Details on model architectures

Autoregressive modeling. GVP-GNN and GVP-Transformer both have encoder-decoder architectures. The encoder only receives the structural features. The decoder receives the encoder output along with the one-hot encoding of the amino acids. In the autoregresive decoder, sequence information only propagate from amino acid i to j for $i < j$.

The last decoder layer produces a 20-way scalar output per position and softmax activation to predict the probabilities for the amino acid identity at the next position in the sequence.

Invariance to rotation and translation. The input features for both GVP-GNN and GVP-Transformer are translation-invariant, making the overall models also invariant to translations.

Each GVP-GNN layer is rotation-equivariant, that is, for a vector feature x and any arbitrary rotation T, $Tf(x) = f(Tx)$. With equivariant intermediate layers and an invariant output projection layer, GVP-GNN is overall invariant to rotations, since the composition of an equivariant function f with an invariant function g produces an invariant function $g(f(x))$.

The GVP-Transformer architecture is also invariant to rotations and translations. The initial GVP-GNN layers in GVP-Transformer output rotation-invariant scalar features and rotation-equivariant vector features for each amino acid. To make the overall GVP-Transformer invariant, we perform a change of basis on GVP-GNN vector outputs to produce rotation-invariant features for the Transformer. More specifically, for each amino acid, we define a local reference frame based on the N, CA, and C atom positions in the amino acid, following Algorithm 21 in AlphaFold2 [3]. We then perform a change of basis according to this local reference frame, rotating the vector features in GVP-GNN outputs into the local reference frames of each amino acid. (If GVP-GNN outputs are used directly as Transformer inputs without this change of basis, the GVP-Transformer model would not be rotation-invariant.) We concatenate this rotated "local version" of vector features together with the scalar features as inputs to the Transformer. The concatenated features are invariant to both translations and rotations on the input backbone coordinates, forming a $L \times E$ matrix where L is the number of amino acids in the protein backbone and E is the feature dimension. For amino acids with masked or missing coordinates, the features are imputed as zeros.

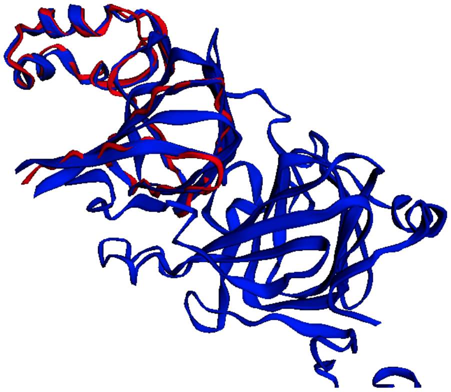

Figure A.4: An illustrative example of structural overlap between CATH topology splits. The jack bean canavalin (PDB code 1DGW; chain Y; red) and the soybean β-Conglycinin (PDB code 1UIJ; chain B; blue) are assigned different topology codes in CATH (1.10.10 and 2.60.120), but they align with TM-score 0.94 and CA RMSD 0.7A on a segment of 90 residues. The difference in topology classifications likely resulted from CATH annotating only a 37-residue mainly helical segment of the jack bean canavalin as a domain while annotating a longer 176-residue mainly beta sheet segment of the soybean β-Conglycinin as a domain.

Transformer. We closely followed the original autoregressive encoder-decoder Transformer architecture [89] except for using learned positional embeddings instead of sinusoidal positional embeddings, attention dropout, and layer normalization inside the residual blocks ("pre-layernorm"). For model scaling experiments, we followed the model sizes in [194], and chose the 142-million-parameter model with 8 encoder layers, 8 decoder layers, 8 attention heads, and embedding dimension 512 based on the best validation set performance (Figure 3.6c shows test set ablation).

The GVP-GNN, GVP-GNN-large, and GVP-Transformer models used in the evaluations in this manuscript are all trained to convergence, with detailed hyperparameters listed in Table A.1.

A.2.2 TM-score-based test set

In addition to the CATH topology-based test set following previous work [22, 23], we also create an even more stringent test set based on pairwise TM-score comparison between train

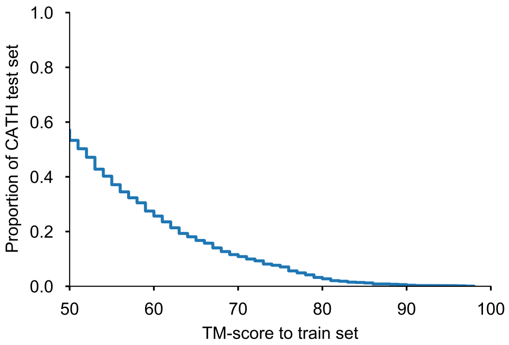

Figure A.5: Distribution of the highest TM-score from each test example to the train set. For example, 54% of the CATH topology split test set has at least one match in the train set with TM-score above 0.5, and 27% of the topology split test set has at least one match in the train set with TM-score above 0.6.

		Perplexity			Recovery %		
Model	Data	Short	Single-chain	All	Short	Single-chain	All
Structured GNN	CATH	10.08	7.04	7.06	27.8%	35.1%	35.4%
GVP-GNN	CATH	8.13	5.76	5.86	31.5%	41.1%	40.4%
	+ AlphaFold2	9.87	6.61	6.50	26.3%	36.3%.	36.8%
GVP-GNN-large	CATH	8.87	6.62	6.68	31.0%	37.2%	37.4%
	+ AlphaFold2	7.08	4.46	4.39	**34.1%**	48.2%	48.7%
GVP-Transformer	CATH	8.80	6.78	6.97	28.5%	36.7%	36.3%
	+ AlphaFold2	**6.99**	**4.36**	**4.34**	33.0%	**48.9%**	**49.5%**

Table A.2: Fixed backbone sequence design performance on the more stringent structurally held-out test set from CATH v4.3 chains (and its short and single-chain subsets) in terms of per-residue perplexity (lower is better) and recovery (higher is better).

and test examples. The CATH topology split does not completely prevent high TM-score matches between train and test structures. We illustrate such an example in Figure A.4, and show overall TM-score statistics Figure A.5.

We constructed a TM-score-based test set of 223 proteins with no TMalign matches (TM-score ≥ 0.5) from the train set, using the foldseek [5] TMalign tool with default parameters for the pairwise search.

We found that the conclusions about model performance overall remains the same on this TM-score-based test set as on the CATH topology split test set. For consistency with prior work, we report metrics on the CATH topology test set in the main manuscript, while showing metrics on the smaller TM-score-based test set in Table A.2.

A.2.3 Additional results and details

Ablation on noise and masking during training. We found that GVP-Transformer models trained with Gaussian noise during training perform slightly better at test time than those trained without (Table A.3). When given full backbone coordinates at test time, training with span masking only very slightly improves model performance compared to no masking or to random masking, even though there is a much larger performance gap between random masking and span masking on regions with masked backbone coordinates (Figure 3.4).

Dual-state design test set from PDBFlex. We test design performance on multiple conformations by finding test split proteins with distinct conformations in the PDBFlex database. From PDBFlex, we looks for experimental structures of protein sequences in the CATH topology split test set (95% sequence identity or above), and take all paired instances that are at least 5 angstroms apart in overall RMSD between conformations. We report perplexity on locally flexible residues (defined as local RMSD above 1 angstrom). To be more

			Perplexity
GVP-Transformer (142M params, mixing ratio 1:40)	Span masking	Gaussian noise	4.10
	Span masking	No noise	4.32
	Independent random masking	Gaussian noise	4.30
	No masking	Gaussian noise	4.20

Table A.3: Effects of adding Gaussian noise to predicted structures and effects of span masking during training, as measured by perplexity on CATH topology split test set.

conservative in our evaluation, we show the better of the two conformations to represent single-state perplexity in Figure 3.7.

Ablation on the number of GVP-GNN encoder layers in GVP-Transformer. Increasing the number of GVP-GNN encoder layers improves the overall model performance (Figure A.6), indicating that the geometric reasoning capability in GVP-GNN is complementary to the Transformer layers.

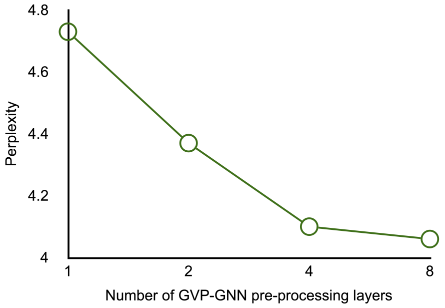

Figure A.6: Effects of varying the number of GVP-GNN pre-processing layers in the GVP-Transformer model, as measured by perplexity on CATH topology split test set.

Model performance when trained only on predicted structures. When trained on the 12 million predicted structures without including any of the experimental structures from CATH in training data, the model performance of GVP-GNN, GVP-GNN-large, and GVP-Transformer is across the board substantially worse than when trained only on the CATH structures (Table A.4). This gap is especially pronounced for the larger GVP-GNN-large and GVP-Transformer models.

	Perplexity
GVP-GNN	6.52
GVP-GNN-large	11.51
GVP-Transformer	10.95

Table A.4: Model performance when trained only using the 12 million predicted structures without CATH training data, as measured by perplexity on CATH topology split test set.

	Pearson correlation			
Fold	Structured GNN [22]	GVP-GNN [13]	GVP-GNN-large+AF2	GVP-Transformer+AF2
$\beta\beta\alpha\beta\beta_{37}$	0.47	0.53	0.62	**0.70**
$\beta\beta\alpha\beta\beta_{1498}$	**0.45**	0.39	0.37	0.33
$\beta\beta\alpha\beta\beta_{1702}$	0.12	**0.26**	0.24	0.22
$\beta\beta\alpha\beta\beta_{1716}$	0.47	0.57	**0.60**	0.58
$\alpha\beta\beta\alpha_{779}$	0.57	0.48	0.62	**0.64**
$\alpha\beta\beta\alpha_{223}$	0.36	0.47	**0.57**	0.55
$\alpha\beta\beta\alpha_{726}$	0.21	0.19	0.24	**0.26**
$\alpha\beta\beta\alpha_{872}$	0.23	0.39	0.38	**0.42**
$\alpha\alpha\alpha_{134}$	0.36	0.44	0.46	**0.50**
$\alpha\alpha\alpha_{138}$	0.41	0.44	0.55	**0.58**
Average	0.37	0.42	0.47	**0.48**

Table A.5: Mutation stability prediction performance for small *de novo* proteins [11], with highest correlation bolded.

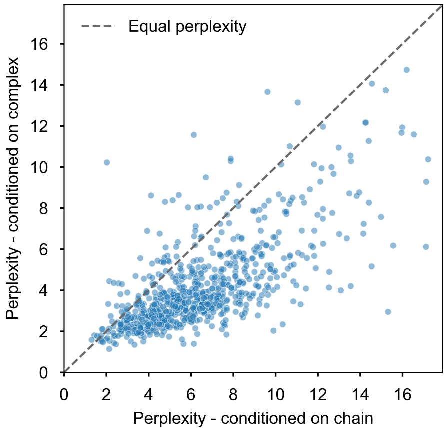

Figure A.7: Fixed backbone sequence design perplexity for protein complexes. The model is evaluated on 796 structurally held-out protein complexes. Comparison of conditioning on the backbone coordinates of individual chains (x-axis) with conditioning on backbone coordinates of the entire complex (y-axis). Note that for both values perplexity is evaluated on the same chain in the complex. The shift to the lower right indicates improved perplexity when the model is given the complete structure of the complex.

Stability prediction on de novo small proteins. We predict protein stability on an experimentally measured stability dataset for *de novo* small proteins [11]. We use the relative difference in sequence conditional log-likelihoods as a predictor for stability and compute Pearson correlation with the mutation effect following [22], assuming that more stable sequences should score higher in log-likelihoods. For each fold, Rocklin et al. [11] starts with a reference protein and generates sequence variants with single amino acid substitutions. We calculate the Pearson correlation between sequence conditional log-likelihood scores and experimental stability measurements for all designed sequences in each fold. With predicted structures as additional training data, the GVP-Transformer model improves the pearson correlation on 8 out of the 10 folds.

Perplexity and sequence recovery of SARS-CoV-2 RBD. We show perplexity and sequence recovery on the SARS-CoV-2 protein receptor binding domain (RBD) as an example for inverse folding. The RBD can exist in a closed-state with the RBD down or in an open-state with the RBD up [7], as illustrated in Figure A.8. The SARS-Cov-2 spike protein

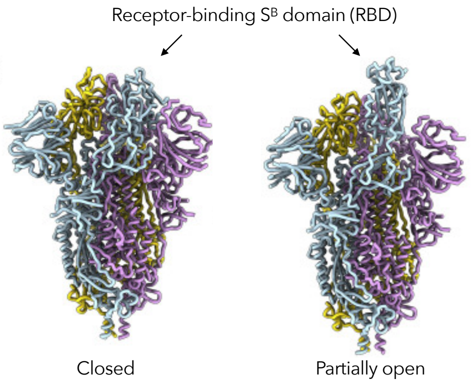

Figure A.8: Illustration of the closed and open states of the SARS-CoV-2 spike protein receptor-binding domain. Cryo-EM structures from [7] (open state: PDB 6XRA; closed state: PDB 6VXX).

	Perplexity			Recovery %		
	Open state	Closed state	Dual-state	Open-state	Closed state	Dual-state
GVP-GNN	4.64	5.13	4.20	45.3%	44.2%	49.7%
GVP-Transformer	4.50	4.96	4.06	49.2%	48.1%	53.6%

Table A.6: Perplexity and sequence recovery on the SARS-Cov-2 spike protein receptor binding domain (RBD), conditioned on either the closed state, the open state, or both states (illustrated in Figure A.8). The inputs to inverse folding models consist of the backbone coordinates for the entire spike protein, while the perplexity evaluation is only on the RBD.

structure has no match with the training data with TM-score above 0.5. The SARS-Cov-2 spike protein has both an open and closed state (open state: PDB 6XRA; closed state: PDB 6VXX). We evaluate perplexity and sequence recovery conditioning on each of the two states independently and jointly. Conditioning on the open state results in better perplexity and sequence recovery than conditioning on the closed state. Conditioning on both states gives improvement in both perplexity and sequence recovery compared to conditioning only on the open state. See Table A.10 for a list of randomly sampled dual-state sequence designs from GVP-Transformer as examples.

Predicting RBD-ACE2 binding affinity. We used the binding affinity dataset provided by Starr et al. [9] (`https://github.com/jbloomlab/SARS-CoV-2-RBD_DMS`), restricting to sites within the RBM subsequence. We used the RBD-ACE2 structure determined by Lan

		AUROC
Supervised	3DCNN	0.57
	GNN	0.62
	ENN	0.57
	GVP-GNN	0.68
Transfer	GVP-GNN	**0.71**
Zero-shot	GVP-GNN (chain)	0.58
	GVP-GNN (complex)	**0.71**
	GVP-GNN-large+AF2 (chain)	0.61
	GVP-GNN-large+AF2 (complex)	**0.71**
	GVP-Transformer+AF2 (chain)	0.60
	GVP-Transformer+AF2 (complex)	0.68

Table A.7: Protein complex stability on SKEMPI test set (binary classification of increase in stability on single-point mutations). Although only trained on single chains, the inverse-folding models generalize to protein complexes. Giving the full complex as input, *complex*, improves performance compared to giving only the chain as input, *chain*. Zero-shot prediction compared to fully supervised and supervised transfer learning methods from [12] and [13] trained on the SKEMPI train set.

	Spearman correlation (zero-shot)			
Evaluation subset	ESM-1v	GVP-GNN	GVP-GNN-large+AF2	GVP-Transformer+AF2
Mutated	−0.23 ± 0.03	**0.34 ± 0.02**	0.29 ± 0.03	**0.31 ± 0.03**
Designed	0.42 ± 0.02	0.65 ± 0.01	**0.72 ± 0.01**	0.67 ± 0.02
High-fitness	0.22 ± 0.02	0.13 ± 0.03	0.21 ± 0.03	**0.26 ± 0.02**
Sampled	−0.21 ± 0.03	**0.35 ± 0.02**	0.30 ± 0.02	0.30 ± 0.03
$\geq$ 2 mutations	−0.20 ± 0.03	**0.35 ± 0.03**	0.29 ± 0.04	0.30 ± 0.02
$\geq$ 3 mutations	0.28 ± 0.03	0.53 ± 0.02	**0.62 ± 0.02**	**0.64 ± 0.02**
$\geq$ 8 mutations	0.20 ± 0.03	0.47 ± 0.02	**0.53 ± 0.02**	**0.55 ± 0.02**

Table A.8: Zero-shot performance on AAV split [14].

et al. [10] (PDB: 6M0J). For mutational effect predictions with ESM-1v, ESM-1b, and ESM-MSA-1b, we scored mutations using the masked-marginal likelihood ratio between the mutant and wildtype amino acids. To generate the MSA used as input to ESM-MSA-1b, we searched `uniclust30_2017_07` [195] with `hhblits` [193] (using two iterations and an E-value cutoff of 0.001) based on the RBD wildtype sequence as the query.

Predicting complex stability changes upon mutations. SKEMPI [94] is a database of binding free energy changes upon single point mutations within protein complex interfaces. This database is used as a task in the Atom3D benchmark suite [12] for comparing supervised stability prediction methods. The task is to classify whether the stability of the complex increases as a result of the mutation. We compare zero-shot predictions using inverse folding models to supervised and transfer learning methods [12, 13] on the Atom3D test set. We find that sequence log-likelihoods from GVP-GNN, GVP-GNN-large, and GVP-Transformer models are all effective zero-shot predictors of stability changes of protein complexes (Table A.7), performing comparably to the best supervised method which uses transfer learning.

Predicting insertion effects on AAV. Using masked coordinate tokens at insertion regions, inverse folding models can also predict the effects of sequence insertions. Adeno-associated virus (AAV) capsids are a promising gene delivery vehicle, approved by the US Food and Drug Administration for use as gene delivery vectors in humans. Focusing on mutating a 28-amino acid segment, Bryant et al. [80] generated more than 200,000 variants of AAV sequences with 12–29 mutations across this region, and measured their ability to package of a DNA payload. This dataset is unique compared to many other mutagenesis datasets in that most sequences feature random insertions in the 28-amino acid segment, as opposed to only random substitutions.

We use inverse folding models to predict insertion and substitution effects as follows: For each sequence, we input the full backbone coordinates of the wild-type (PDB: 1LP3), and insert one masked token into the input backbone coordinates for each insertion. Then we compare the conditional sequence log-likelihood on this input with masks to the conditional sequence log-likelihood of the wild-type sequence on the wild-type backbone. The difference in these two conditional log-likelihoods are used as the score for predicting packaging ability.

We report the zero-shot performance on each of the 7 data subsets evaluated in the FLIP [14] benchmark suite. For amino acid insertions (marked as lowercase letters in the FLIP data), the corresponding backbone coordinates for those amino acids are marked as unknown in the input structure. As shown in Table A.8, GVP-Transformer trained with predicted structures outperforms the sequence-only zero-shot prediction baseline ESM-1v on 6 out of the 7 data subsets. The reported standard deviations are calculated by sampling different subsets of 10,000 variants from the evaluation data.

For ESM-1v, we scored variant sequences based on the independent marginals formula as described in Equation 1 from Meier et al. [95], scoring mutations using the log odds ratio at the mutated position, assuming an additive model when a set of multiple mutations T exist in the same sequence:

$$\sum_{t \in T} \log p(x_t = x_t^{mt} | x_{\backslash T}^{ins}) - \log p(x_t = x_t^{wt} | x_{\backslash T}) \tag{A.1}$$

where $x_{\backslash T}^{ins}$ in the first term is the wild-type sequence with mask tokens at insertion positions and $x_{\backslash T}$ in the second term is the wild-type sequence without insertions.

Confusion matrix. We calculated the substitution scores between native sequences and sampled sequences (sampled with temperature $T = 1$) by using the same log odds ratio formula as in the BLOSUM62 substition matrix. For two amino acids x and y, the substitution score $s(x, y)$ is

$$s(x, y) = \log\left(\frac{p(x, y)}{q(x)q(y)}\right), \tag{A.2}$$

where $p(x, y)$ is the jointly likelihood that native amino acid x is substituted by sampled amino acid y, $q(x)$ is the marginal likelihood in the native distribution, and $q(y)$ is the marginal likelihood in the sampled distribution.

Calibration. Calibration curves examines how well the probabilistic predictions of a classifier are calibrated, plotting the true frequency of the label against its predicted probability. When computing the calibration curve, for each amino acid, we bin the predicted probabilities into 10 bins and then compare with the true probability.

Placement of hydrophobic residues. We define the amino acids IVLFCMA as hydrophobic residues, and inspect the distribution of solvent accessible surface area for both hydrophobic residues and polar (non-hydrophobic) residues. Solvent accessible surface area calculated with the Shrake-Rupley ("rolling probe") algorithm from the biotite package [56] and summed over all atoms in each amino acid. All models have similar distributions of accessible surface area for hydrophobic residues, also similar to the distribution in native sequences (Figure A.11).

Sampling speed. We profile the sampling speed with PyTorch Profiler, averaging over the sampling time for 30 sequences in each sequence length bucket on a Quadro RTX 8000 GPU with 48GB memory. For the generic Transformer decoder, we use the incremental causal decoding implementation in fairseq [15]. For GVP-GNN, we use the implementation from the gvp-pytorch GitHub repository.

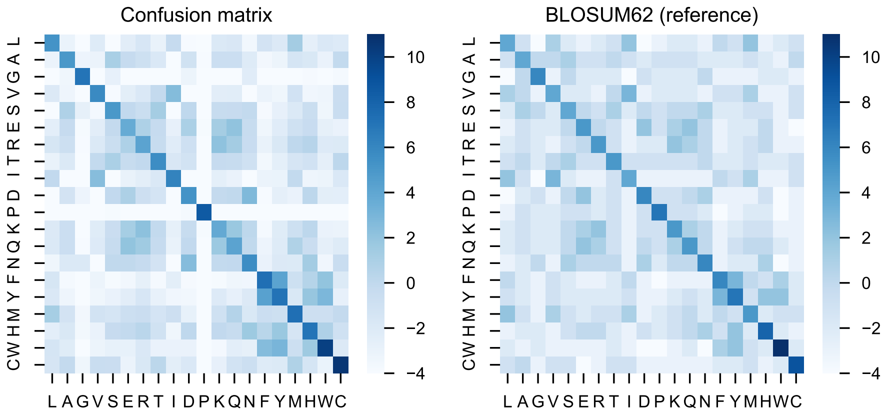

Figure A.9: Confusion matrix between native sequence and sampled sequences from the model, compared to BLOSUM62 as reference.

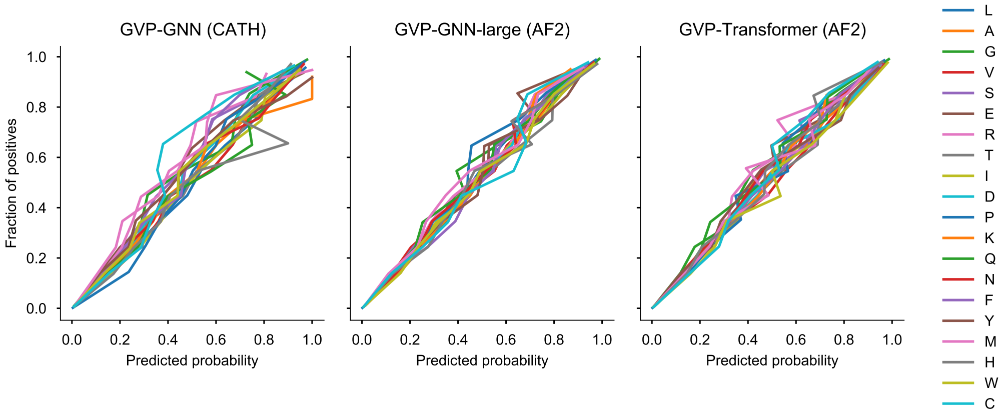

Figure A.10: Calibration.

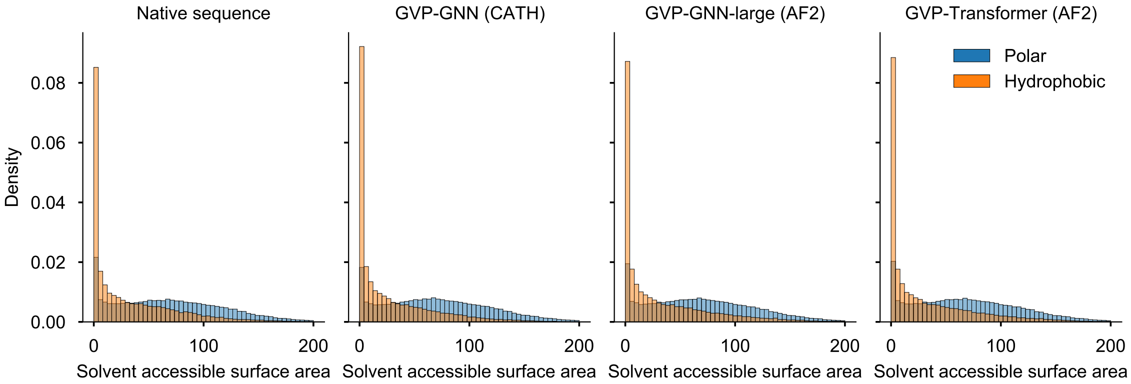

Figure A.11: The majority of hydrophobic residues are buried, following a long tail accessible surface area distribution as in native sequences.

	Average sampling time per sequence				
Sequence length	≤ 100	$100 - 200$	$200 - 300$	$300 - 400$	$400 - 500$
GVP-GNN (3 layers)	3.7s	9.3s	20.8s	76.9s	150.3s
GVP-GNN-large (8 layers)	6.7s	11.8s	47.5s	90.3s	168.8s
GVP-Transformer (8 layers)	1.5s	2.6s	9.0s	16.2s	26.0s

Table A.9: Average time required for sampling one sequence, using open source implementation of GVP-GNN and open source implementation of Transformer from fairseq [15].

```
RBD native sequence:
FPNITNLCPFGEVFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDLCFTNVYADSFVIRGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNNLDSKVGGNYNYLYRLFRKSNLKPFERDIYFPLQSYGFQPTNGVGYQPYRVVVLSFELLHAPATVCGPKK
Designed sequences:
YPDLTNGCPLESVFQRSEYASVVDWSKEEIHDCVLDFTELRSSSDWDTIQCYGVTPQELNELCFTKVQAFSFVIQGFKVKEVAPGKKGVLADKTFRLDDDFTGCVVAKSARDKYATLMPNRNYYYRTNSSVKLLPYQRTIKDPLVSFGFTRTNSLPNQPYRVVVLRFDENYHFVQVCGPRG
FPRISTSCPTKKIFNTAQFAQVHSYSSKKIENCVLNMDEIVSSAEWDTVECYGVSKEKLNTLCFSQVEALSFVIEGRSTAEIKPGATGWIADYDFRLPDDFTGCVYAKSAVNNLAQQEPNHGYWFREIAQQHIPPYEKSTSFPLKYYAFNPSASEDLQPYQVVVFKFDTHLTFASVCGPVS
YPENENVCNTDEIFDKKKYAQVYKYANEKISNCVLDFSEVFESAKWETKTCFGVNKSDLNQICFTKVNAHSFIIRGDKVSQVAPGMEGVLADDTFKLPNDFVGCVVAKTSANKYAAEKANNNYWYLKVDLKAIAPYEQKDESPLSYNSFKPTNGVALQPFKVVVYEFDMSLKKASVCGPLD
FPENSRRCPLDRVFKRKHFHSSDNWGEEQLENCVLDFTDVYTNQRFDSVKCFGVKEEELEEFCAVDVRAHRFVIKGALRDEVAPGMHGPFADYDFRLDSDFTGCVVARSAKNSIAKEPVNNSFWYVHTRAIKIRPYERINQKNDSYYGFRETAAQDHQPFEVVVFEFIFKNAFADVCGPVE
NRDNSNTCPLHAIFKIESYNTVHDWGAFDISDCVLDFSIVVENTRWDTIECYGDAKEDLNSLCFTSVKAHSFVIESKDVKQIAPGMSGMIADFTFKLPDDFVGCVISKTAVEKYAKKPENTSYYYRVKDNSKILPYQRLTPTPLYYWGFKGTNEYEKQPYEVVVFDFDEEIKTCSVCGPQG
YPLIQNSCPLNQVFDVHQFASIDNYSTTKLENCVWNFSVTYNSNSWDEIVCYGEAKNELNDRCYVKVRAHQFVIEGKDVKQIAPGRTGVYADNTFRLPDNYTGCVISKSTRNRIAKEDGNRDYLYTTSVDSEIAPFENKISSPLQSYTFSPTAPYNCEPYEVTVFSFDTTYADAKVCGPKN
NPSIENVCDLNADFELEHFHPVNNFGNSTISNCILDMTKVINNNAWDTVECFGVSKYLLEDMCFTAVFAHRFVIEGKLVKEIAPGKTGSIADVTFRLPNNFVGCVVSRSTVNDDAKVEGNTSYHFRRTASEKIQPFEKRLESPCEHFSFSPTDSEDLQPYQVSVYDFDTNYNFVSVCGPRG
YPDISKACPTDAVFMKDHYASSVNYSQIKIQDCVMDFSWVDESTKWDTKHCYGVHNNWLDEMAFKEVNAHCFVIKGHKYKEIAPGRHGVIADYTFKLPDNFVGCVVSRSSRDYIARDGFNTSYLFRTFDRAVIEPYEIQELQPLKKYNFTPYSPDNSQPYHVVVFEMDTKNDFKNVCGPQK
RPPIFESCPLKDIFEKSSFASVSNWSKEVIRDCVLDMSAVTQSSKWDTFRCYGVEKDELNHQCYTSVYAHSFVVRGKDVVEIAPGKKGEIANATFRLPDSFVGCVVAKSERNNYAEVPPNLNFLFRRERDEKLAPYEVLTSSGLEMYGFTPSEAAGKQPYEVVIFAFDEGKNASNVCGPQA
YPLVSESCNFDAIFNADHYDSVHSWGKKKISNCVLNFDEIITSDKFHSKTCYGVKVHELNHICYTEVQAHSFVIRRADVKEVAPGKNGILADKTFRLPDDFVGCVVSHTAKNKISKDKADFGYYFRRLSRFSIAPYSNKTHSGLEYFNFEPSSGYNEQPYRVVVFELNLQRQACDVCGPIT
YPTINRQCPLDEVFDKEMFSSVDAYSVSVISDCVLDMSSTTDSPRWDCIRCYGDFKYGLNDDCYTSVEAHNFRIRNSEIVEIAPGKHGYLADDTFRLPDNFVGCVVCRSARNKYAKLNWNTDYYYVNLTPEKILPFENKSCHPLKFYNFASTNGTEEQPFDVVVFAFWKQDRARNVCGPIS
YPDNDKTCPTRDVFEVHQFDSVYDFSKKKLKDCVLNMSEITNSREWSTRYCYGVTPNDLNSLCFTEVYAHSFVIEGSKVKEIAPGRTGDLADFTFRLPNDFKGIVVARTSRYTVAREFGNERYKYRNNLSKEINPFENLEINPISHYSFSPNNRSDMQPYEVVVYDFDENYDFCDVCGPQH
YPPITKRCPLDTVFNTSTYATVDNWSKTSIHNCILDTSIVYNSTTFDTIQCYGVSRYQLNSDSFITVFAHTFIIKSELVSEVAPGKTGTIADYTFKLPDTFTGCVVDRSARFSLYEPTANDKYFYRKNASSKIAPYKRNFENPLKFMNFSPNNTRDNRPREVVVFAFNTHFAKVDVYGPVK
NPTITENCPLATVFKTREFSSVVNFGAKDITDCVLDWSKVTSSKRYDTIKCFGVGKSRLETMCFSNVTAHSFIMRGKDYAEVAPGRRGEYADTTFRLPDNFVGCVVARNAKDVLAKLAANRGYKYRLQLKSTINPYERIESEPLQYYNFSPSNSLPDQPYAVVVYEFNQELILSNVCGPVA
HPLNQNICPFDKIYATDRLSSVYNYSRDIIQDCVLNFNLIYLSSAWDTIKCFGVSRFDLNERCYSKVSAHSFIIRGDRVSEIAPGKHGIIADATFRLPDNFTGCVISKSLRNRYAINPPNIFYYYRRQLDAKINPFERVDTNPVKKWNFRPTNLADDQPYAVNVFEMDDNYVFAHVCGPQN
NNAITKDCPLREVFSRKEYASVCNWSVTNISYCVMNMSIIHESSEWDTIDCYGVKKTTLNNLCFTKVKALSFVIKGSEVNEIAPGKQGYIADLCFRLPDTFEGCVVARNAVDTYAQCPRNRMYYYRKNSSKKLAPFEKLNNSPLKYHDFNNTDSFDEQPYKVVVFQLDKKRSAQSICGPQK
YPLNTNHCPLDEVFDTETYAMVHDWSRTILSNCRLDMSQIYSSTSFTTKRCFGINTDDLNDMCFTKVEAHNFVIRGDEVKQIAPGMTGRLADHTFKLDDTFTGCVVARSTRNTIAKTGANDAYLYVQQAQKKIAPYQQKKSDPQCTFEFSPVCSRDLAPFEVVVFDFDTTHDFKDVDGPRS
RPRLLNKCPLSTIFEKDTFEEISDWDETAIKDCIFDFSTIVNNTKWNSKSCFGVKANELNDLCFTYVSALSFVIQGKDVSQVAPGKSGKIADLTFKLPDDFVGCVVAKSYVAKEAWFGSNRTFYYRQDSSANVKPFEKNNNTPCHAYPFQATQAEHEQAFGVVVFRFDREWSEGTVHGPQT
YPRITTTCPTNYVFRRNEYASVDRFSAENITNCVLDMSTITNSADFNTIRCYGVKANDLNNLCFVKVQAHSFVIKGHDVNEVAPGKSGVIADYTFKLSNNFSGCVIAKSAKDRYAKPPANNNYYYLNRSTGHIQPFEVLHGSPLLHYGFSPSKSYDNQPFEVIVLDFNTELRSSSVCGPQN
RPNNTGTCSTDKIFDQRYYASVDAFSSATIKNCVLDFRKVTNASVNDIQKCYGVQQDGLDSLCFVEIKAHNFVIRGDLTVEVAPGRKGYVADATFRLPDSFTGCVVSRSARNHLAARGENLDFRYRPHRDQKLKPYERSNANPNASYGFNPGATYPNQPYEVVVYSFDLREREADVCGPIS
YPAITNTCPTSDIFRRSRYSSVYKWSQGKFENCILNLDNVHTSTDWDEVKCYGTLASKLNDLCFTKVFGHSFIIRGSEVSQIAPGEHGSLADYTLRLDDDFTGCVFAKDYTSTYAKEPINNHYWYRAVANAPINPFERREDRPLQFYNFSPSSHKNDEPYEVVVFSLDIRHDFCRVYGPEG
YPLVTTKCPFDQILKRDTYASVDNFASQLLSECVLDFTEIYNSRRWDQIRCYGVASKHLNELCFVKVRAHSFVIRGDMVSQIAPGQNGVIADDTFRLPDDFTGCVVSRDAKNEYATDGRNNGYMYRYIAAKDIAPWQRHDDKPDHSYGFRPSNPEERAPYRVVVFEFDLAKKKSSVCGDIT
RPNILNRCPTSKIFDRQTFASVHKWDKVDITSCVLDTSGIVNSGKYAMYKCFGIRPEDLNSLCFTDVKAYSFVIQGKDVKEIAPGMTGELADLTFRLSDDFAGCVIARSAVNKYSKDDANKSFFWRKVRMQKVKPFQRNLFSNLEYRSFKKSNEKDNQPYEVVVFDFNLEKEACSVCGPIL
YPTNSNSCPFESVFLQENYASIYNWESTLIENCILDMTVINEGMQWDTKSCFGVRKSDLNNMCFTKVVAHSLVIEGSKASQIKPGEHGILADRTFTLDSNFVGCVVSKSAVHKFAKEGYNDSYYYVECVVGRQKPFQNKEQTPLRYYNFQPTHLQSQQPYQVVIFEMDESDHRCSVCGPQS
YPNVTTKCPTEDIFDRQTFAQEVLWSKDLIDNCVLDFSYITDSEKWNTFQCYGDNKHDLNDRCYTNVRALNFVVEGKKTLEIAPGRHGLLADVTFRLPDNFIGCVVARSESNRIAKDGFNHNFLYRFNAVRDILPFERTSDSPLAYYGFRPTNLTNLQPYKVVVFVFDENYRERSVCGPIQ
RPDIVNTCPLANVLRRAGYSSVRNWAITRISNCVFDFAIIQNSLLADTLYCFGVNAQDLNSRCFTEVNAHSFVVQGKDIVEIAPGKRGSLADNTFKLDSDFVGCVVSRSALNYEAKTEGNFEYFYRTSADRDISPYERRIRTPLRYYEFKPDMADHEQPFKVVVFAFDEREGFSTVCGPQD
YADIKTRCDFRKIFDRKEYASSYNYSMKHISDCVFDASIVRQSTKWDTIECYGVHKDDLNDMCFVDIKAHNFVIEGQSVSQIAPGRHGQIADNTFRLPDDFVGCVVAKNLRNKYASKDPNRNYLYTISRDGDIEPYERRIVDPLREFHFNRQDPYGWQPFAINVFDFDLREQRTDVCGPLN
NPETSNSCPFSSVFSRSHYSSVDNFGADEIKDCVLDFDSIVKSNKFDTYKCFGVAESDLDTKCFTKVVAHSFIIRGDMVKQIAPGKTGTIANNTFNLPNDFNGCVVAKSSRYEIAGRRYNANYRERLIHVSRIEPYQNPVASPEIAWDFRPTDPTDKQPYDVVVYEFDTSWNACRVCGPVN
YPSINNVCSVTDVFNTDKYASVVNFSNTLIKDCVLDFRIVNSSKEFSVKSCYGVAKDNLNTLCFTSVTANRFVIRGKDVTEISPGKSGHIADYTFRLEDNFTGCVIAKSDLNKLAKTTPNTKYFYRSKRPSAITPFEKLNENPLKFWNFKANNSQGNQPFEVTVFRFDLTEERRTVCGPQK
FPPVTTRCPTDNVFRKENYAAVNTFSETLIKDCVFDFTNIRKSKEWDTFECYGVDPKDLNYLCFTKVWAYRFVIRGDLVRQIAPGERGVVADSTFTLPDDFVGCVVARSQRDDLARRPDNNAYLYRKGDGLFLQPFERYNSNPLEFYGFSPNNEYANQPYEVVVLAMDERYDFTKVHGPVI
RPKNKEVCPTSLVFQSNEFAPVSKWSKSEISNCVLDMSETVNDQDWATVRSYGITKNKLNEECFTEVHAHSMIIRGDMVKEVAPGQRGEIADASFRLPDDFVGVVVARSAVSHYAKGGANDKYYYTVLNTYKIKPYEKMTFNPLKFYQFESDNEAPFNPYKIVVFDFDLNDDFRNVCGPRD
YPNIKETCPFDNVLEKSAYSSVDNWSSTEVKDCVADFSVVVNSAKYDTVTCYGVNKFDLNAICMYKVIAHSFIIEGHRVKEIAPGKHGILADDTFKLPDDFVGCVISRSARNMLAKDRPNDAYYYTKNNGYTRKPFENNKSSPLRYYAFKPTNEEKDQPYEVVIFAFATDYRQADVCGPVK
YPEIQTQCPLKNILDRESYASVKQFSVSKISNCVLDFRETNDSKRWDMIHCYGVDKNSLNSRSYTEVYAHSFVVRGDLVREIAPGQSGILADKTFKLSDDYVGCVVARSAKDKLATSEPDTTYLYRNHSASSIKPFENALRTPLFAYGFNPRNNFQDQPFEVVVFEFDFNLDRASVSGPQA
YPANTTKCPYSAIFDTAEYRSVHEYSSSKISNCVLNFSDVNESTQWDTKLCYGMKPESLDDECFKMVTAHNFVIRGEFVRQIAPGTTGLLADNSFKLPDDFIGCVIAKSGVEVYANSTLSNNYCYISNSSNPIKPFEVGDTSPLQSWPFTPENSISYQPYLVTVLEFNRDFAFYDVCGPID
FPDINRVCPLDDIFYRTNYNQSTNAAETNISSCVFDFSDIILDRKFNSVQCYGVDKHDLNTRCFTQVNAHSFVIEGEYRTQIAPGKHGIIADDTFKLHNSFTGCVVSRDAGDYIASGRGNERYWYRITQSRDIAPYQNQSPSPLRFYNFTPSGSVPHRPFEVVVFDFDYTYDFTNVCGPLL
RPEITAECPFDSVFERTEYAPVFKWSSETLADCILDFRTVYQSSRYDTRVCYGVQKRDLDSLCAVKVEAHSFIIRGDKVSQIAPGKHGVLADNTFKLPNDFVGCVVARSSKNLYAKPHGNENYFYAEGRGSDIDPFENLCINPLKFYNFTPSNMKDNQPYEVIVFAMDYRMRKCSVCGPVE
YPDINNRCPTENIFYRRSFSEVCHFSNDRIENCVLDMRLVISDKKYDTFTPYGVERDELNFQSFVEVRALSFVIRGDKVHQIAPGTTGEIADYTFRLDNNFTGCVVCRDAQNEIAQIGPNMDYTWRCNDRSTIKPYERLTNQPLRRYMWSKTKGFDNQPYKVVVYEFDESFSETDVCGPRN
YPDVLNECPLDQIFDRDHYSSVNAFDMKDIKNCVLDFSSISNSSKWSIKKCFGVALNNLNRLCFKKIYAHRMIINNDRVREIAPGQHGTIAVFDFRLPDDFRGCVIARSSIHRDAKNNCNNAYLWRSGASERILPYQRNYNSPLKYYNFSNDNNDANQPYKVVIFSYNEDYDFCSVCGPQR
YPRIVQHCPTKEIFDTFDYSSVYNYSMEKIHNCVLNFSSVVQSKRWSTVRCYGVNAEDLDDRCYTEVYAHRFVIMGQHVKQIAPGRSGVLADYTFKLPDDFIGCVVARNGNDSLATENYNNKYWWRQSSESSIDPFERRNSSPLFYYNFKPTNERPNQPYMVVVFEMDTKYDFTNVCGPQE
YPQVTTTCPLSEVFDKKKFASVHDFSMTRIEDCVLDFNRIVESSVWKTFKCYGVQPRDLNHICAVRVDAHMFVIRGDWVKQIAPGEHGSIADYTFRLKDDFEGCVVARSKRFELANESINDSYWYRDNWMRDIEPYERLQSSDLKFYGFKRSNSQNDQPFEVVVFELFPEMSFSNVCGPVS
NPPDENECPFDAIFNTEEFASVSKYSNKTITDCVADFSRITNSQEFDSIHCYGVDKHDLNDLAFTNVYAHRFVIQGKDVNEIAPGKTGNLANATFRLDDSFDGCVVSRSGKNKIATPFGNKQYFFRTDRLAAINPYERIVSTSLLTYCFHPEDDYALQPYDTVILELNESFQFRSVCGDEN
RPPNTTECPIDKIFNRSTYASVVNFSSSQLQNCVFDFDIITKSKKWDKIICYGVEKEDLNSLTFSKVMAHSFVIAHDLVGEVAPGKVGEIADVDFRLPDDFVGCVVSQSGVNFLSQAPGNKSYYYRLYANETIKPYEKVNENPLEFYNFTPTNPEDQQPYDVVIFDMDTTDAFRTVCGPVG
RPNITQKCKFQNIIDTNSYASVHNWSESVLENCVFDFSKVVNDQKWESVKCFGISKYDLNDLCYSKVRALSFVIRGDQADEIAPGKHGTIADDTFRLPDKFIGCVVWRSSKNKISESVEDRRFCYIGKSQQWINPYEVKKVTSVQFFNFSPINPEEQQPYEVVVFDLDETHSFASVCGPEL
LPDNTVKCPLSEVFDTKAYSSAVIWSETKIQSCVMDWSIVTKSRSFTTVTCYGVDSTQLNDLSFVEVRAMSFVIKGSLISQIAPGQTGIIADEDFRLPNDFIGCVVARDANNTLARTTINSDYKYRDLDIQTIEPFEVRIRSPMRCYNFEPNNVADKQPYDVVVFSLDKSMNSAHVNGPVN
LPEKTTTCPFSEIFDRRKYASIHKYSNQVISDCVLDFRIVVESSTYDSTDCYGISKDELNKLCFTKVYAHSFVIEGKNVAQIAPGKTGKLADNTFRLPDNFTGCVVHRSAKSRIASTGGNQTYIYRDVATSSLAPYERRYDTPCKTYDFKPEDEANKEPFDVVVFEMNVVHDFCSVCGPVD
NPPINKVCPTSEIFFREAYASVNLWNNFLIRNCVLDFSEILKSKRYDVFQCYGVSKSSLNFLCAQQINAHNFIIRGERVEEIAPGKRGVLADKTFKLPDNFVGCVVAHDMKNRYSLYPSNNSYFYQLLRRKTIQPFENWDNTPLQEYGFSRGAVQNNQPFQVVVLEFDLKYIINDVCGPQN
RPQINNICPLRDIFEKKTYASVHNYSKMEIKDCVLDFSTVYSSSKFDTITCYGVKADDLNKRCFVNVYAHNFVIRGKDVSQIKPGKWGEIADNDFKLSDDFTGCVISRSAANYIAQNPPNTKYIYIKARSSKLEPYQNFQSSPVKFYRFTPSNADNEQPYEIVVFSFNETLRFSSVCGPQG
VPLISETCPASEIFDIESYSSVNAYSRKIISDCLLDWNEITVSTDWSMIHCYGETPEDLNSLCYTRVTAHSFVIQGQNIAEIAPGKSGTIANKSFKLPDDFQGCVISKDMRKKIARYKPNKTYWYRVERNTSLKPFEVLNSTPLKYFEFNILQRRDNRPYAVVVFSFDVQNNFRTVCGPMD
YPANTHHCPFNEIFETKHYSSVNDFSMHNIHNCVADMSNIVNSTVWDIKNCYGVYCEDLNELSFTRVNAHSFVIRGDKVAEIAPGRHGGIADVTFRLPDSFVGCVITRDSRNFIAQGSSDDGYFYSECRQSVIAPFERSFNNQTRYYCFSPFNGPARQPFDVVVFEFNLKRDFSSVCGPQL
FPPITKTCDFNSVFNKKNFASVHKWAMDDISNCTANWDTITKSQSWNMFKCYGEKKNTLNIRSFTNVYALSFVIKNDYVKQIRPGENGILADYTFRLDTDFVGCVVARTAKDKYAKPYGDNTYYYVSKYSRKIKPFEINDASPLYHFVFRREDGNENQPYRIVVFQLDTQYSKPDVCGPRE
```

Table A.10: 50 randomly sampled RBD dual-state sequence designs from the GVP-Transformer model with sampling temperature 1.0 and conditioned on both the open and closed states.

A.3 Evolutionary-scale prediction of atomic level protein structure with a language model

A.3.1 Data

A.3.1.1 Sequence dataset used to train ESM-2

UniRef50, September 2021 version, is used for the training of ESM models. The training dataset was partitioned by randomly selecting 0.5% (≈ 250,000) sequences to form the validation set. The training set has sequences removed via the procedure described in Meier et al. [95]. MMseqs search (`-min-seq-id 0.5 -alignment-mode 3 -max-seqs 300 -s 7 -c 0.8 -cov-mode 0`) is run using the train set as query database and the validation set as target database. All train sequences which match a validation sequence with 50% sequence identity under this search are removed from the train set. The 50% identity threshold is chosen because the purpose of the validation set is primarily to detect overfitting (as is common in the machine learning community), rather than to test generalization. Generalization performance is tested through performance on downstream tasks (such as structure prediction on the CASP14 and CAMEO test sets).

De-novo designed proteins are filtered out from the pretraining dataset via two filters. First, any sequence in UniRef50 and UniRef90 that was annotated as "artificial sequence" by a taxonomy search on the UniProt website, when `2021_04` was the most recent release (1,027 proteins), was removed. Second, jackhmmer was used to remove all hits around a manually curated set of 81 de-novo proteins. jackhmmer was run with `-num-iter 1 -max` flags, with each of the 81 de-novo proteins as a query and UniRef100 as a search database. All proteins returned by jackhmmer were removed from both UniRef50 and UniRef90 via their UniRef IDs (58,462 proteins). This filtering is performed to enable future work evaluating the generalization of language models to de-novo sequences.

To increase the amount of data and its diversity, a minibatch of UniRef50 sequences is sampled for each training update. Each sequence is then replaced with a sequence sampled uniformly from the corresponding UniRef90 cluster. This allowed ESM-2 models to train on over 60M protein sequences.

A.3.1.2 Structure training sets for ESMFold

For training ESMFold, we follow the training procedure outlined in Jumper et al. [3]. We find all PDB chains until 2020-05-01 with resolution less than or equal to 9Å and length greater than 20. All proteins where over 20% of the sequence is the same residue is not considered. MMseqs easy-cluster with default parameters is used to cluster resulting sequences at 40% sequence identity. Only individual chains are used during training, even when the chain is part of a protein complex. This results in 25,450 clusters covering a total of 325,498 chains.

At training time, each cluster is sampled evenly, and then a random protein is sampled from each cluster. Rejection sampling is applied to train on longer proteins more frequently, where protein chains are accepted with probability $\frac{1}{512}\max(\min(\mathrm{Nres}, 512), 256)$.

As described in Hsu et al. [6], we generated a set of 13,477,259 structure predictions with AlphaFold2 using MSAs generated via the process in Rao et al. [85]. The dataset is then filtered to select only sequences with mean pLDDT $>$ 70. Because of the way the dataset is constructed, only 1.5% of the dataset is removed with this filter. Additionally, loss is not calculated for residues with pLDDT $<$ 70. We found that this is necessary to obtain increased performance using predicted structures. Predicted structures are sampled 75% of the time, and real structures 25% of the time during training. Data processing is done with Biotite [56].

A.3.1.3 Structure validation and test sets

During method development (e.g. hyperparameter selection), we used a temporally held out validation set obtained from the Continuous Automated Model Evaluation (CAMEO) server [154] by filtering from August 2021 to January 2022.

We report results by testing 3D structure prediction models on two test sets, both chosen to be temporally held out from our supervised training set. The first test set is from CAMEO, consisting of all 194 test proteins from April 01, 2022 through June 25, 2022. Our second test set consists of 51 targets from the CASP14 competition [155]. For both test sets, metrics are computed on all modeled residues in the PDB file. The full CASP14 target list is:

T1024, T1025, T1026, T1027, T1028, T1029, T1030, T1031, T1032, T1033, T1034, T1035, T1036s1, T1037, T1038, T1039, T1040, T1041, T1042, T1043, T1044, T1045s1, T1045s2, T1046s1, T1046s2, T1047s1, T1047s2, T1049, T1050, T1053, T1054, T1055, T1056, T1057, T1058, T1064, T1065s1, T1065s2, T1067, T1070, T1073, T1074, T1076, T1078, T1079, T1080, T1082, T1089, T1090, T1091, T1099.

These is the full extent of the publicly available CASP14 targets as of July 2022.

No filtering is performed on these test sets, as ESMFold is able to make predictions on all sequences, including the length-2166 target T1044.

A.3.1.4 CAMEO Dataset Difficulty Categories

The CAMEO evaluation places each target into three categories: easy, medium, and hard. This placement is done based on the average performance of all public structure prediction servers. Targets are classified as "easy" if the average LDDT is > 0.75, "hard" if the average LDDT is < 0.5, and "medium" otherwise. In the main text, we report average performance across all targets in CAMEO. In Table A.14 we provide statistics for each difficulty category.

A.3.2 Language Models

A.3.2.1 Computing unsupervised contact prediction from language models

We use the methodology of Rao et al. [137] to measure unsupervised learning of tertiary structure in the form of contact maps. A logistic regression is used to identify contacts. The probability of a contact is defined as

$$p(c_{ij}) = \left(1 + \exp(-\beta_0 - \sum_{l=1}^{N}\sum_{k=1}^{K} \beta_{kl} a_{ij}^{kl})\right)^{-1} \tag{A.3}$$

where c_{ij} is a boolean random variable which is true if amino acids i, j are in contact. Suppose our transformer has N layers and K attention heads per layer. Then A_{kl} is the symmetrized and APC-corrected [196] attention map for the k-th attention head in the l-th layer of the transformer, and a_{ij}^{kl} is the value of that attention map at position i, j.

The metric we use, long range P@L, for each protein, is defined as the precision of the top L predicted long range contacts by confidence for a protein of length L. long range is defined as contacts that are ≥ 24 residues apart in the protein sequence. This is averaged over each protein that we test over. We also use P@L/5 in some sections of this work, which computes precision over the top $L/5$ predictions instead.

The parameters are fit in scikit-learn [197] using L1-regularized logistic regression with λ $= 0.15$. The regression is fit using the same 20 protein training set used in Rao et al. [137], which was simply a random selection from the trRosetta [25] training set. We performed a variability analysis using 20 bootstrapped samples of 20 training proteins from the total set of 14862 proteins. The average long range P@L was 0.4287 with a standard deviation of 0.0028. We also performed experiments using larger training sets, but observed no significant performance change. Given these results, we are confident that selecting a subset of 20 proteins for training provides a good estimate of contact precision performance.

Unsupervised contact prediction results are reported for the 14842 protein test set used in Rao et al. [137], which is also derived from the trRosetta training set, excluding the 20 proteins used in fitting the regression. For both training and test a contact is defined as two amino acids with C-α distance < 8Å.

A.3.2.2 Language model perplexity calculations

Perplexity is a measure of a language model's fidelity and is defined as the exponential of the negative log-likelihood of the sequence. Unfortunately, there is no efficient method of computing the log-likelihood of a sequence under a masked language model. Instead, there are two methods we can use for estimating perplexity.

First, let the mask M be a random variable denoting a set of tokens from input sequence x. Each token has a 15% probability of inclusion. If included the tokens have an 80% probability of being replaced with a mask token, a 10% probability of being replaced with a random token, and a 10% probability of being replaced with an unmasked token. Let $\hat{x}_{i \in M}$ denote the set of modified input tokens. The perplexity is then defined as

$$\textsc{Perplexity}(x) = \exp\Big\{ -\log p(x_{i \in M} \mid x_{j \notin M} \cup \hat{x}_{i \in M}) \Big\} \tag{A.4}$$

As the set M is a random variable, this expression is non-deterministic. This makes it a poor estimate of the perplexity of a single sequence. However, it requires only a single forward pass of the model to compute, so it is possible to efficiently obtain an estimate of the expectation of this expression over a large dataset. When reporting the perplexity over a large dataset (such as our UniRef validation set), this estimate is used.

The second perplexity calculation is the pseudo-perplexity, which is the exponential of the negative pseudo-log-likelihood of a sequence. This estimate provides a deterministic value for each sequence, but requires L forward passes to compute, where L is the length of the input sequence. It is defined as

$$\textsc{PseudoPerplexity}(x) = \exp\left\{-\frac{1}{L}\sum_{i=1}^{L}\log p(x_i \mid x_{j\neq i})\right\} \tag{A.5}$$

When reporting the perplexity for an individual sequence (e.g. on CASP14 or CAMEO), this estimate is used. For brevity, we refer to both of these estimates as the "perplexity," as they can be interpreted in a similar manner.

A.3.2.3 ESM-2 model architecture

We use a BERT [141] style encoder only transformer architecture [89] with modifications. We change the number of layers, number of attention heads, hidden size and feed forward hidden size as we scale the ESM model (Table A.13).

The original transformer paper uses absolute sinusoidal positional encoding to inform the model about token positions. These positional encodings are added to the input embeddings at the bottom of the encoder stack. In ESM-1b [16], we replaced this static sinusoidal encoding with a learned one. Both static and learned absolute encodings provide the model a very cheap way of adding positional information. However, absolute positional encoding methods don't extrapolate well beyond the context window they are trained on. In ESM-2, we used Rotary Position Embedding (RoPE) [198] to allow the model extrapolate beyond the context window it is trained on. RoPE slightly increases the computational cost of the model, since it multiplies every query and key vector inside the self attention with a sinusoidal embedding. In our experiments, we observed that this improves model quality for small models. However, we observed that the performance improvements start to disappear as the model size and training duration get bigger.

A.3.2.4 Training ESM-2

In ESM-2, we have made multiple small modifications to ESM-1b with the goal of increasing the effective capacity. ESM-1b had dropout both in hidden layers and attention which we removed completely to free up more capacity. In our experiments, we did not observe any significant performance regressions with this change.

We trained most of our models on a network with multiple nodes connected via a network interface. As the models get bigger, the amount of communication becomes the fundamental bottleneck for the training speed. Since BERT style models have been shown to be amenable to very large batch sizes [199], we increased our effective batch size to 2M tokens.

For model training optimization, we used Adam with $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-8}$ and L_2 weight decay of 0.01 for all models except the 15 billion parameter model, where we used a weight decay of 0.1. The learning rate is warmed up over the first 2,000 steps to a peak value of 4e-4 (1.6e-4 for the 15B parameter model), and then linearly decayed to one tenth of its peak value over the 90% of training duration. We trained all models for 500K updates except the 15B model which we trained for 270K steps. All models used 2 million tokens as batch size except the 15B model where we used 3.2 million tokens batch size. In order to efficiently process large proteins, we cropped long proteins to random 1024 tokens. We used BOS and EOS tokens to signal the beginning and end of a real protein, to allow the model to separate a full sized protein from a cropped one.

We used standard distributed data parallelism for models up to 650M parameters and used sharded data parallelism (FSDP) [200] for the 2.8B and 15B parameter models. FSDP shards model weights and optimization parameters across multiple GPUs, allowing us to train models that can't fit into a single GPU memory.

We trained each model over 512 NVIDIA V100 GPUs. ESM2 700M took 8 days to train. The 3B parameter LM took 30 days. The 15B model took 60 days. All language models were trained for 500K updates, except the 15B language model which we stopped after 270K updates due to computational constraints.

A.3.2.5 ESM-2 ablation experiments

We ran ablation experiments using 150M parameter models trained for 100K steps. Ablations were performed for RoPE, the training dataset (comparing to the ESM-1b training dataset), and UniRef90 sampling (Table A.15).

Unsupervised contact prediction results show that both RoPE and newer data significantly improve the results. We do observe a slight regression when sampling from UniRef90 clusters, however we believe this difference is small and the UniRef90 cluster sampling is likely to help for the larger models.

A.3.2.6 Relationship between change in Perplexity and Contact Accuracy

The relationship between improvements in perplexity and improvements in contact accuracy can be measured via normalized discounted cumulative gain (NDCG). In particular, we hypothesize that large improvements in perplexity correspond with large improvements in contact accuracy. We define the change in perplexity as the difference in language model perplexity for a particular protein sequence between adjacent model sizes. Similarly, we define the change in contact accuracy as the difference in unsupervised contact precision for a particular protein sequence between adjacent model sizes. By ranking proteins according to the change in perplexity, we then compute the NDCG with respect to the change in contact accuracy. The average NDCG across the five model classes is 0.87.

Algorithm 1 Folding block.

```
procedure FoldingBlock($s \in \mathbb{R}^{C_s \times L}, z \in \mathbb{R}^{C_z \times L \times L}, C_s = 1024, C_z = 128$)
    $b \leftarrow \text{Linear}(z)$
    $s \leftarrow s + \text{MultiHeadSelfAttention}(s, \text{bias} = b)$
    $s \leftarrow s + \text{MLP}(s)$
    $z \leftarrow z + \text{Linear}(\text{Concat}([\text{OuterProduct}(s), \text{OuterDifference}(s)]))$
    $z \leftarrow z + \text{TriangularMultiplicativeUpdateOutgoing}(z)$
    $z \leftarrow z + \text{TriangularMultiplicativeUpdateIncoming}(z)$
    $z \leftarrow z + \text{TriangularSelfAttentionOutgoing}(z)$
    $z \leftarrow z + \text{TriangularSelfAttentionIncoming}(z)$
    $z \leftarrow z + \text{MLP}(z)$
    return $s, z$
end procedure
```

A.3.3 ESMFold

A.3.3.1 ESMFold model architecture

The ESMFold model uses a simple architecture that leverages the evolutionary information captured by the language model. The architecture is split into two parts, similarly to AlphaFold2: a folding module which takes the language model features as input and produces representations, and a structure module which takes the output from the folding module and outputs 3d atomic coordinates. For the structure module, we use the equivariant transformer architecture with invariant point attention proposed in AlphaFold2. For the folding block we simplify the Evoformer block used in AlphaFold2. No templates are used in ESMFold.

The major change that needs to be made to adapt the Evoformer block to language model features is to remove its dependence on MSAs. Since MSAs are two dimensional, the Evoformer employs axial attention [201] over the columns and rows of the MSA. The language model features are one dimensional, so we can replace the axial attention with a standard attention over this feature space. The self-attention uses a bias derived from the pairwise representations. The sequence representation communicates with pairwise representation via both an outer product and outer difference. Other operations in the Evoformer block are

Algorithm 2 ESMFold with N folding blocks. ESM_hiddens returns all hidden representations from an ESM language model. layer_weights contains a trainable weight for each layer of ESM.

```
procedure ESMFold(sequence, $C_{\text{esm}} = 1280, C_s = 1024, C_z = 128, N = 48, L = \text{Length}$)
    ESMFold(sequence)
    $s \leftarrow$ ESM_hiddens(sequence)
    $s \leftarrow$(softmax(layer_weights) * $s$).sum(*over layers*)
    $s \leftarrow \text{MLP}(s)$
    $z \leftarrow \text{PairwiseRelativePositionalEncoding}(L)$
    for $i \leftarrow 1, \ldots, N$ do
        $s, z \leftarrow \text{FoldingBlock}_i(s, z)$
    end for
    return StructureModule($s, z$)
end procedure
```

kept the same. We call this simplified architecture the Folding block, described in detail in Algorithm 1, and shown in Figure 4.2a.

Our final architecture, ESMFold, described in Algorithm 2, has 48 folding blocks. It is trained for an initial 125K steps on protein crops of size 256, and then fine-tuned with the structural violation loss for 25K steps, on crop sizes of 384. We use the Frame Aligned Point Error (FAPE) and distogram losses introduced in AlphaFold2, as well as heads for predicting LDDT and the pTM score. We omit the masked language modeling loss. Language model parameters are frozen for training ESMFold. We use the 3B parameter ESM-2 language model, the largest model that permits inference on a single GPU.

We use a learned weighted sum of ESM embeddings to produce the initial hidden state into the model. This is then fed through a multi-layer perceptron (MLP). The initial pairwise state is simply the pairwise relative positional encoding described in Jumper et al. [3]. We found that using the attention maps initially gives a boost in performance, but this disappears during training. For experiments that do not use any folding blocks, we use an MLP applied to the ESM attention maps as input, and add the pairwise relative positional encoding to the attention map scores. Finally, the StructureModule projects these representations into coordinates.

The predicted LDDT head is output from the hidden representation of the StructureModule. The predicted TM head uses the pairwise representation z. Finally, we also predict the distogram, from the same representation.

A.3.3.2 Masked prediction

It is possible to sample alternate predictions from ESMFold by masking inputs to the language model. We test this procedure with the following protocol: Input 1000 different sequences into ESMFold with different masking patterns in the language model. The masking patterns are uniformly sampled, where 0 to 15% of the sequence is masked out. A prediction is made for each masked sequence, and the sequence with highest pLDDT is chosen as the final model prediction. On average, applying this procedure only results in a 0.021 LDDT increase on CAMEO, but on some PDBs can substantially improve the accuracy, e.g. for PDB 6s44, TM-score improves from 0.81 to 0.94 (Figure A.17).

A.3.3.3 Extracting coordinates from ESM-2

The following methodology is used to project out coordinates from the language model representations (Figure 4.1, Table A.11). We train an equivariant structure module directly on top of the frozen ESM representations using a dataset of experimentally determined structures. The training set is the same as used for ESMFold, and we use the same losses and architecture as the AlphaFold2 structure module. We initialize the pairwise representation of the structure module with the output of an MLP that processes the attention maps of the language model. Note that we do not use the predicted structures dataset as data augmentation in these experiments; we train the projection only with experimentally determined structures.

As language models grow in size, we find a large increase in LDDT, from 0.48 on the 8M parameter LM to 0.72 on the 15B parameter LM. This demonstrates that a simple head on top of a powerful language model already gives reasonably accurate structure predictions.

A.3.3.4 Timing analysis

We evaluate the speed of the model by testing sequences of varying length on a single NVIDIA V100 GPU. ESMFold makes a prediction on a protein with 384 residues in 14.2 seconds, 6x faster than a single AlphaFold2 model. On shorter sequences we see a 60x improvement (Figure A.13). Note that this excludes the CPU time for MSA and template search, as well as the 5x from the default ensemble of models used by AlphaFold2. ESMFold can be run reasonably quickly on CPU, and an Apple M1 Macbook Pro makes the same prediction in just over 5 minutes.

ESMFold provides multiple options for reducing GPU memory utilization including chunked attention, mixed precision, and CPU offloading, some of which come at the cost of inference speed. Combined, the optimizations allow predictions on long sequences (such as length-2166 CASP14 target T1044) on an NVIDIA V100 GPU.

A.3.4 Metagenomic predictions

A.3.4.1 Folding 617 million sequences from Mgnify

We obtained MGnify [59] version 2022 at 90% sequence similarity (MGnify90). We built a fault tolerant distributed system with a main node which, via TCP, communicates sequences to many workers and receives results as folded protein structures. We were able to leverage the resources of a heterogeneous GPU cluster consisting of P100s, V100s, and A100s of various configurations. We estimate that on a homogeneous network GPU cluster of V100s, the entire 620 million sequences would take approximately 28,000 GPU days to fold, which we were able to do in 2 weeks time. We obtained structure predictions and corresponding pLDDT values for each of these sequences.

A.3.4.2 Analysis of folded metagenomics structures

On a random sample of 1M high confidence structures, we used Foldseek search (version 3.915ef7d) [5] to perform an all-by-all structural similarity search against the PDB (as of April 12, 2022) based on TM-score. We use foldseek with default parameters, except increasing the E-value to 1.0 from the default 1e-3 (foldseek search -e 1.0), to increase recall. We also used MMseqs2 search (version 13.45111) to perform an all-by-all sequence similarity search against UniRef90. We use MMseqs2 with default parameters, except that we re-ran MMseqs2 with the most sensitive setting (-s 7.0) for any sequences that returned an empty result, to increase the recall.

As mentioned in Section 4.4 for 3.4% (33,521 proteins) of the Atlas 1M high confidence subsample, no significant match is found in UniRef90 with MMseqs2. For reference, a random subsample of MGnify90, without confidence threshold, has 26.4% (264,075 proteins) without hits in UniRef90. For this larger set of sequences without close homologs, the predicted structures, despite being below or stringent confidence threshold, can still be valuable as they may have regions of well-predicted structure which can be enough to enable discovery of novel proteins.

To visualize this landscape of 1M MGnify sequences, we first used ESM-1b to embed each sequence as a 1280-dimensional vector. These embeddings were then visualized using the umap version 0.5.3, scanpy version 1.9.1, and anndata 0.8.0 Python packages [202, 203, 204], where dimensionality reduction was applied directly to the embedding vectors (use_rep='X' in scanpy.tl.umap) with default parameters (15-nearest-neighbors graph via approximate Euclidean distance, UMAP min_dist=0.5).

We further analyzed a random subsample of very high-confidence structures with mean pLDDT greater than 0.9, corresponding to ∼59K structures. For each of these structures, we used Foldseek easy-search (`-alignment-type 1`) to identify similar structures in the PDB. To assess the quality of structure predictions with no Foldseek matches, we used full AlphaFold2 with MSAs to also obtain structure predictions, where we picked the top of five relaxed models ranked by mean pLDDT. We then computed RMSD values of aligned backbone coordinates and all-atom TM-score between the ESMFold-predicted and AlphaFold2-predicted structures and found good agreement of the predictions between both methods (Figure A.18).

To select our case studies, we then used blastp version 2.10.0+ to search for similar sequences in UniRef90 to compute sequence identity. For case-study sequences with no significant matches in UniRef90, we also used the jackhmmer web server (https://www.ebi.ac.uk/Tools/hmmer/search/jackhmmer) [161] to manually query four reference proteomes for similar sequences. Highlighted structure predictions with low similarity to known structures were manually selected and are summarized in Figure 4.4. For these structures, we also performed an additional structural similarity search using the Foldseek webserver (https://search.foldseek.com/search) with default parameters to identify the closest structures in PDB100 211201 beyond the TM-score cutoff of 0.5.

A.3.5 Multimer Benchmark

A.3.5.1 Recent-PDB-Multimers

To evaluate ESMFold on protein complexes. We construct an evaluation set using the methods described in Evans et al. [82]. This dataset consists of targets deposited in the Protein Data Bank between 2020-05-01 and 2022-06-01. The following filtering steps are performed:

- Complexes must contain more than 1 chain and less than 9 chains.
- Chains with length < 20 residues, or where one residue makes up $> 20\%$ of the chain are excluded.
- Complexes must contain fewer than 1536 residues, excluding chains which fail the

previous step.

- Each chain is assigned to a 40% overlap cluster using clusters provided by the PDB
- Each complex is assigned a cluster which is the union of chain cluster ids
- From each cluster complex, the example with highest resolution is selected as the representative

These steps result in a total of 2978 clusters. Predictions are made on the full complex, but metrics are computed on a per chain-pair basis using the DockQ program [4]. Chain pairs are greedily selected for evaluation if their pair cluster id has not been previously evaluated. Chain pairs which DockQ identifies as having no contacting residues in the ground truth are not evaluated. This results in a total of 3505 unique chain pairs.

A.3.5.2 Multimer Predictions

To predict complexes in the benchmark shown in Figures 4.2d and A.15, we give a residue index break of 1000 to ESMFold and link chains with a 25-residue poly-glycine linker, which we remove before displaying. Note that this is using ESMFold out of distribution since single chains are used during training.

A.3.6 Orphan Proteins

Orphan proteins are sequences with few to no evolutionary homologs in either structure or sequence databases. Due to a lack of evolutionary information, these sequences can be very challenging for current structure prediction models. To evaluate ESMFold on orphan proteins, we construct an orphan protein dataset using the following procedure:

- Select structures deposited in the PDB from 2020-05-01 to 2022-05-01 with resolution greater than 9Å and at least 20 modeled residues.

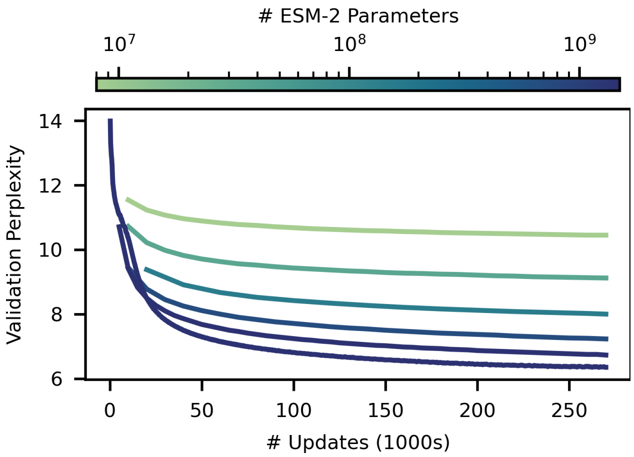

Figure A.12: *ESM-2 masked language modeling training curves.* Training curves for ESM-2 models from 8M (highest curve, light) to 15B parameters (lowest curve, dark). Models are trained to 270K updates. Validation perplexity is measured on a 0.5% random-split holdout of UniRef50. After 270K updates the 8M parameter model has a perplexity of 10.45, and the 15B model reaches a perplexity of 6.37.

- Cluster at a 70% sequence identity threshold with mmseqs, and select the cluster representatives.
- Run hhblits for 1 iteration (all other parameters default) against UniRef (2020_06), select sequences with no hits.
- Run the standard AlphaFold2 MSA generation pipeline against UniRef, MGnify, and BFD, selecting sequences with < 100 total sequence hits and no template hits with TM-score > 0.5.

Figure A.19 shows results at different MSA depth thresholds. After filtering, there are 104 sequences with MSA depth ≤ 100, 70 sequences with MSA depth ≤ 10, and 22 sequences with MSA depth $= 1$. Beyond the constraint that no template has TM-score > 0.5, no filtering on the number of templates is performed.

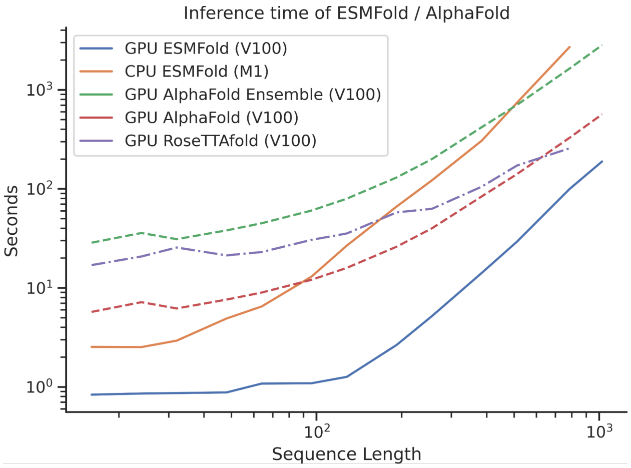

Figure A.13: *ESMFold timing.* Comparison to AlphaFold2 and RoseTTAfold. We test the speed of ESMFold on sequence lengths up to 1024. Note that this comparison is only on the network forward time, and does not include the cost of the search to generate MSAs. ESMFold performance at low sequence lengths is dominated by the forward pass of the language model. At high sequence lengths the $O(N^3)$ computation of pairwise representations takes over. Most of ESMFold's speed advantage comes from not needing to process the MSA branch. We see an over 60x speed advantage for shorter protein sequences, and a reasonable speed advantage for longer protein sequences. We do not count Jax graph compilation times or MSA search times for AlphaFold2, meaning in practice there is a larger performance difference in the cold start case. We also use an optimized Colabfold 1.3.0 [8] to do speed comparison. No significant optimization has been performed on ESMFold, and we suspect that further gains can be made by optimizing ESMFold as well. For RoseTTAfold, the speed of the SE(3) Transformer dominates, especially at low sequence lengths. The number of SE(3) max-iterations are artificially limited to 20 (default 200) and no MSAs are used as input for these measurements. For RoseTTAfold predictions we do not include the cost of computing sidechains with PyRosetta.

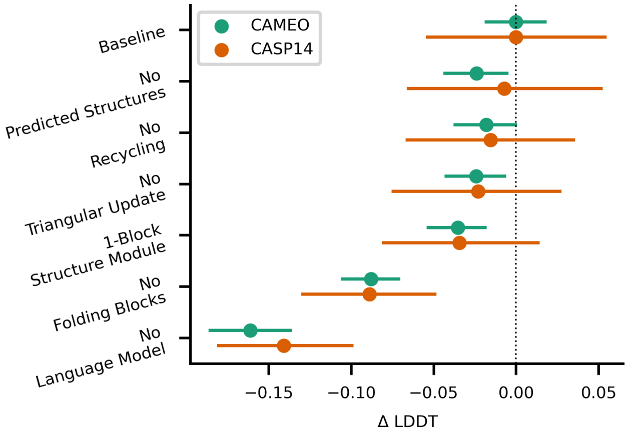

Figure A.14: *ESMFold ablations on CAMEO and CASP14.* ESMFold ablations on CAMEO and CASP14 test sets show the largest contributing factors to performance are the language model and the use of folding blocks. Other ablations reduce performance on CASP14 and CAMEO by 0.01-0.04 LDDT.

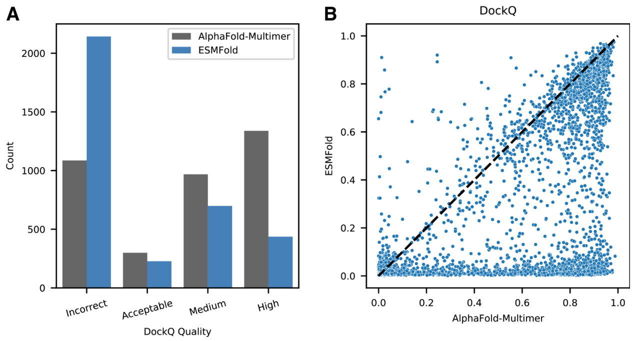

Figure A.15: *Comparison of ESMFold and AlphaFold-Multimer on recent-PDB-multimers dataset.* DockQ [4] scores for AlphaFold-Multimer and ESMFold predictions for chain pairs in the Recent-PDB-Multimers dataset. DockQ qualitative categorizations (left) and quantitative comparison (right) are provided for all chain pairs. ColabFold [8] was used to generate paired MSAs for each complex using the 'paired+unpaired' MSA generation setting. UniRef, environmental, and template databases were used. ESMFold predictions are in the same qualitative DockQ categorization for 53.2% of complexes, even though ESMFold is not trained on protein complexes. Dataset generation and scoring methodology described in Appendix A.3.5.1.

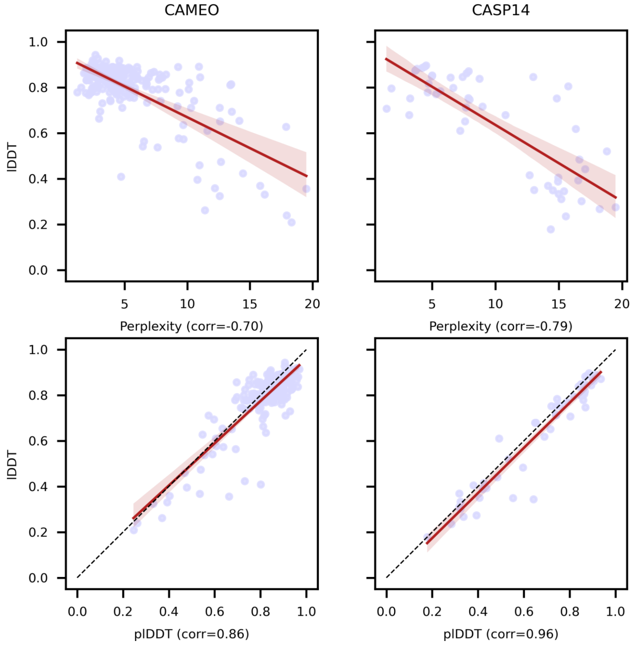

Figure A.16: *ESMFold calibration with respect to perplexity and pLDDT on CASP14 and CAMEO.* Language model perplexity and ESMFold pLDDT are both well correlated with actual structure prediction accuracy on CASP14 and CAMEO. Well understood sequences with language model perplexity < 6 are usually well predicted by ESMFold. The strong correlation between pLDDT and LDDT suggests filtering predictions by pLDDT will mostly capture well predicted structures.

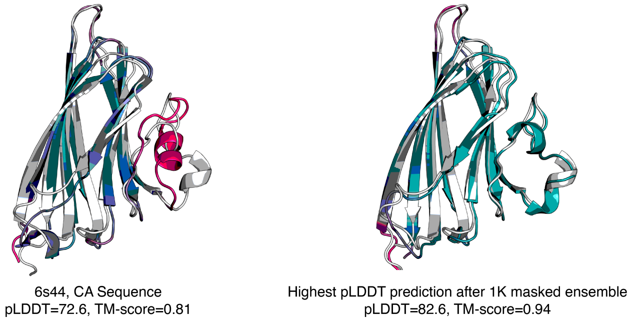
6s44, CA Sequence
pLDDT=72.6, TM-score=0.81
Highest pLDDT prediction after 1K masked ensemble
pLDDT=82.6, TM-score=0.94

Figure A.17: *Masked prediction on Cα sequence of PDB 6s44.* Left: ESMFold prediction (TM-score=0.81) on the Cα sequence of PDB 6s44. Right: Best prediction out of 1000 masked sequences generated via the procedure described in Appendix A.3.3.2. Prediction with highest pLDDT is shown, and has improved TM-score (Tm-score=0.94).

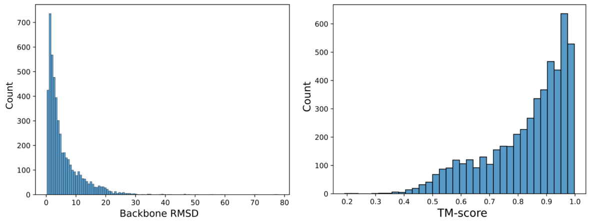

Figure A.18: *Comparison to AlphaFold2 of structurally remote ESMFold predictions.* Distributions of backbone RMSDs (left) and TM-scores (right) of ESMFold-AlphaFold2 predictions of the same sequence, where the ESMFold prediction has both high confidence (mean pLDDT > 0.9) and low structural similarity to the PDB (Foldseek closest PDB TM-score < 0.5).

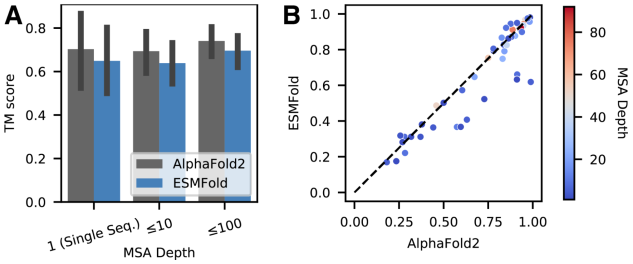

Figure A.19: *Comparison of ESMFold and AlphaFold2 on a set of orphan proteins.* Performance of ESMFold and AlphaFold2 on a set of "orphan proteins" - sequences with few sequence or structural homologs. All compared sequences are temporally held out from the training set. The standard AlphaFold2 sequence and template search pipeline is used to find homologs (dataset construction described in Appendix A.3.6). (A) Comparison on natural proteins with various MSA depths. Depth is the total number of hits across UniRef and metagenomic databases. (B) TM-score comparison of all individual orphans.

Model	# Params	# Updates	Validation Perplexity	LR P@L	LR P@L/5	CASP14	CAMEO
ESM-2	8M	270K	10.45	0.16	0.28	0.37	0.48
	35M	270K	9.12	0.29	0.49	0.41	0.56
	150M	270K	8.00	0.42	0.68	0.47	0.63
	650M	270K	7.23	0.50	0.77	0.51	0.68
	3B	270K	6.73	0.53	0.80	0.51	0.71
	8M	500K	10.33	0.17	0.29	0.37	0.48
	35M	500K	8.95	0.30	0.51	0.41	0.56
	150M	500K	7.75	0.44	0.70	0.49	0.65
	650M	500K	6.95	0.52	0.79	0.51	0.70
	3B	500K	6.49	**0.54**	0.81	0.52	**0.72**
	15B	270K	**6.37**	**0.54**	**0.82**	**0.55**	**0.72**
ESM-1b	650M	—	—	0.41	0.66	0.42	0.64
Prot-T5-XL (UR50) (21)	3B	—	—	0.48	0.72	0.50	0.69
Prot-T5-XL (BFD) (21)	3B	—	—	0.36	0.58	0.46	0.63
CARP (24)	640M	—	—	—	—	0.42	0.59

Table A.11: *Detailed language model comparison.* Comparison at different numbers of parameters and at different numbers of training updates. Training updates and validation perplexity are not reported for baseline models, since there is no straightforward comparison. For the number of training updates, different models use different batch sizes, so the number of sequences seen can vary even if the number of updates are the same. For validation perplexity, baseline models are not trained on the same dataset, and do not share a common heldout validation set with ESM-2. Prot-T5 is an encoder-decoder language model. Only the encoder portion of the model was used in this evaluation, however the number of parameters reported is the total number of parameters used for training. Unsupervised contact precision results, in the form of long range precision at L and at L / 5, do allow us to compare all transformer language models despite variance in training data. However, CARP, a convolution based language model, does not have attention maps. Note: ESM-1b is evaluated only on sequences of length < 1024, due to constraints with position embedding.

MGnify ID	**Mean plDDT**	**ESM-2 (3B) Perplexity**	**Foldseek server closest TM-score**	**Foldseek server closest PDB**	**Closest blastp sequence identity (UniRef90)**	**Closest blastp sequence (UniRef90)**
MGYP000712274586	0.96	3.4	0.45	1ttg_A	54%	UniRef90_A0A539E457 Uncharacterized protein (Acidimicrobiaceae bacterium)
MGYP000911143359	0.90	6.5	0.67	5nni_A	43%	UniRef90_A0A7Y5V7P8 Uncharacterized protein (Flavobacteriales bacterium)
MGYP001220175542	0.94	4.2	0.38	5y1x_A	98%	UniRef90_UPI0013011942 Helix-turn-helix domain-containing protein (Caenibacillus caldisaponilyticus)
MGYP001812528822	0.93	4.4	0.39	5hh3_C	50%	UniRef90_A0A545U581 Fatty acid desaturase (Exilibacterium tricleocarpae)
MGYP000706186022	0.92	11.5	0.47	1xks_A	29%	UniRef90_A0A6N6S1Z1 Uncharacterized protein (Candidatus brocadia)
MGYP000279975524	0.93	5.9	0.49	4l5s_B	38%	UniRef90_A0A1F4EWL6 Uncharacterized protein (Betaproteobacteria bacterium)
MGYP004000959047	0.90	7.8	0.80	6bym_A	No significant matches	NA
MGYP000936678158	0.95	10.6	0.68	5yet_B	No significant matches	NA

Table A.12: *Information on highlighted MGnify proteins.* MGnify sequence identifiers corresponding to predicted structures highlighted throughout this study, including the PDB chain and corresponding TM-score of the closest structure identified by the Foldseek webserver as well as the UniRef90 entry and sequence identity of the closest sequence identified by blastp (Appendix A.3.4.2).

	8M	35M	150M	650M	3B	15B
Dataset	UR50/D	UR50/D	UR50/D	UR50/D	UR50/D	UR50/D
Number of layers	6	12	30	33	36	48
Embedding dim	320	480	640	1280	2560	5120
Attention heads	20	20	20	20	40	40
Training steps	500K	500K	500K	500K	500K	270K
Learning rate	4e-4	4e-4	4e-4	4e-4	4e-4	1.6e-4
Weight decay	0.01	0.01	0.01	0.01	0.01	0.1
Clip norm	0	0	0	0	1.0	1.0
Distributed backend	DDP	DDP	DDP	DDP	FSDP	FSDP

Table A.13: *ESM-2 model parameters at different scales.*

Dataset	Split	Count	MSA Depth (Total)	MSA Depth (UniRef)	ESMFold	AlphaFold2	RoseTTAFold
CAMEO	easy	97	21,458	17,627	0.90	0.93	0.89
	medium	89	3,032	860	0.79	0.86	0.76
	hard	8	328.5	56	0.45	0.62	0.49
CASP14	—	51	1228	161	0.68	0.85	0.81

Table A.14: *CAMEO dataset statistics broken down by difficulty class.* Median MSA depth is reported for each difficulty class of the CAMEO dataset, along with mean TM-score for ESMFold, AlphaFold, and RoseTTAFold. Half of the samples from the CAMEO dataset consist of "easy" examples, which are well predicted by all models. Differentiation is greater in the "medium" and "hard" classes, which have lower MSA depth and are better predicted by AlphaFold2. Statistics for CASP14 are provided as a comparison. MSA depth numbers provided are from the AlphaFold2 MSA generation pipeline.

	LR P@L	LR P@L/5	Validation Perplexity
Baseline	0.381	0.626	8.42
No RoPE	0.365	0.599	8.62
Older UniRef Data	0.368	0.599	7.98
No UR90 Sampling	0.387	0.631	8.40

Table A.15: *ESM-2 architecture ablations.*

A.4 A high-level programming language for generative protein design

A.4.1 High-level programming language and energy-based optimization

A program in our language is fully specified by (1) a syntax tree and (2) a set of constraints. This program compiles to an energy function, which is used to guide black-box optimization of a protein sequence while also leveraging its predicted structure.

A.4.1.1 Syntax tree

The syntax tree consists of nonterminal symbols, which we denote as x_i, as well as terminal symbols, which in our examples we denote as uppercase alphabetic characters such as A, B, C, etc. Each terminal symbol defines a unique protein sequence. The nonterminal symbol x_1 is designated as the special start symbol; all programs must have x_1. Additional nonterminal symbols are used to define hierarchical complexity. For example, specifying two levels of hierarchy requires a nonterminal production rule in addition to a terminal production rule, for example,

$$x_1 \to x_2 x_2 \quad \text{(nonterminal production)} \quad \text{and} \tag{A.6}$$

$$x_2 \to AA \quad \text{(terminal production rule).} \tag{A.7}$$

In the example above, the x_2 nonterminal enables an intermediate level of hierarchy. A nonterminal can produce any finite-length permutation of higher-numbered nonterminals (for example, $x_1 \to x_2 x_3$ is permitted but not $x_2 \to x_1 x_3$ or $x_2 \to x_2 x_3$). A nonterminal can also produce any finite-length permutation of terminals (for example, $x_1 \to AB$) or any finite-length permutation of mixed terminals and higher-numbered nonterminals (for example, $x_1 \to x_2B$ or $x_1 \to Bx_2$).

A complete syntax tree is built by fully expanding the nonterminal x_1 into a set of terminals. The production rules define the connectivity structure of the tree, where the parent node corresponds to the left side of the production rule and the child node(s) corresponds to the right side of the production rule. Across the entire syntax tree, each internal node corresponds to a nonterminal symbol and each leaf corresponds to a terminal symbol. Example syntax trees are provided in the main text figures.

A.4.1.2 Constraints

A program in our language also requires a set of constraints, where a single constraint is defined with respect to a single node and all of its descendants in the syntax tree. Note that this includes constraints on the leaves of the tree (corresponding to the terminal symbols). More specifically, a constraint is a function that takes as input the (sub)tree, as well as its corresponding (sub)sequence and (sub)structure, and outputs a real number. For example, a constraint defined with respect to the node corresponding to x_1 simply receives as input the entire syntax tree, the full-length sequence, and the full protein structure. The same constraint (i.e., the same function) can be applied to multiple nodes in the tree. We will use $f_j(v)$ to denote a constraint j defined with respect to node v.

A.4.1.3 Compilation of constraints into an energy function

We compile a program into an energy function. In our study, we simply compute a linear combination of all the constraints in the user-specified set, i.e., $E(x) = \sum_{v \in \mathcal{V}} \sum_j f_j(v)$, where $f_j(v)$ is defined as zero when a constraint is not applied to a given node and $\mathcal{V}$ denotes the set of nodes in the syntax tree. In practice, we explicitly keep track of a scalar multiplicative weight on each constraint, i.e., $E(x) = \sum_{v \in \mathcal{V}} \sum_j a_j f_j(x_i)$. This energy is used in the simulated annealing optimization procedure described below. Specific examples of constraint functions used in our study are also provided below.

Linear combinations work well for our choice of generative model, but in principle any combination of the energy terms could be used here. For example, if we compiled our program into an energy function for a generative model that used a reward function (like a reinforcement learning agent), we might prefer a multiplicative combination of the inverse of our current energy terms.

A.4.1.4 Simulated annealing

The energy function is used as part of an iterative black-box optimization loop, where over multiple iterations, a change to a given state (in this case, a protein sequence design) is accepted with some probability. We use a simulated annealing algorithm in which the acceptance probability is controlled by a temperature value such that the optimization can tolerate higher energy changes at the beginning of the optimization before favoring changes that decrease the energy toward the end of the optimization. In our study, we begin by initializing the sequence state (one unique sequence per terminal symbol) with uniform amino-acid probability to a given user-specified length; we also compute an initial structure prediction from this sequence.

Each iteration proposes a mutation to the protein sequence. To make this proposal, first, one of the terminal symbols is chosen with uniform probability, and second, one of a substitution, insertion, or deletion is chosen with some probability (we default to 60%, 20%, and 20%, respectively). For substitutions and insertions, the new amino acid is chosen with some probability (unless otherwise specified, we apply uniform probability over a reduced amino acid alphabet that excludes cysteine). We default to uniform probability over all possible sequence positions.

The next step in the iteration is to obtain a structure prediction corresponding to the sequence with the proposed change. This prediction provides the structural information that is used to compute the values of the individual constraint functions. These values are then combined to produce the value of the energy function, as described above. This energy function is evaluated on the overall design with and without the proposed mutation, which we denote $E\left(x^{(i)}\right)$ and $E(x^*)$, respectively. The mutation is accepted with probability

$$\min\left(\exp\left\{\frac{1}{T_i}\left(E(x^*) - E\left(x^{(i)}\right)\right)\right\}, 1\right)$$

where T_i is the temperature parameter at iteration i that decays geometrically over the optimization. Upon acceptance, the value of x^* is set to the value of $x^{(i)}$ and the optimization loop continues. By default, our optimization leverages user-specified values T_{max} and T_{min}, with a decay schedule given as

$$T_i = \left(\frac{T_{\text{min}}}{T_{\text{max}}}\right)^{i/M}$$

where M is the user-specified number of annealing steps. We report specific values for T_{max} and M in the experiment descriptions below and default to $T_{\text{min}} = 0.0001$.

A.4.1.5 Single chain constraint

Our language accommodates both single- and multi-chain design through the use of a special "single chain" constraint. By default, without this constraint applied, all terminal symbols are assumed to correspond to separate chains. When this constraint is applied to a given node, it constrains all of the terminal symbols to be part of a single chain according to the left-to-right order defined in the syntax tree. For example, consider a syntax subtree with $x_2 \to x_3AB$ and $x_3 \to CD$ productions. A single chain constraint applied to node x_2 would create a chain consisting of a contiguous sequence $CDAB$. Unlike other constraints, this constraint is enforced as part of structure prediction, prior to the energy function compilation.

A.4.2 Constraint implementation

A.4.2.1 ESMFold structure prediction

We obtain all-atom structure predictions using ESMFold [171], where the prediction is made over the entire protein sequence and is represented as a set of atomic coordinates and their corresponding residue identities and indices. This predicted structure is the basis for the structural information passed to each of the specific constraint functions. When a constraint is defined on a subtree, that constraint only has access to the structural information (atomic coordinates, etc.) of the sequence encoded by that subtree.

A.4.2.2 Structure prediction confidence (pTM and pLDDT)

ESMFold produces a pTM score, which indicates the model's confidence in the overall structure prediction, and a per-atom pLDDT score, which indicates the model's confidence in the specific atomic coordinate prediction. The pTM value and the mean of the backbone pLDDT values are constraints that are meant to steer the optimization toward structures with higher structure prediction confidence, which is associated with naturally plausible and designable structures. We use a linear combination of the quantities $1 - \text{pTM}$ and $1 - \text{pLDDT}$ (since a higher confidence/lower energy is desirable), with user-specified weights, as the returned value of the confidence constraint.

A.4.2.3 Surface-exposed hydrophobics

The surface exposed hydrophobics constraint aims to reduce the hydrophobicity of the protein surface, where high hydrophobicity leads to protein aggregation and insolubility. We implement this constraint using the Shrake-Rupley "rolling probe" algorithm to determine the surface exposed atoms [205] as implemented in the biotite Python package version 0.35.0 [56]. We then calculate the fraction of atoms involved in hydrophobic residues that are also surface exposed, and we use this fraction as the output of the constraint function.

A.4.2.4 Globularity

It is sometimes desirable to encourage a protein chain to pack into a globular structure. Our globularity constraint is implemented by computing the centroid of a set of atomic coordinates, where the globularity constraint function returns the variance of the distances from all coordinates to this centroid. Intuitively, low variance indicates that all coordinates that are largely equidistant to the centroid, which is more consistent with globular packing.

A.4.2.5 Secondary structure

The secondary structure constraint steers the energy toward user-defined secondary structure. To annotate residue secondary structure, we use the P-SEA algorithm [206] as implemented by the biotite Python package [56]. This constraint function returns one minus the fraction of residues that belong to the desired secondary structure element (since a higher fraction/lower energy is desirable).

A.4.2.6 Rotational symmetry

To design symmetry, we first find it helpful to tie the sequence identities across the subsequences corresponding to the asymmetric units. The first symmetry we consider is rotational symmetry, which only consider the centroids of the immediate children of the constraint's node; for example, rotational symmetry defined on a node x_1 where $x_1 \rightarrow x_2 x_3 x_4$ would only consider the centroids of the individual substructures defined by x_2, x_3, or x_4.

Using the left-to-right order of these children in the production rule, the rotational symmetry function first computes the distances among adjacent centroids, circularly wrapping to include the distance between the first and last symbols; for example, rotational symmetry defined on $x_1 \rightarrow x_2 x_3 x_4$ would compute the set of distances among pairs (x_2, x_3), (x_3, x_4), and (x_4, x_2). The final value returned by this constraint function is the variance among all adjacent distances; intuitively, a rotational or ring-like symmetry would have equal distances among centroids. This rotational symmetry function is adopted from that used by Wicky et al. [160].

A.4.2.7 Globular symmetry

The globular symmetry constraint is defined on the centroids of the immediate children of the constraint's node; for example, globular symmetry defined on a node x_1 where $x_1 \to x_2x_3x_4$ would only consider the centroids of the individual substructures defined by x_2, x_3, or x_4 (this is similar to rotational symmetry described above). The globularity symmetry function computes all pairwise distances among centroids and returns the variance of these distances. In practice, this constraint function is useful for defining symmetry that is not rotational, for example, the symmetry observed in polyhedral assemblies.

A.4.2.8 All-atom coordination

One approach to designing functional proteins is to constrain (a portion of) the protein to match the structure of a known functional site in nature. We accomplish this with an all-atom coordination constraint. This constraint is first defined with respect to a list of atoms from a native protein structure (outside of our designed protein), which we denote y_{native}. We then constrain all of the atoms in the corresponding (sub)tree to match, which we denote y_{design}, as closely as possible, the coordination of the atoms in y_{native}. We achieve this with two functions. The first is the constrained root mean square deviation (cRMSD),

$$\text{cRMSD}(y_{\text{native}}, y_{\text{design}}) = \min_T \left(\frac{1}{n} \sum_{i=1}^{n} \| a_i(y_{\text{native}}) - T(a_i(y_{\text{design}})) \|_2^2 \right)^{1/2} \quad \text{(A.8)}$$

where T is a structural transformation, a_i denotes the atomic coordinates of the ith atom out of n total atoms considered, and $\|\cdot\|$ denotes a vector norm. We implement the structural alignment using the Kabsch algorithm [207] as implemented by biotite [56]. The second function for constraining atomic coordination is the distance-matrix RMSD (dRMSD),

$$\mathrm{dRMSD}(y_{\mathrm{native}}, y_{\mathrm{design}}) = \left(\frac{2}{n(n-1)}\sum_{i=1}^{n}\sum_{j=1}^{n}(d_{ij}(y_{\mathrm{native}}) - d_{ij}(y_{\mathrm{design}}))^2\right)^{1/2} \tag{A.9}$$

where d_{ij} is the Euclidean distance between the ith and jth atoms. The returned final value is a linear combination of the cRMSD and dRMSD values with user-specified weights. In practice, cRMSD is sometimes excluded (i.e., its weight is set to zero) in conjunction with dRMSD, as cRMSD alone does not appear sufficiently stable to create a sufficiently smooth energy landscape.

A.4.2.9 Backbone atom coordination

For a class of design tasks called fixed backbone design, we desire to only constrain the backbone atoms of the protein structure and have the optimization produce sequences that match a known backbone. This constraint is largely equivalent to the all-atom constraint described above, but rather than constraining all atoms (including side chains), this constraint is only applied to the carbon, α-carbon, and nitrogen atoms in the protein backbone.

A.4.2.10 Surface exposure

In some cases, we desire that a given set of residues be exposed on the surface of the protein (for example, when scaffolding a protein binding site). As with the hydrophobics constraint, we leverage the Shrake-Rupley algorithm [205] as implemented by biotite [56]. We then calculate the fraction of surface exposed atoms within the structure corresponding to the constraint's subtree, and we use one minus this fraction as the output of the function.

A.4.2.11 Length

The length constraint requires a user-specified number of residues. In practice, we can enforce a hard length constraint by disallowing insertions and deletions during the optimization procedure, or through a function that returns increasingly high values when a sequence length goes beyond a user-specified range. In this study, whenever we apply a length constraint we take the former approach.

A.4.3 Design tasks and experiments

A.4.3.1 Free hallucination

Free hallucination simply requires applying confidence and surface-exposed hydrophobic constraints to the whole protein, where we place equal weight on each term (pTM, pLDDT, and hydrophobics). In the experiments described in this study, we ran simulated annealing over 30,000 iterations with $T_{\text{max}} = 1$ across 200 seeds. We also evaluated single-sequence AlphaFold2 [3] on the final sequences produced by these 200 optimization runs.

A.4.3.2 Fixed backbone design

For fixed backbone design, we apply a weight of 2 on the dRMSD constraint, a weight of 1 on the cRMSD, pTM, and pLDDT constraints, and a weight of 0.5 on the hydrophobics constraint. As the target backbones, we used the de novo structures with PDB IDs 1QYS, 5L33, 6D0T, 6MRS, 6W3W, and 6WVS. In the experiments described in this study, we ran simulated annealing over 30,000 iterations with $T_{\text{max}} = 1$ across at least 50 seeds for each de novo backbone.

A.4.3.3 Secondary structure design

We performed protein design with partial constraints on a protein by constraining the secondary structure corresponding to different segments of the protein sequence. We place a weight of 10 on the secondary structure constraint and weights of 1 on pTM, pLDDT and hydrophobics constraints. Our programs specify the secondary structure corresponding to two discrete subsequences, where we program (1) all alpha, (2) all beta, and (3) mixed alpha and beta secondary structure. We ran simulated annealing over 30,000 iterations $T_{\max} = 1$ for 10 seeds for each of these three programs (30 optimization trajectories in total).

A.4.3.4 Single functional site scaffolding

To program functional site scaffolding on a de novo backbone, we divide a single-chain sequence into three segments: a sequence in the middle segment (with an all-atom coordination constraint and a surface exposure constraint) flanked by two “free” sequences. pTM, pLDDT, and hydrophobics constraints are also applied to the full protein. We apply a weight of 2 to the cRMSD and dRMSD constraints, and a weight of 1 to the pTM, pLDDT, and hydrophobics constraints.

We attempted to scaffold five protein binding sites, the first three of which were successfully scaffolded by Wang et al. [159]:

1. IL10: We used the residue indices 31–40, inclusive, of chain L in the PDB structure 1Y6K, corresponding to the IL10 binding site of IL-10R1 [208].

2. ACE2: We used the residue indices 5–23, inclusive, of chain A in the PDB structure 6M0J, corresponding to the ACE2 binding site of the SARS-CoV-2 spike receptor binding domain (RBD) [10].

3. C3d: We used the residue indices 104–126 and 170–184, inclusive, of chain A in the

PDB structure 1GHQ, corresponding to the C3d binding site of complement receptor 2 [209].

4. HA2: We used the residue indices 14–21, 33–42, and 45–49, inclusive, of chain B in the PDB structure 5JW3, corresponding to the influenza HA2 epitope of the antibody MEDI8825 [210].

5. RBD: We used the residue indices 439–450 and 498–506, inclusive, of chain C in the PDB structure 7MMO, corresponding to the SARS-CoV-2 RBD epitope of the antibody bebtelovimab [211].

We ran simulated annealing over 30,000 iterations with $T_{\max} = 1$ for 1,000 seeds for each of the five binding sites (5,000 optimization trajectories in total).

A.4.3.5 Symmetric and homo-oligomer design

We first program single-chain symmetry using a rotational symmetry constraint applied to the top-level node. In our program, we also tie the sequences across the subsequences corresponding to the asymmetric units such that we only use a single terminal symbol; an example program for designing 3-fold symmetry is provided in Figure 5.3a. We place a weight of 1 on the symmetry constraint, as well as weights of 1 on pTM, pLDDT, and hydrophobics constraints. We also place length constraints on the terminal nodes. We specify programs where we increase the fold-symmetry from 3- to 8-fold. We also vary the lengths that constrain the terminal symbol such that the full sequence has approximately 200, 300, or 400 residues (for example, a 200-residue protein with 3-fold symmetry would have length constraints of 66 on its terminal symbols). We ran simulated annealing over 30,000 iterations with a starting temperature of 1 for 10 seeds for each of the six fold symmetries and each of the three length constraints (for a total of 180 optimization trajectories).

We also designed larger homo-oligomers similarly, but removing the single-chain constraint from the top-level node. We designed trimeric, tetrameric, hexameric, and octameric homo-oligomers with a globular symmetry constraint applied to the top level node. We placed a weight of 1 on the symmetry constraint, as well as weights of 1 on pTM, pLDDT, and hydrophobics constraints. We also placed a weight of 0.1 on globularity constraints that are applied to each terminal symbol. We applied length constraints such that the full complex contained 720 residues (for example, the hexamer would consist of length-120 protomers). We ran simulated annealing over 30,000 iterations with $T_{\text{max}} = 1$ for 10 seeds for each of oligomerization levels (for a total of 40 optimization trajectories).

A.4.3.6 Two-level symmetry design

We program two levels of symmetry using the productions

$$x_1 \to x_2 \cdots \quad \text{(top-level)} \quad \text{and} \tag{A.10}$$

$$x_2 \to A \cdots \quad \text{(bottom-level).} \tag{A.11}$$

In these programs, we place the single-chain constraint on x_2, so the final designs are protein homo-oligomers. We place a globularity symmetry constraint on x_1; to control the top-level symmetry, we repeat x_2 according to the desired oligomerization. We place a rotational symmetry constraint on x_2; to control the bottom-level symmetry, we repeat A according to desired fold symmetry. We also place pTM, pLDDT, and hydrophobics constraints on the full protein; we place globularity constraints on x_2. We compile constraints into an energy function with weights of 1 on all terms.

We enumerated programs over the grid varying both the top and bottom levels of symmetry from 2 to 4. We constrained lengths to 200 residues in total for the dimer of 2-fold; length-250 for the dimer of 3-fold; length-400 for the dimer of 4-fold, the trimer of 2-fold, the trimer of 3-fold, and the tetramer of 2-fold; length-450 for the trimer of 4-fold and the tetramer of 3-fold; and length-500 for the tetramer of 4-fold. We ran simulated annealing over 30,000 iterations with $T_{\max} = 1$ for 10 seeds for each of these programs (for a total of 90 optimization trajectories).

A.4.3.7 Structural novelty

We quantify a given design for structural novelty by running an exhaustive search over the PDB version 2022-08 (http://www.rcsb.org/) [81] to find the experimental structure with the highest TM-score to the designed structure, normalizing by the designed structure length, using TM-align version 20210107 [1].

A.4.3.8 Inverse folding roundtrip experiments

We assessed the "designability" of a structure prediction produced by our optimization procedure by "roundtripping" the protein through an inverse folding model. More specifically, given a predicted structure from our optimization loops, we first use ESM-IF1 [6], an independently trained inverse folding model, to sample 10 sequences with temperature 0.1 from the backbone coordinates. We then run these sequences through ESMFold and compute the cRMSD between the starting and the roundtripped backbone atoms of the predicted structure.

We performed this roundtrip experiment for 1,000 predicted structures that were obtained by first uniformly sampling one of the 180 symmetric single-chain optimization trajectories and then uniformly sampling one of the intermediate structure predictions within a given design loop (i.e., we do not restrict this analysis to the best pLDDT structure over a design loop, which are highly biased toward high pLDDTs). We report the relationship between ESM-IF1 perplexities of all sample structures and the corresponding cRMSD values. We also report the relationship between the pLDDT of the starting structure and the minimum

RMSD over the structure for the 10 inverse-folded sequences. We also repeated the same experiment for 1,000 predicted structures that were obtained by first uniformly sampling over the 90 two-level symmetry optimization trajectories and then uniformly sampling one of the intermediate structure predictions within a given design loop. We also report the same metrics as in the single-chain evaluation.

A.4.3.9 Symmetric functional site scaffolding

We designed proteins that symmetrically scaffold multiple functional sites by using the tree described for the single-site functional scaffold but replicating it according to the desired fold symmetry and adding a rotational symmetry constraint to the top-level node; an example program for a 3-fold functional site scaffold can be found in Figure 5.5a. We use weights of 10 on the cRMSD and dRMSD constraints and weights of 1 on the pTM, pLDDT, rotational symmetry, binding site surface exposure, and hydrophobics constraints. We ran simulated annealing over 30,000 iterations with a starting temperature of 1 over 20 seeds for the design of 3-fold scaffolds of the IL10 and ACE2 binding sites described above, as well as 20 seeds for the design of 5-fold scaffolds of the ACE2 binding site (for a total of 60 optimization loops).

A.4.3.10 Hierarchical asymmetric symmetry design

We increased the level of hierarchical complexity in our programs by designing with three levels of constraints. The top level specifies two asymmetric subunits. Each asymmetric subunit itself has two-level symmetry (similar to the setting described above): we specifically consider the dimer of 2-fold (2x2), the dimer of 3-fold (2x3), and the trimer of 2-fold (3x2). We write programs consisting of (1) two asymmetric 2x2s complexed together, (2) a 2x2 and a 3x2 complexed together, and (3) a 2x2 and a 2x3 completed together. An example program for the two asymmetric 2x2s is provided in Figure 5.5c. We use weights of 1 on all constraints (pTM, pLDDT, hydrophobics, rotational/globular symmetry, and globularity).

We ran simulated annealing over 30,000 iterations with $T_{\text{max}} = 1$ over 10 seeds for each of the three programs described above (for a total of 30 optimization loops).

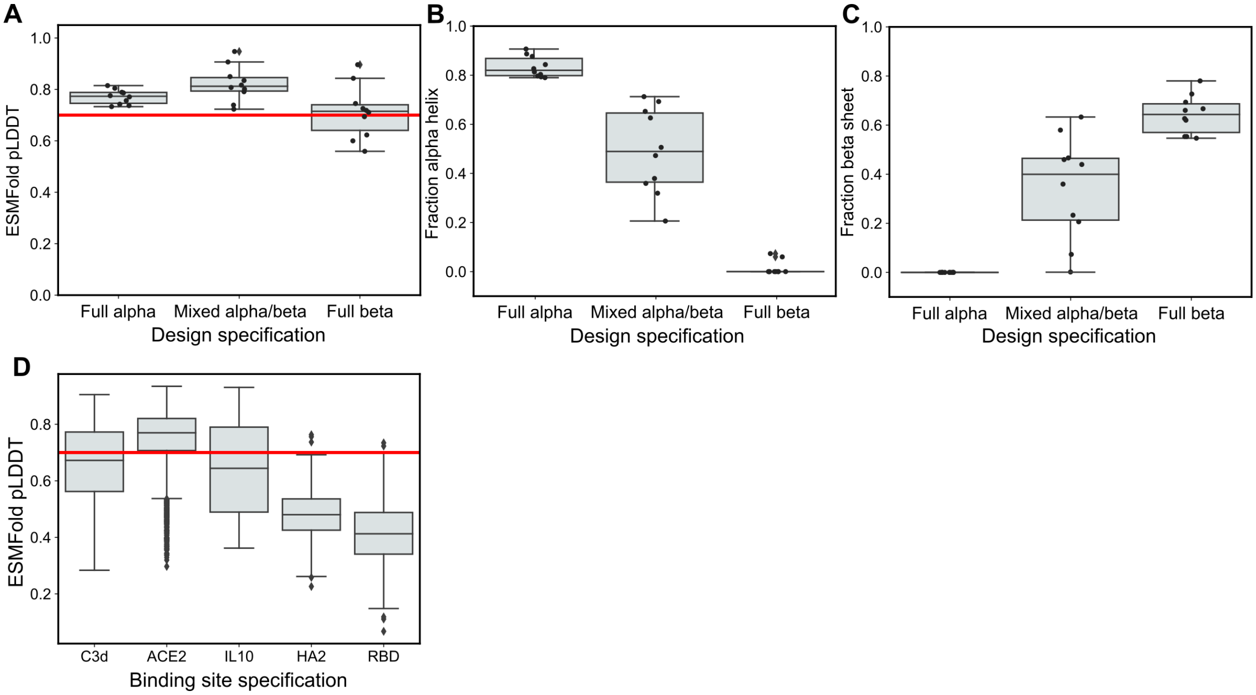

Figure A.20: *Additional plots for secondary structure design and functional site scaffolding.* (**A**) ESMFold pLDDT values for different secondary structure design specifications (10 seeds per specification). A red line is plotted at pLDDT = 0.7. (**B**) The fraction of residues that are part of alpha helices for different secondary structure design specifications (10 seeds per specification). (**C**) The fraction of residues that are part of beta sheets for different secondary structure design specifications (10 seeds per specification). (**D**) ESMFold pLDDT values for different functional site scaffolding design runs (1,000 seeds per binding site).

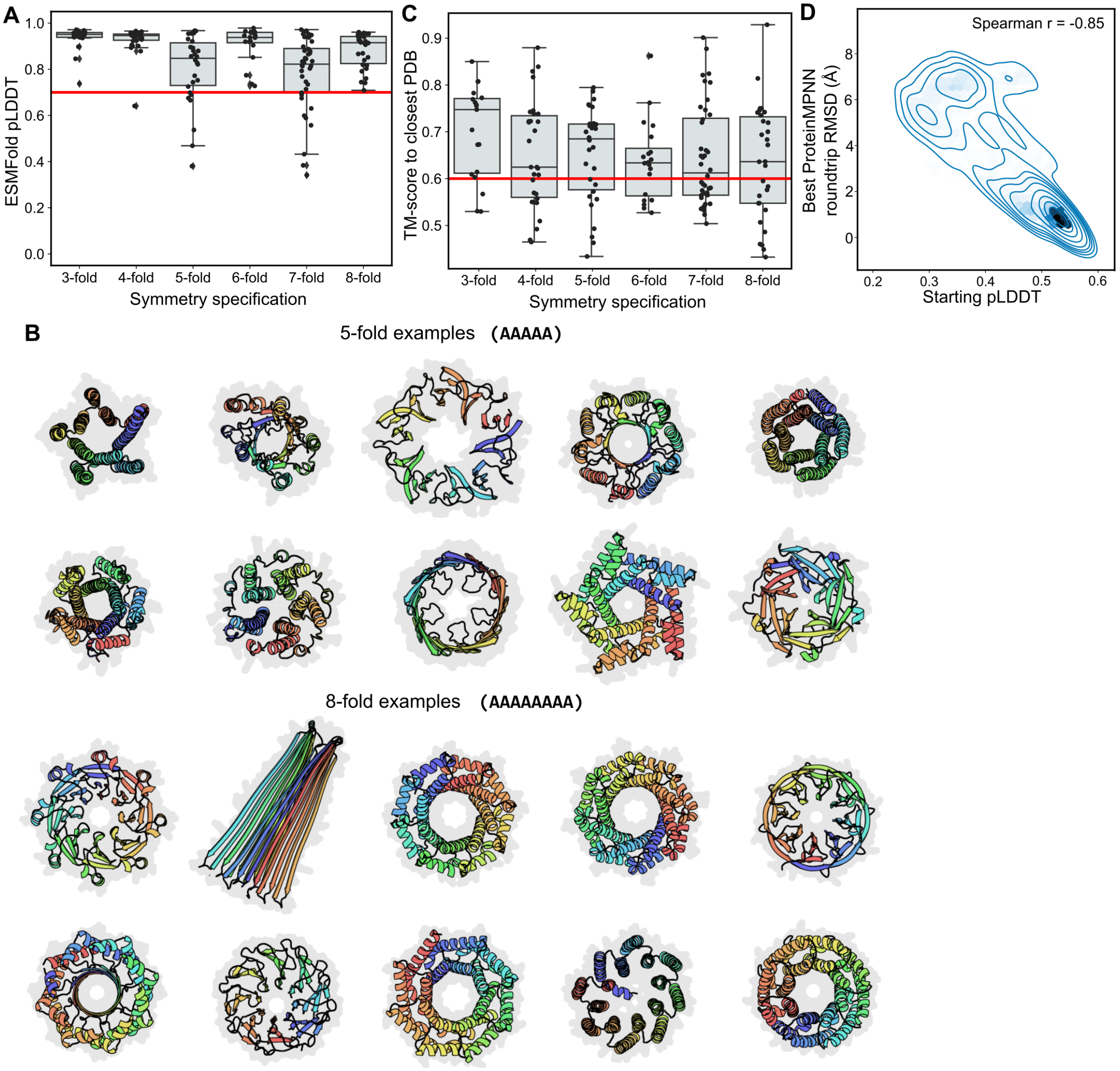

Figure A.21: *Additional plots for the design of symmetric single chains.* (**A**) ESMFold pLDDT values for different fold symmetry design specifications (30 seeds per specification). A red line is plotted at pLDDT = 0.7. (**B**) Ten randomly sampled designs for the design of 5- and 8-fold symmetry. (**C**) TM-scores for different fold symmetry design specifications (30 seeds per specification); the TM-score is between the best design and the closest structure in the PDB. A red line is plotted at TM-score = 0.6. (**D**) Samples obtained by inverse folding with ProteinMPNN. On the x-axis is the pLDDT of the designed structure prior to the roundtrip and on the y-axis is the roundtrip RMSD.

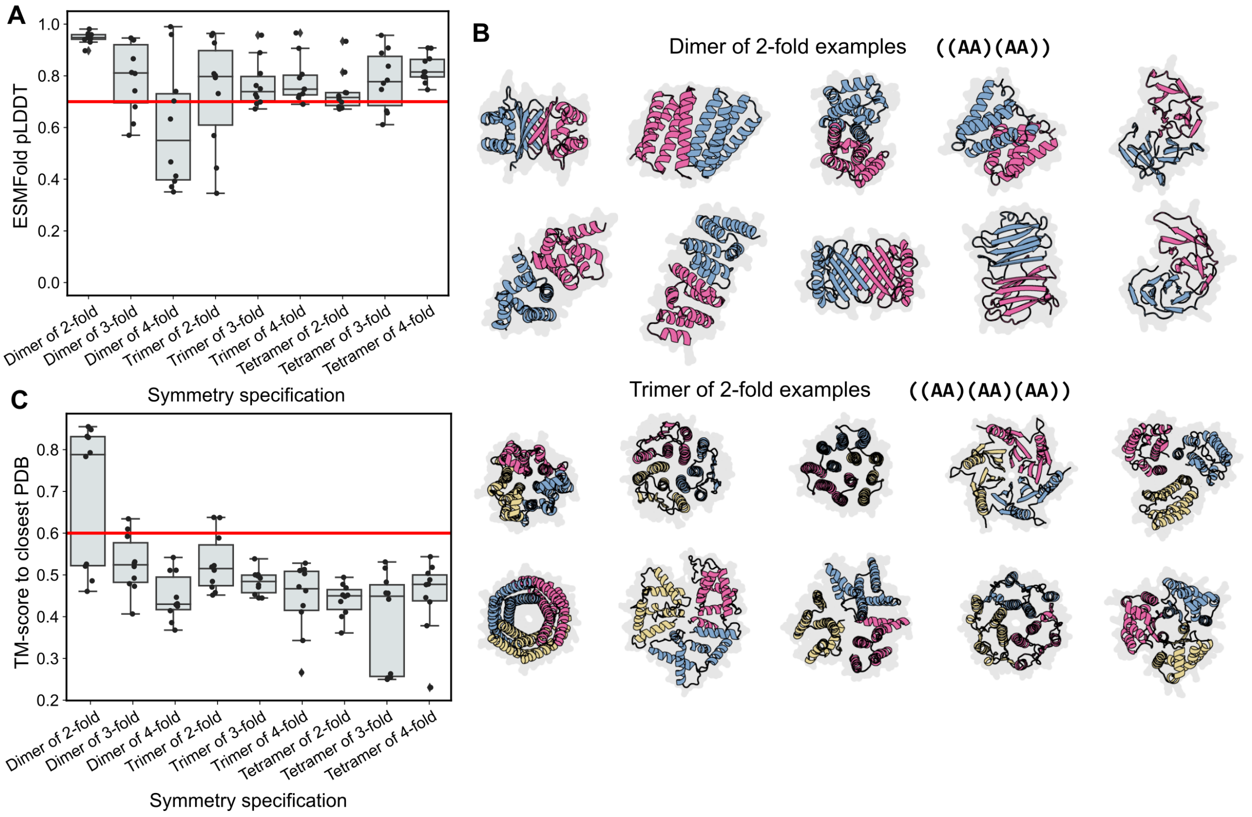

Figure A.22: *Additional plots for the design of two-level symmetry.* (**A**) ESMFold pLDDT values for different two-level symmetry design specifications (10 seeds per specification). A red line is plotted at pLDDT = 0.7. (**B**) All ten of the designs of a dimer of 2-fold and of a trimer of 2-fold symmetry. (**C**) TM-scores for different two-level symmetry design specifications (10 seeds per specification); the TM-score is between the best design and the closest structure in the PDB. A red line is plotted at TM-score = 0.6.

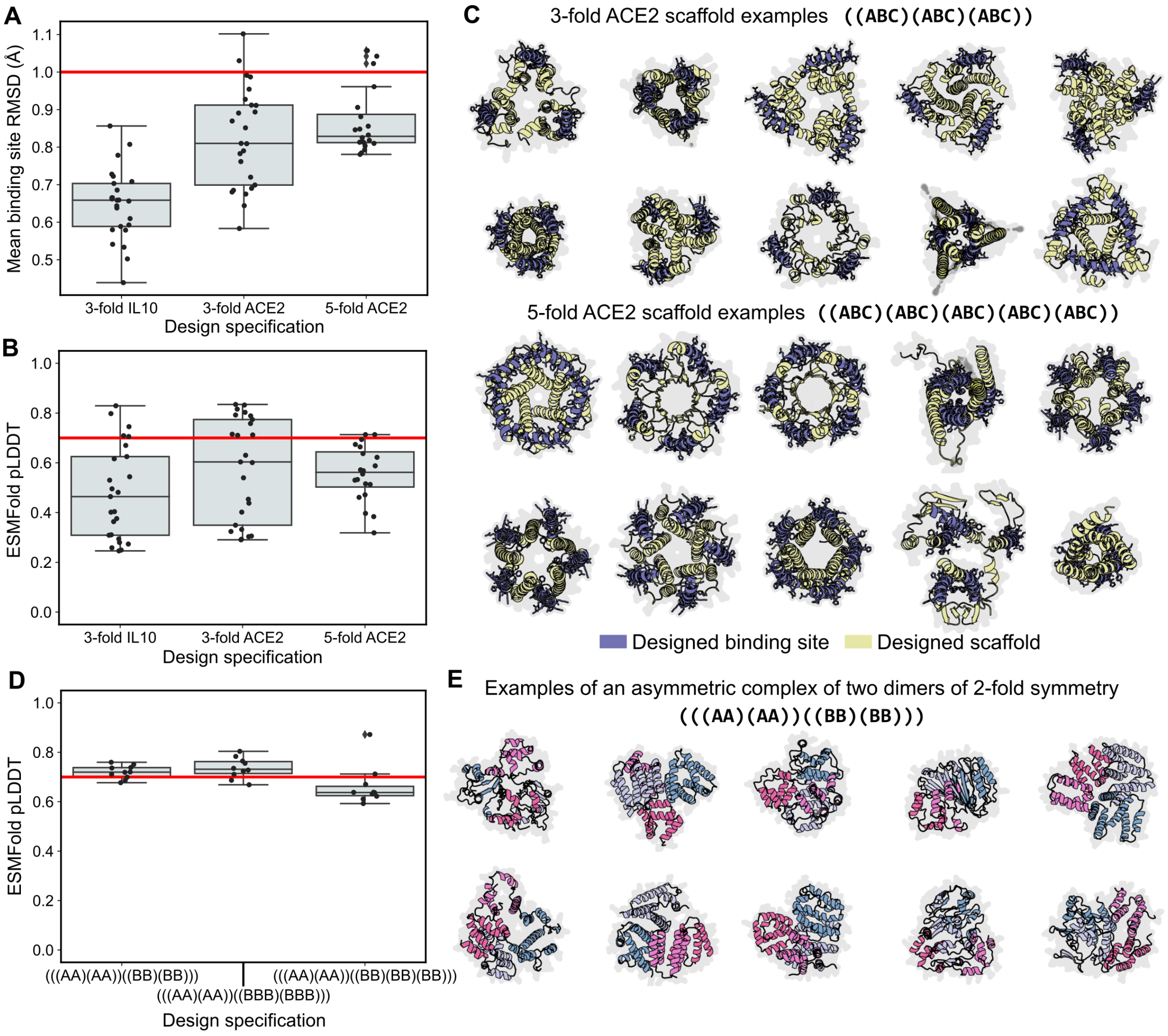

Figure A.23: *Additional plots for multi-level hierarchical design* (**A**) ESMFold pLDDT values for different binding site scaffolds with fold symmetry (20 seeds per specification). A red line is plotted at pLDDT = 0.7. (**B**) RMSD values for different binding site scaffolds with fold symmetry (20 seeds per specification). The mean RMSD is reported across either three or five binding sites. A red line is plotted at pLDDT = 0.7. (**C**) Ten randomly sampled designs for the design of 3- and 5-fold symmetric ACE2 binding site scaffolds. (**D**) ESMFold pLDDT values for different asymmetric-symmetric design specifications (10 seeds per specification). A red line is plotted at pLDDT = 0.7. (**E**) All ten of the designs of an asymmetric complex of two dimers of 2-fold symmetry.

Bibliography

[1] Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Res.*, 33(7):2302–2309, April 2005. ISSN 0305-1048. doi: 10.1093/nar/gki524. URL https://doi.org/10.1093/nar/gki524.

[2] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[3] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596 (7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL https://www.nature.com/articles/s41586-021-03819-2. Bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Number: 7873 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computational biophysics;Machine learning;Protein structure predictions;Structural biology Subject_term_id: computational-biophysics;machine-learning;protein-structure-predictions;structural-biology.

[4] Sankar Basu and Björn Wallner. DockQ: A Quality Measure for Protein-Protein Docking Models. *PLOS ONE*, 11(8):e0161879, August 2016. ISSN 1932-6203. doi: 10.1371/journal.pone.0161879. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0161879. Publisher: Public Library of Science.

[5] Michel van Kempen, Stephanie Kim, Charlotte Tumescheit, Milot Mirdita, Johannes Söding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *bioRxiv*, February 2022. doi: 10.1101/2022.02.07.479398. URL http://biorxiv.org/lookup/doi/10.1101/2022.02.07.479398.

[6] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In

Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8946–8970. PMLR, June 2022. URL https://proceedings.mlr.press/v162/hsu22a.html. ISSN: 2640-3498.

[7] Alexandra C Walls, Young-Jun Park, M Alejandra Tortorici, Abigail Wall, Andrew T McGuire, and David Veesler. Structure, function, and antigenicity of the sars-cov-2 spike glycoprotein. *Cell*, 181(2):281–292, 2020.

[8] Milot Mirdita, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. ColabFold: making protein folding accessible to all. *Nature Methods*, 19(6):679–682, June 2022. ISSN 1548-7091, 1548-7105. doi: 10.1038/s41592-022-01488-1. URL https://www.nature.com/articles/s41592-022-01488-1.

[9] Tyler N Starr, Allison J Greaney, Sarah K Hilton, Daniel Ellis, Katharine HD Crawford, Adam S Dingens, Mary Jane Navarro, John E Bowen, M Alejandra Tortorici, Alexandra C Walls, et al. Deep mutational scanning of sars-cov-2 receptor binding domain reveals constraints on folding and ace2 binding. *Cell*, 182(5):1295–1310, 2020.

[10] Jun Lan, Jiwan Ge, Jinfang Yu, Sisi Shan, Huan Zhou, Shilong Fan, Qi Zhang, Xuanling Shi, Qisheng Wang, Linqi Zhang, and Xinquan Wang. Structure of the SARS-CoV-2 spike receptor-binding domain bound to the ACE2 receptor. *Nature*, 581(7807):215–220, May 2020.

[11] Gabriel J Rocklin, Tamuka M Chidyausiku, Inna Goreshnik, Alex Ford, Scott Houliston, Alexander Lemak, Lauren Carter, Rashmi Ravichandran, Vikram K Mulligan, Aaron Chevalier, et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357(6347):168–175, July 2017. ISSN 10959203. doi: 10.1126/science.aan0693. URL http://science.sciencemag.org/. Publisher: American Association for the Advancement of Science.

[12] Raphael J. L. Townshend, Martin Vögele, Patricia Suriana, Alexander Derry, Alexander Powers, Yianni Laloudakis, Sidhika Balachandar, Brandon M. Anderson, Stephan Eismann, Risi Kondor, Russ B. Altman, and Ron O. Dror. ATOM3D: tasks on molecules in three dimensions. *CoRR*, abs/2012.04035, 2020.

[13] Bowen Jing, Stephan Eismann, Pratham N Soni, and Ron O Dror. Equivariant graph neural networks for 3d macromolecular structure. *Proceedings of the International Conference on Machine Learning*, 2021.

[14] Christian Dallago, Jody Mou, Kadina E Johnston, Bruce J Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, 2021.

[15] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

[16] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, April 2021. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.2016239118. URL https://www.pnas.org/content/118/15/e2016239118. Publisher: National Academy of Sciences Section: Biological Sciences.

[17] Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2014. Publisher: Oxford University Press.

[18] Helen Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide Protein Data Bank. *Nature Structural & Molecular Biology*, 10(12):980–980, December 2003. ISSN 1545-9985. doi: 10.1038/nsb1203-980. URL https://www.nature.com/articles/nsb1203-980.

[19] Rebecca F Alford, Andrew Leaver-Fay, Jeliazko R Jeliazkov, Matthew J O'Meara, Frank P DiMaio, Hahnbeom Park, Maxim V Shapovalov, P Douglas Renfrew, Vikram K Mulligan, Kalli Kappel, et al. The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, 13(6):3031–3048, 2017.

[20] Namrata Anand and Possu Huang. Generative modeling for protein structures. *Advances in Neural Information Processing Systems*, 31, 2018. ISSN 10495258. URL https://papers.nips.cc/paper/2018/hash/afa299a4d1d8c52e75dd8a24c3ce534f-Abstract.html.

[21] Raphael R. Eguchi, Namrata Anand, Christian A. Choe, and Po-Ssu Huang. IG-VAE: Generative Modeling of Immunoglobulin Proteins by Direct 3D Coordinate Generation. *bioRxiv*, page 2020.08.07.242347, August 2020. doi: 10.1101/2020.08.07.242347. URL https://www.biorxiv.org/content/10.1101/2020.08.07.242347v1.

[22] John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative Models for Graph-Based Protein Design. page 12, 2019. URL https://papers.nips.cc/paper/9711-generative-models-for-graph-based-protein-design.

[23] Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael JL Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. *arXiv:2009.01411*, 2020.

[24] Jinbo Xu. Distance-based protein folding powered by deep learning. *Proceedings of the National Academy of Sciences*, 116(34):16856–16865, August 2019. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1821309116. URL https://www.pnas.org/content/116/34/16856. Publisher: Proceedings of the National Academy of Sciences.

[25] 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, January 2020. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1914677117. URL https://www.pnas.org/content/117/3/1496. Publisher: National Academy of Sciences Section: Biological Sciences.

[26] Pehr B Harbury, Joseph J Plecs, Bruce Tidor, Tom Alber, and Peter S Kim. High-resolution protein design with backbone freedom. *Science*, 282(5393):1462–1467, November 1998.

[27] Brian Kuhlman, Gautam Dantas, Gregory C Ireton, Gabriele Varani, Barry L Stoddard, and David Baker. Design of a novel globular protein fold with atomic-level accuracy. *Science*, 302(5649):1364–1368, November 2003.

[28] Hui-Hsu Gavin Tsai, Chung-Jung Tsai, Buyong Ma, and Ruth Nussinov. In silico protein design by combinatorial assembly of protein building blocks. *Protein Sci.*, 13 (10):2753–2765, October 2004.

[29] Nobuyasu Koga, Rie Tatsumi-Koga, Gaohua Liu, Rong Xiao, Thomas B Acton, Gaetano T Montelione, and David Baker. Principles for designing ideal protein structures. *Nature*, 491(7423):222–227, November 2012.

[30] H Zhao, L Giver, Z Shao, J A Affholter, and F H Arnold. Molecular evolution by staggered extension process (StEP) in vitro recombination. *Nat. Biotechnol.*, 16(3): 258–261, March 1998.

[31] Christopher A Voigt, Carlos Martinez, Zhen-Gang Wang, Stephen L Mayo, and Frances H Arnold. Protein building blocks preserved by recombination. *Nat. Struct. Biol.*, 9(7):553–558, July 2002.

[32] Marc Ostermeier. Engineering allosteric protein switches by domain insertion. *Protein Eng. Des. Sel.*, 18(8):359–364, August 2005.

[33] Joe G. Greener, Lewis Moffat, and David T. Jones. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific Reports*, 8(1):16189, November 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-34533-1. URL https://www.nature.com/articles/s41598-018-34533-1.

[34] Ivan Anishchenko, Samuel J Pellock, Tamuka M Chidyausiku, Theresa A Ramelot, Sergey Ovchinnikov, Jingzhou Hao, Khushboo Bafna, Christoffer Norn, Alex

Kang, Asim K Bera, et al. De novo protein design by deep network hallucination. *Nature*, 600(7889):547–552, December 2021. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-021-04184-w. URL https://www.nature.com/articles/s41586-021-04184-w.

[35] Namrata Anand, Raphael Eguchi, and Po-Ssu Huang. Fully differentiable full-atom protein backbone generation. March 2019. URL https://openreview.net/forum?id=SJxnVL8YOV.

[36] Mostafa Karimi, Shaowen Zhu, Yue Cao, and Yang Shen. De novo protein design for novel folds using guided conditional wasserstein generative adversarial networks (gcwgan). *bioRxiv*, page 769919, 2019.

[37] Zeming Lin, Tom Sercu, Yann LeCun, and Alexander Rives. Deep generative models create new and diverse protein structures. In *Machine Learning for Structural Biology Workshop, NeurIPS*, 2021.

[38] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023.

[39] Brian Hie, Salvatore Candido, Zeming Lin, Ori Kabeli, Roshan Rao, Nikita Smetanin, Tom Sercu, and Alexander Rives. A high-level programming language for generative protein design. *bioRxiv*, pages 2022–12, 2022.

[40] Cyrus Levinthal. How to fold graciously. *Mossbauer spectroscopy in biological systems*, 67:22–24, 1969.

[41] Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted inter-residue orientations. *bioRxiv*, page 846279, 2019.

[42] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Kathryn Tunyasuvunakool, Olaf Ronneberger, Russ Bates, Augustin Žídek, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Anna Potapenko, Andrew J Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Martin Steinegger, Michalina Pacholska, David Silver, Oriol Vinyals, Andrew W Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. High Accuracy Protein Structure Prediction Using Deep Learning. In *Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book)*, volume 22, page 22. 2020.

[43] Andrew Leaver-Fay, Michael Tyka, Steven M. Lewis, Oliver F. Lange, James Thompson, Ron Jacak, Kristian Kaufman, P. Douglas Renfrew, Colin A. Smith, Will Sheffler, Ian W. Davis, Seth Cooper, Adrien Treuille, Daniel J. Mandell, Florian Richter, Yih-En Andrew

Ban, Sarel J. Fleishman, Jacob E. Corn, David E. Kim, Sergey Lyskov, Monica Berrondo, Stuart Mentzer, Zoran Popović, James J. Havranek, John Karanicolas, Rhiju Das, Jens Meiler, Tanja Kortemme, Jeffrey J. Gray, Brian Kuhlman, David Baker, and Philip Bradley. ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *Methods in Enzymology*, 487:545–574, 2011. ISSN 1557-7988. doi: 10.1016/B978-0-12-381270-4.00019-6.

[44] Naomi K. Fox, Steven E. Brenner, and John-Marc Chandonia. SCOPe: Structural Classification of Proteins—extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research*, 42(D1):D304–D309, January 2014. ISSN 0305-1048. doi: 10.1093/nar/gkt1240. URL https://doi.org/10.1093/nar/gkt1240. Publisher: Oxford University Press.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity Mappings in Deep Residual Networks. *arXiv:1603.05027 [cs]*, July 2016. URL http://arxiv.org/abs/1603.05027. arXiv: 1603.05027.

[46] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. MLP-Mixer: An all-MLP Architecture for Vision. *arXiv:2105.01601 [cs]*, May 2021. URL http://arxiv.org/abs/2105.01601. arXiv: 2105.01601.

[47] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, May 2014. URL http://arxiv.org/abs/1312.6114. arXiv: 1312.6114.

[48] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder Variational Autoencoders. *arXiv:1602.02282 [cs, stat]*, May 2016. URL http://arxiv.org/abs/1602.02282. arXiv: 1602.02282.

[49] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving Variational Inference with Inverse Autoregressive Flow. *arXiv:1606.04934 [cs, stat]*, January 2017. URL http://arxiv.org/abs/1606.04934. arXiv: 1606.04934.

[50] Arash Vahdat and Jan Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. *arXiv:2007.03898 [cs, stat]*, January 2021. URL http://arxiv.org/abs/2007.03898. arXiv: 2007.03898.

[51] Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. September 2020. URL https://openreview.net/forum?id=RLRXCV6DbEJ.

[52] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. *arXiv:1711.00937 [cs]*, May 2018. URL http://arxiv.org/abs/1711.00937. arXiv: 1711.00937.

[53] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. *arXiv:1906.00446 [cs, stat]*, June 2019. URL http://arxiv.org/abs/1906.00446. arXiv: 1906.00446.

[54] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-Shot Text-to-Image Generation. *arXiv:2102.12092 [cs]*, pages 8821–8831, February 2021. URL http://arxiv.org/abs/2102.12092. arXiv: 2102.12092.

[55] Andrew W. Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander W. R. Nelson, Alex Bridgland, Hugo Penedones, Stig Petersen, Karen Simonyan, Steve Crossan, Pushmeet Kohli, David T. Jones, David Silver, Koray Kavukcuoglu, and Demis Hassabis. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792): 706–710, January 2020. ISSN 1476-4687. doi: 10.1038/s41586-019-1923-7. URL https://www.nature.com/articles/s41586-019-1923-7. Publisher: Nature Research.

[56] Patrick Kunzmann and Kay Hamacher. Biotite: a unifying open source computational biology framework in Python. *BMC Bioinformatics*, 19(1):346, October 2018. ISSN 1471-2105. doi: 10.1186/s12859-018-2367-z. URL https://doi.org/10.1186/s12859-018-2367-z.

[57] E. N. Baker and R. E. Hubbard. Hydrogen bonding in globular proteins. *Progress in Biophysics and Molecular Biology*, 44(2):97–179, 1984. ISSN 0079-6107. doi: 10.1016/0079-6107(84)90007-5.

[58] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 1983.

[59] Alex L Mitchell, Alexandre Almeida, Martin Beracochea, Miguel Boland, Josephine Burgin, Guy Cochrane, Michael R Crusoe, Varsha Kale, Simon C Potter, Lorna J Richardson, Ekaterina Sakharova, Maxim Scheremetjew, Anton Korobeynikov, Alex Shlemov, Olga Kunyavskaya, Alla Lapidus, and Robert D Finn. MGnify: the microbiome analysis resource in 2020. *Nucleic Acids Research*, 48(D1):D570–D578, January 2020. ISSN 0305-1048. doi: 10.1093/nar/gkz1035. URL https://doi.org/10.1093/nar/gkz1035.

[60] L. Steven Johnson, Sean R. Eddy, and Elon Portugaly. Hidden Markov model speed heuristic and iterative HMM search procedure. *BMC Bioinformatics*, 11(1):431, August 2010. ISSN 1471-2105. doi: 10.1186/1471-2105-11-431. URL https://doi.org/10.1186/1471-2105-11-431. Publisher: BioMed Central.

[61] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[62] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 2006.

[63] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[64] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[65] Sheng Wang, Siqi Sun, Zhen Li, Renyu Zhang, and Jinbo Xu. Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. *PLOS Computational Biology*, 13(1):1–34, January 2017. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1005324. URL https://dx.plos.org/10.1371/journal.pcbi.1005324. Publisher: Public Library of Science.

[66] Kristian Davidsen, Branden J Olson, William S DeWitt III, Jean Feng, Elias Harkins, Philip Bradley, and Frederick A Matsen IV. Deep generative models for t cell receptor protein sequences. *Elife*, 8:e46935, 2019.

[67] Alex Hawkins-Hooker, Florence Depardieu, Sebastien Baur, Guillaume Couairon, Arthur Chen, and David Bikard. Generating functional protein variants with variational autoencoders. *PLoS computational biology*, 17(2):e1008736, February 2021. doi: 10.1371/journal.pcbi.1008736. URL https://doi.org/10.1371/journal.pcbi.1008736. Publisher: Public Library of Science.

[68] Donatas Repecka, Vykintas Jauniskis, Laurynas Karpus, Elzbieta Rembeza, Irmantas Rokaitis, Jan Zrimec, Simona Poviloniene, Audrius Laurynenas, Sandra Viknander, Wissam Abuajwa, et al. Expanding functional protein sequence spaces using generative adversarial networks. *Nature Machine Intelligence*, 3(4):324–333, April 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00310-5. URL https://www.nature.com/articles/s42256-021-00310-5. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 4 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Machine learning;Protein engineering;Protein sequence analyses Subject_term_id: machine-learning;protein-engineering;protein-sequence-analyses.

[69] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 16(12):1–8, 2019.

[70] Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R. Eguchi, Po-Ssu Huang, and Richard Socher. ProGen: Language Modeling for Protein Generation. *bioRxiv*, March 2020. URL http://arxiv.org/abs/2004.03497.

[71] Jingxue Wang, Huali Cao, John Z. H. Zhang, and Yifei Qi. Computational Protein Design with Deep Learning Neural Networks. *Scientific Reports*, 8(1):6349, April 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-24760-x. URL https://www.nature.com/articles/s41598-018-24760-x.

[72] Alexey Strokach, David Becerra, Carles Corbi-Verge, Albert Perez-Riba, and Philip M Kim. Fast and flexible design of novel proteins using graph neural networks. *bioRxiv*, 11(4):868935, 2020.

[73] Namrata Anand, Raphael Ryuichi Eguchi, Alexander Derry, Russ B Altman, and Possu Huang. Protein sequence design with a learned potential. *bioRxiv*, 2020.

[74] Po-Ssu Huang, Scott E. Boyken, and David Baker. The coming of age of de novo protein design. *Nature*, 537(7620):320–327, September 2016. ISSN 1476-4687. doi: 10.1038/nature19946. Publisher: Nature Publishing Group.

[75] Namrata Anand-Achim, Raphael R Eguchi, Irimpan I Mathews, Carla P Perez, Alexander Derry, Russ B Altman, and Po-Ssu Huang. Protein sequence design with a learned potential. *Biorxiv*, pages 2020–01, 2021.

[76] Jue Wang, Sidney Lisanza, David Juergens, Doug Tischer, Ivan Anishchenko, Minkyung Baek, Joseph L Watson, Jung Ho Chun, Lukas F Milles, Justas Dauparas, et al. Deep learning methods for designing proteins scaffolding functional sites. *bioRxiv*, 2021.

[77] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Deep neural language modeling enables functional protein generation across families. *bioRxiv*, 2021. doi: 10.1101/2021.07.18.452833.

[78] Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):1–11, 2021.

[79] Vladimir Gligorijevic, Daniel Berenberg, Stephen Ra, Andrew Watkins, Simon Kelow, Kyunghyun Cho, and Richard Bonneau. Function-guided protein design by deep manifold sampling. *bioRxiv*, 2021.

[80] Drew H Bryant, Ali Bashir, Sam Sinai, Nina K Jain, Pierce J Ogden, Patrick F Riley, George M Church, Lucy J Colwell, and Eric D Kelsic. Deep diversification of an aav capsid protein by machine learning. *Nature Biotechnology*, 39(6):691–696, 2021.

[81] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, January 2000. ISSN 0305-1048. doi: 10.1093/nar/28.1.235. Publisher: Oxford University Press.

[82] Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer, March 2022. URL https://www.biorxiv.org/content/10.1101/2021.10.04.463034v2. Pages: 2021.10.04.463034 Section: New Results.

[83] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv:1511.06709*, 2015.

[84] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

[85] Roshan Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. MSA Transformer. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8844–8856. PMLR, July 2021. URL https://proceedings.mlr.press/v139/rao21a.html. ISSN: 2640-3498.

[86] Viktor Hornak, Robert Abel, Asim Okur, Bentley Strockbine, Adrian Roitberg, and Carlos Simmerling. Comparison of multiple amber force fields and development of improved protein backbone parameters. *Proteins: Structure, Function, and Bioinformatics*, 65 (3):712–725, 2006.

[87] Christine A Orengo, Alex D Michie, Susan Jones, David T Jones, Mark B Swindells, and Janet M Thornton. Cath–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.

[88] Jonathan Lees, Corin Yeats, James Perkins, Ian Sillitoe, Robert Rentzsch, Benoit H Dessailly, and Christine Orengo. Gene3d: a domain-based resource for comparative genomics, functional annotation and protein network analysis. *Nucleic acids research*, 40(D1):D465–D471, 2012.

[89] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[90] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020. doi: 10.1162/tacl_a_00300. URL https://aclanthology.org/2020.tacl-1.5. Place: Cambridge, MA Publisher: MIT Press.

[91] Robert A Langan, Scott E Boyken, Andrew H Ng, Jennifer A Samson, Galen Dods, Alexandra M Westbrook, Taylor H Nguyen, Marc J Lajoie, Zibo Chen, Stephanie

Berger, et al. De novo design of bioactive protein switches. *Nature*, 572(7768):205–210, 2019.

[92] Alfredo Quijano-Rubio, Hsien-Wei Yeh, Jooyoung Park, Hansol Lee, Robert A Langan, Scott E Boyken, Marc J Lajoie, Longxing Cao, Cameron M Chow, Marcos C Miranda, et al. De novo design of modular and tunable protein biosensors. *Nature*, 591(7850): 482–487, 2021.

[93] Thomas Hrabe, Zhanwen Li, Mayya Sedova, Piotr Rotkiewicz, Lukasz Jaroszewski, and Adam Godzik. Pdbflex: exploring flexibility in protein structures. *Nucleic acids research*, 44(D1):D423–D428, 2016.

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

[95] Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 34, July 2021. doi: 10.1101/2021.07.09.450648. URL http://biorxiv.org/lookup/doi/10.1101/2021.07.09.450648.

[96] Arthur G Street and Stephen L Mayo. Computational protein design. *Structure*, 7(5): R105–R109, 1999.

[97] Bassil I Dahiyat and Stephen L Mayo. Probing the role of packing specificity in protein design. *Proceedings of the National Academy of Sciences*, 94(19):10172–10177, 1997.

[98] William F DeGrado, Daniel P Raleigh, and Tracey Handel. De novo protein design: what are we learning? *Current Opinion in Structural Biology*, 1(6):984–993, 1991.

[99] Jianfu Zhou, Alexandra E Panaitiu, and Gevorg Grigoryan. A general-purpose protein design framework based on mining sequence–structure relationships in known protein structures. *Proceedings of the National Academy of Sciences*, 117(2):1059–1068, 2020.

[100] Zhixiu Li, Yuedong Yang, Eshel Faraggi, Jian Zhan, and Yaoqi Zhou. Direct prediction of profiles of sequences compatible with a protein structure by neural networks with fragment-based local and energy-based nonlocal profiles. *Proteins: Structure, Function, and Bioinformatics*, 82(10):2565–2573, 2014.

[101] James O'Connell, Zhixiu Li, Jack Hanson, Rhys Heffernan, James Lyons, Kuldip Paliwal, Abdollah Dehzangi, Yuedong Yang, and Yaoqi Zhou. SPIN2: Predicting sequence profiles from protein structures using deep neural networks. *Proteins*, 86(6): 629–633, June 2018. ISSN 1097-0134. doi: 10.1002/prot.25489.

[102] Wouter Boomsma and Jes Frellsen. Spherical convolutions and their application in molecular modelling. *Advances in Neural Information Processing Systems*, 30:3436–3446, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/1113d7a76ffceca1bb350bfe145467c6-Abstract.html.

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

[104] Bian Li, Yucheng T Yang, John A Capra, and Mark B Gerstein. Predicting changes in protein thermodynamic stability upon point mutation with deep 3d convolutional neural networks. *PLoS computational biology*, 16(11):e1008291, 2020.

[105] Yifei Qi and John ZH Zhang. Densecpd: improving the accuracy of neural-network-based computational protein sequence design with densenet. *Journal of Chemical Information and Modeling*, 60(3):1245–1252, 2020.

[106] Yuan Zhang, Yang Chen, Chenran Wang, Chun-Chao Lo, Xiuwen Liu, Wei Wu, and Jinfeng Zhang. Prodconn: Protein design using a convolutional neural network. *Proteins: Structure, Function, and Bioinformatics*, 88(7):819–829, 2020.

[107] Sheng Chen, Zhe Sun, Lihua Lin, Zifeng Liu, Xun Liu, Yutian Chong, Yutong Lu, Huiying Zhao, and Yuedong Yang. To improve protein sequence profile prediction through image captioning on pairwise residue distance map. *Journal of chemical information and modeling*, 60(1):391–399, 2019.

[108] Christoffer Norn, Basile IM Wicky, David Juergens, Sirui Liu, David Kim, Doug Tischer, Brian Koepnick, Ivan Anishchenko, David Baker, and Sergey Ovchinnikov. Protein sequence design by conformational landscape optimization. *Proceedings of the National Academy of Sciences*, 118(11), 2021.

[109] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Alexis Courbet, Robbert J. de Haas, Neville Bethel, Philip J. Y. Leung, Timothy F. Huddy, Sam Pellock, Doug Tischer, Frederick Chan, Brian Koepnick, Hannah Nguyen, Alex Kang, Banumathi Sankaran, Asim Bera, Neil P. King, and David Baker. Robust deep learning based protein sequence design using proteinmpnn. *bioRxiv*, 2022. URL https://www.biorxiv.org/content/early/2022/06/04/2022.06.03.494563.

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

[111] Wengong Jin, Jeremy Wohlwend, Regina Barzilay, and Tommi Jaakkola. Iterative refinement graph neural network for antibody sequence-structure co-design. *arXiv preprint arXiv:2110.04624*, 2021.

[112] Namrata Anand and Tudor Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models, 2022.

[113] Jeanne Trinquier, Guido Uguzzoni, Andrea Pagnani, Francesco Zamponi, and Martin Weigt. Efficient generative modeling of protein sequences using simple autoregressive models. *arXiv preprint arXiv:2103.03292*, 2021.

[114] Tristan Bepler and Bonnie Berger. Learning protein sequence embeddings using information from structure. *arXiv:1902.08661*, 2019.

[115] Michael Heinzinger, Ahmed Elnaggar, Yu Wang, Christian Dallago, Dmitrii Nechaev, Florian Matthes, and Burkhard Rost. Modeling aspects of the language of life through transfer-learning protein sequences. *BMC bioinformatics*, 20(1):723, 2019.

[116] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S. Song. Evaluating Protein Transfer Learning with TAPE. In *Neural Information Processing Systems*, volume 32. Cold Spring Harbor Laboratory, June 2019. doi: 10.1101/676825. URL https://doi.org/10.1101/676825http://arxiv.org/abs/1906.08230.

[117] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Debsindhu Bhowmik, and Burkhard Rost. ProtTrans: Towards Cracking the Language of Lifes Code Through Self-Supervised Deep Learning and High Performance Computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):1–1, July 2021. doi: 10.1109/TPAMI.2021.3095381. URL https://www.osti.gov/pages/biblio/1817585. Institution: Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States).

[118] Zachary Wu, Kadina E Johnston, Frances H Arnold, and Kevin K Yang. Protein sequence design with deep generative models. *Current Opinion in Chemical Biology*, 65:18–27, 2021.

[119] Kevin K. Yang, Zachary Wu, and Frances H. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature Methods*, 16(8):687–694, August 2019. ISSN 1548-7105. doi: 10.1038/s41592-019-0496-6. URL https://www.nature.com/articles/s41592-019-0496-6. ISSN: 15487105 Issue: 8 Pages: 687–694 Publication Title: Nature Methods Volume: 16 _eprint: 1811.10775.

[120] Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations*, 2019.

[121] David H Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. *arXiv:1901.10060*, pages 773–782, January 2019. URL http://arxiv.org/abs/1901.10060.

[122] Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric D Kelsic. Adalead: A simple and robust adaptive greedy search algorithm for sequence design. *arXiv preprint arXiv:2010.02141*, 2020.

[123] 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, 11 2021. ISSN 0305-1048. doi: 10.1093/nar/gkab1061. URL https://doi.org/10.1093/nar/gkab1061.

[124] Ian R Humphreys, Jimin Pei, Minkyung Baek, Aditya Krishnakumar, Ivan Anishchenko, Sergey Ovchinnikov, Jing Zhang, Travis J Ness, Sudeep Banjade, Saket R Bagde, et al. Computed structures of core eukaryotic protein complexes. *Science*, 374(6573), 2021.

[125] C Yanofsky, V Horn, and D Thorpe. Protein Structure Relationships Revealed By Mutational Analysis. *Science (New York, N.Y.)*, 146(3651):1593–4, December 1964. ISSN 0036-8075. URL http://www.ncbi.nlm.nih.gov/pubmed/14224506.

[126] D Altschuh, T Vernet, P Berti, D Moras, and K Nagai. Coordinated amino acid changes in homologous protein families. *Protein Engineering, Design and Selection*, 2(3):193–199, 1988. ISSN 0269-2139. URL http://www.ncbi.nlm.nih.gov/pubmed/3237684. Publisher: Oxford University Press.

[127] Ulrike Göbel, Chris Sander, Reinhard Schneider, and Alfonso Valencia. Correlated mutations and residue contacts in proteins. *Proteins: Structure, Function, and Bioinformatics*, 18(4):309–317, 1994. ISSN 1097-0134. doi: 10.1002/prot.340180402. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.340180402. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.340180402.

[128] Alan S. Lapedes, Bertrand G. Giraud, LonChang Liu, and Gary D. Stormo. Correlated Mutations in Models of Protein Sequences: Phylogenetic and Structural Effects. *Lecture Notes-Monograph Series*, 33:236–256, 1999. ISSN 07492170. doi: 10.2307/4356049. URL http://www.jstor.org/stable/4356049. Publisher: Institute of Mathematical Statistics.

[129] John Thomas, Naren Ramakrishnan, and Chris Bailey-Kellogg. Graphical models of residue coupling in protein families, April 2008. URL https://pubmed.ncbi.nlm.nih.gov/18451428/. ISSN: 15455963 Issue: 2 Pages: 183–197 Publication Title: IEEE/ACM Transactions on Computational Biology and Bioinformatics Volume: 5.

[130] Martin Weigt, Robert A White, Hendrik Szurmant, James A Hoch, and Terence Hwa. Identification of direct residue contacts in protein–protein interaction by message

passing. *Proceedings of the National Academy of Sciences*, 106(1):67–72, January 2009. ISSN 00278424. doi: 10.1073/pnas.0805923106. URL https://www.pnas.org/content/106/1/67https://www.pnas.org/content/106/1/67.abstract. Publisher: National Academy of Sciences.

[131] Faruck Morcos, Andrea Pagnani, Bryan Lunt, Arianna Bertolino, Debora S Marks, Chris Sander, Riccardo Zecchina, José N Onuchic, Terence Hwa, and Martin Weigt. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proceedings of the National Academy of Sciences*, 108(49):E1293–E1301, December 2011. ISSN 00278424. doi: 10.1073/pnas.1111471108. URL https://www.pnas.org/content/108/49/E1293. Publisher: National Academy of Sciences.

[132] Yang Liu, Perry Palmedo, Qing Ye, Bonnie Berger, and Jian Peng. Enhancing Evolutionary Couplings with Deep Convolutional Neural Networks. *Cell Systems*, 6(1):65–74, January 2018. ISSN 24054720. doi: 10.1016/j.cels.2017.11.014. URL https://pubmed.ncbi.nlm.nih.gov/29275173/. Publisher: Cell Press.

[133] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy DeGiovanni, Jose H. Pereira, Andria V. Rodrigues, Alberdina A. van Dijk, Ana C. Ebrecht, Diederik J. Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K. Rathinaswamy, Udit Dalwadi, Calvin K. Yip, John E. Burke, K. Christopher Garcia, Nick V. Grishin, Paul D. Adams, Randy J. Read, and David Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373 (6557):871–876, August 2021. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.abj8754. URL https://www.science.org/doi/10.1126/science.abj8754.

[134] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6):654–669.e3, June 2021. ISSN 2405-4712. doi: 10.1016/j.cels.2021.05.017. URL https://www.sciencedirect.com/science/article/pii/S2405471221002039.

[135] Michael Heinzinger, Ahmed Elnaggar, Yu Wang, Christian Dallago, Dmitrii Nechaev, Florian Matthes, and Burkhard Rost. Modeling the language of life – Deep Learning Protein Sequences. *bioRxiv*, page 614313, 2019. doi: 10.1101/614313. URL https://www.biorxiv.org/content/10.1101/614313v3. Publisher: Cold Spring Harbor Laboratory.

[136] Jesse Vig, Ali Madani, Lav R. Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. BERTology Meets Biology: Interpreting Attention in Protein Language Models, March 2021. URL http://arxiv.org/abs/2006.15222. arXiv:2006.15222 [cs, q-bio].

[137] Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. Transformer protein language models are unsupervised structure learners. In *International Conference on Learning Representations*, page 2020.12.15.422761. Cold Spring Harbor Laboratory, December 2021. doi: 10.1101/2020.12.15.422761.

[138] Ratul Chowdhury, Nazim Bouatta, Surojit Biswas, Christina Floristean, Anant Kharkar, Koushik Roy, Charlotte Rochereau, Gustaf Ahdritz, Joanna Zhang, George M. Church, Peter K. Sorger, and Mohammed AlQuraishi. Single-sequence protein structure prediction using a language model and deep learning. *Nature Biotechnology*, 40(11): 1617–1623, November 2022. ISSN 1546-1696. doi: 10.1038/s41587-022-01432-w. URL https://www.nature.com/articles/s41587-022-01432-w. Number: 11 Publisher: Nature Publishing Group.

[139] C. E. Shannon. Prediction and entropy of printed English. *The Bell System Technical Journal*, 30(1):50–64, January 1951. ISSN 0005-8580. doi: 10.1002/j.1538-7305.1951.tb01366.x. Conference Name: The Bell System Technical Journal.

[140] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[141] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL http://arxiv.org/abs/1810.04805.

[142] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. *CoRR*, abs/2005.14165:1877–1901, 2020. URL https://arxiv.org/abs/2005.14165. _eprint: 2005.14165.

[143] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned Language Models Are Zero-Shot Learners. page 46, 2022.

[144] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain of Thought Prompting Elicits Reasoning in Large Language Models, June 2022. URL http://arxiv.org/abs/2201.11903. arXiv:2201.11903 [cs].

[145] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling Language Modeling with Pathways, April 2022. URL http://arxiv.org/abs/2204.02311. arXiv:2204.02311 [cs].

[146] Martin Steinegger, Milot Mirdita, and Johannes Söding. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nature Methods*, 16 (7):603–606, 2019. ISSN 1548-7105. doi: 10.1038/s41592-019-0437-4. URL https://doi.org/10.1038/s41592-019-0437-4.

[147] Supratim Mukherjee, Dimitri Stamatis, Jon Bertsch, Galina Ovchinnikova, Jagadish Chandrabose Sundaramurthi, Janey Lee, Mahathi Kandimalla, I.-Min A. Chen, Nikos C. Kyrpides, and T. B. K. Reddy. Genomes OnLine Database (GOLD) v.8: overview and updates. *Nucleic Acids Research*, 49(D1):D723–D733, January 2021. ISSN 1362-4962. doi: 10.1093/nar/gkaa983.

[148] Kathryn Tunyasuvunakool, Jonas Adler, Zachary Wu, Tim Green, Michal Zielinski, Augustin Žídek, Alex Bridgland, Andrew Cowie, Clemens Meyer, Agata Laydon, et al. Highly accurate protein structure prediction for the human proteome. *Nature*, 596 (7873):590–596, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03828-1. URL https://www.nature.com/articles/s41586-021-03828-1. Number: 7873 Publisher: Nature Publishing Group.

[149] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. Alphafold protein structure database: Massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 2021.

[150] Osamu Shimomura, Frank H. Johnson, and Yo Saiga. Extraction, purification and properties of aequorin, a bioluminescent protein from the luminous hydromedusan, aequorea. *Journal of Cellular and Comparative Physiology*, 59(3):223–239, 1962. doi: https://doi.org/10.1002/jcp.1030590302. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcp.1030590302.

[151] K. Mullis, F. Faloona, S. Scharf, R. Saiki, G. Horn, and H. Erlich. Specific enzymatic amplification of DNA in vitro: the polymerase chain reaction. *Cold Spring Harbor Symposia on Quantitative Biology*, 51 Pt 1:263–273, 1986. ISSN 0091-7451. doi: 10.1101/sqb.1986.051.01.032.

[152] Martin Jinek, Krzysztof Chylinski, Ines Fonfara, Michael Hauer, Jennifer A. Doudna, and Emmanuelle Charpentier. A programmable dual-RNA-guided DNA endonuclease in adaptive bacterial immunity. *Science (New York, N.Y.)*, 337(6096):816–821, August 2012. ISSN 1095-9203. doi: 10.1126/science.1225829.

[153] Stephen K Burley, Helen M Berman, Charmi Bhikadiya, Chunxiao Bi, Li Chen, Luigi Di Costanzo, Cole Christie, Ken Dalenberg, Jose M Duarte, Shuchismita Dutta, Zukang Feng, Sutapa Ghosh, David S Goodsell, Rachel K Green, Vladimir Guranoví, Dmytro Guzenko, Brian P Hudson, Tara Kalro, Yuhe Liang, Robert Lowe, Harry Namkoong, Ezra Peisach, Irina Periskova, Andreas Prlí, Chris Randle, Alexander Rose, Peter Rose, Raul Sala, Monica Sekharan, Chenghua Shao, Lihua Tan, Yi-Ping Tao, Yana Valasatava, Maria Voigt, John Westbrook, Jesse Woo, Huanwang Yang, Jasmine Young, Marina Zhuravleva, and Christine Zardecki. RCSB Protein Data Bank: biological macromolecular structures enabling research and education in fundamental biology, biomedicine, biotechnology and energy. *Nucleic Acids Research*, 47, 2019. doi: 10.1093/nar/gky1004. URL https://academic.oup.com/nar/article-abstract/47/D1/D464/5144139.

[154] Jürgen Haas, Alessandro Barbato, Dario Behringer, Gabriel Studer, Steven Roth, Martino Bertoni, Khaled Mostaguir, Rafal Gumienny, and Torsten Schwede. Continuous Automated Model EvaluatiOn (CAMEO) complementing the critical assessment of structure prediction in CASP12. *Proteins: Structure, Function and Bioinformatics*, 86 (Suppl 1):387–398, March 2018. ISSN 10970134. doi: 10.1002/prot.25431. Publisher: John Wiley and Sons Inc.

[155] 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. ISSN 1097-0134. doi: 10.1002/prot.26237. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.26237. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.26237.

[156] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins*, 57(4):702–710, December 2004. ISSN 1097-0134. doi: 10.1002/prot.20264.

[157] Gustaf Ahdritz, Nazim Bouatta, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, Arkadiusz Nowaczynski, Bei Wang, Marta M Stepniewska-Dziubinska, Shang Zhang, Adegoke Ojewole, Murat Efe Guney, Stella Biderman, Andrew M Watkins, Stephen Ra,

Pablo Ribalta Lorenzo, Lucas Nivon, Brian Weitzner, Yih-En Andrew Ban, Peter K Sorger, Emad Mostaque, Zhao Zhang, Richard Bonneau, and Mohammed AlQuraishi. Openfold: Retraining alphafold2 yields new insights into its learning mechanisms and capacity for generalization. *bioRxiv*, 2022. doi: 10.1101/2022.11.20.517210. URL https://www.biorxiv.org/content/10.1101/2022.11.20.517210.

[158] J Dauparas, I Anishchenko, N Bennett, H Bai, R J Ragotte, L F Milles, B I M Wicky, A Courbet, R J de Haas, N Bethel, P J Y Leung, T F Huddy, S Pellock, D Tischer, F Chan, B Koepnick, H Nguyen, A Kang, B Sankaran, A K Bera, N P King, and D Baker. Robust deep learning-based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, October 2022. doi: 10.1126/science.add2187. URL https://www.science.org/doi/10.1126/science.add2187. Publisher: American Association for the Advancement of Science.

[159] Jue Wang, Sidney Lisanza, David Juergens, Doug Tischer, Joseph L. Watson, Karla M. Castro, Robert Ragotte, Amijai Saragovi, Lukas F. Milles, Minkyung Baek, Ivan Anishchenko, Wei Yang, Derrick R. Hicks, Marc Expòsit, Thomas Schlichthaerle, Jung-Ho Chun, Justas Dauparas, Nathaniel Bennett, Basile I. M. Wicky, Andrew Muenks, Frank DiMaio, Bruno Correia, Sergey Ovchinnikov, and David Baker. Scaffolding protein functional sites using deep learning. *Science*, 377(6604):387–394, July 2022. doi: 10.1126/science.abn2100. URL https://www.science.org/doi/abs/10.1126/science.abn2100. Publisher: American Association for the Advancement of Science.

[160] B. I. M. Wicky, L. F. Milles, A. Courbet, R. J. Ragotte, J. Dauparas, E. Kinfu, S. Tipps, R. D. Kibler, M. Baek, F. DiMaio, X. Li, L. Carter, A. Kang, H. Nguyen, A. K. Bera, and D. Baker. Hallucinating protein assemblies, June 2022. URL https://www.biorxiv.org/content/10.1101/2022.06.09.493773v1. Pages: 2022.06.09.493773 Section: New Results.

[161] Simon C Potter, Aurélien Luciani, Sean R Eddy, Youngmi Park, Rodrigo Lopez, and Robert D Finn. HMMER web server: 2018 update. *Nucleic Acids Research*, 46(Web Server issue):W200–W204, July 2018. ISSN 0305-1048. doi: 10.1093/nar/gky448. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6030962/.

[162] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, November 2017. ISSN 1546-1696. doi: 10.1038/nbt.3988. URL https://www.nature.com/articles/nbt.3988. Number: 11 Publisher: Nature Publishing Group.

[163] Yang Zhang. Protein structure prediction: when is it useful? *Current Opinion in Structural Biology*, 19(2):145–155, April 2009. ISSN 1879-033X. doi: 10.1016/j.sbi.2009.02.005.

[164] Cyrus Levinthal. Are there pathways for protein folding? *Extrait du Journal de Chimie Physique*, 65(1):44, 1968.

[165] A M Lesk and G D Rose. Folding units in globular proteins. *Proc. Natl. Acad. Sci. U. S. A.*, 78(7):4304–4308, July 1981.

[166] William R. Taylor. A 'periodic table' for protein structures. *Nature*, 416(6881):657–660, April 2002. ISSN 0028-0836. doi: 10.1038/416657a.

[167] Craig O Mackenzie, Jianfu Zhou, and Gevorg Grigoryan. Tertiary alphabet for the observable protein structural universe. *Proc. Natl. Acad. Sci. U. S. A.*, 113(47):E7438–E7447, November 2016.

[168] W Kabsch and C Sander. On the use of sequence homologies to predict protein structure: Identical pentapeptides can have completely different conformations. *Proc. Natl. Acad. Sci. U. S. A.*, 81(4):1075–1078, February 1984.

[169] D L Minor, Jr and P S Kim. Context-dependent secondary structure formation of a designed protein sequence. *Nature*, 380(6576):730–734, April 1996.

[170] Che Yang, Fabian Sesterhenn, Jaume Bonet, Eva A van Aalen, Leo Scheller, Luciano A Abriata, Johannes T Cramer, Xiaolin Wen, Stéphane Rosset, Sandrine Georgeon, Theodore Jardetzky, Thomas Krey, Martin Fussenegger, Maarten Merkx, and Bruno E Correia. Bottom-up de novo design of functional proteins with complex structural features. *Nat. Chem. Biol.*, 17(4):492–500, April 2021.

[171] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic level protein structure with a language model. *bioRxiv*, July 2022. doi: 10.1101/2022.07.20.500902. URL https://www.biorxiv.org/content/10.1101/2022.07.20.500902v1. Pages: 2022.07.20.500902 Section: New Results.

[172] Georges Vauquelin and Steven J Charlton. Exploring avidity: understanding the potential gains in functional affinity and target residence time of bivalent and heterobivalent ligands. *Br. J. Pharmacol.*, 168(8):1771–1785, April 2013.

[173] William P. Russ, Matteo Figliuzzi, Christian Stocker, Pierre Barrat-Charlaix, Michael Socolich, Peter Kast, Donald Hilvert, Remi Monasson, Simona Cocco, Martin Weigt, and Rama Ranganathan. An evolution-based model for designing chorismate mutase enzymes. *Science*, 369(6502):440–445, July 2020. ISSN 10959203. doi: 10.1126/science.aba3304. Publisher: American Association for the Advancement of Science.

[174] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nat. Commun.*, 13(1):4348, July 2022.

[175] John Ingraham, Max Baranov, Zak Costello, Vincent Frappier, Ahmed Ismail, Shan Tie, Wujie Wang, Vincent Xue, Fritz Obermeyer, Andrew Beam, and Gevorg Grigoryan. Illuminating protein space with a programmable generative model. *bioRxiv*, 2022. doi: 10.1101/2022.12.01.518682.

[176] Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. *bioRxiv*, 2022. doi: 10.1101/2022.12.09.519842.

[177] Zaixiang Zheng, Yifan Deng, Dongyu Xue, Yi Zhou, Fei Ye, and Quanquan Gu. Structure-informed language models are protein designers. *bioRxiv*, pages 2023–02, 2023.

[178] Yitian Fang, Yi Jiang, Leyi Wei, Qin Ma, Zhixiang Ren, Qianmu Yuan, and Dong-Qing Wei. Deepprosite: structure-aware protein binding site prediction using esmfold and pretrained language model. *Bioinformatics*, 39(12):btad718, 2023.

[179] Wengong Jin, Caroline Uhler, and Nir Hacohen. Se (3) denoising score matching for unsupervised binding energy prediction and nanobody design. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.

[180] Luise Dunsche, Nikita Ivanisenko, Shamala Riemann, Sebastian Schindler, Stefan Beissert, Cristian Angeli, Stephanie Kreis, Mahvash Tavassoli, Inna Lavrik, and Dagmar Kulms. A cytosolic mut p53 (e285k) variant confers chemoresistance of malignant melanoma. *Cell Death & Disease*, 14(12):831, 2023.

[181] Alvaro Martin Hermosilla, Carolin Berner, Sergey Ovchinnikov, and Anastassia A Vorobieva. Validation of de novo designed water-soluble and transmembrane proteins by in silico folding and melting. *bioRxiv*, pages 2023–06, 2023.

[182] Mary E. Petrone, Joe Grove, Jonathon C. O. Mifsud, Rhys H. Parry, Ezequiel M. Marzinelli, and Edward C. Holmes. A 39.8kb flavi-like virus uses a novel strategy for overcoming the rna virus error threshold. *bioRxiv*, 2024. doi: 10.1101/2024.01.08.574764. URL https://www.biorxiv.org/content/early/2024/01/09/2024.01.08.574764.

[183] Joakim Nøddeskov Clifford, Magnus Haraldson Høie, Sebastian Deleuran, Bjoern Peters, Morten Nielsen, and Paolo Marcatili. Bepipred-3.0: Improved b-cell epitope prediction using protein language models. *Protein Science*, 31(12):e4497, 2022.

[184] Yanay Rosen, Maria Brbić, Yusuf Roohani, Kyle Swanson, Ziang Li, and Jure Leskovec. Towards universal cell embeddings: Integrating single-cell rna-seq datasets across species with saturn. *bioRxiv*, 2023.

[185] Kiarash Jamali, Lukas Käll, Rui Zhang, Alan Brown, Dari Kimanius, and Sjors HW Scheres. Automated model building and protein identification in cryo-em maps. *bioRxiv*, 2023.

[186] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv:1701.05517*, 2017.

[187] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR arXiv:1512.03385*, 2015.

[188] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing. *arXiv:1903.10145 [cs, stat]*, June 2019. URL http://arxiv.org/abs/1903.10145. arXiv: 1903.10145.

[189] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[190] Partha Ghosh, Mehdi SM Sajjadi, Antonio Vergari, Michael Black, and Bernhard Schölkopf. From variational to deterministic autoencoders. *arXiv:1903.12436*, 2019.

[191] Sarel J. Fleishman, Andrew Leaver-Fay, Jacob E. Corn, Eva-Maria Strauch, Sagar D. Khare, Nobuyasu Koga, Justin Ashworth, Paul Murphy, Florian Richter, Gordon Lemmon, Jens Meiler, and David Baker. RosettaScripts: A Scripting Language Interface to the Rosetta Macromolecular Modeling Suite. *PLOS ONE*, 6(6):e20161, June 2011. ISSN 1932-6203. doi: 10.1371/journal.pone.0020161. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0020161.

[192] Simon C Potter, Aurélien Luciani, Sean R Eddy, Youngmi Park, Rodrigo Lopez, and Robert D Finn. Hmmer web server: 2018 update. *Nucleic acids research*, 46(W1):W200–W204, 2018.

[193] Martin Steinegger, Markus Meier, Milot Mirdita, Harald Vöhringer, Stephan J. Haunsberger, and Johannes Söding. HH-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics*, 20(1):473, September 2019. ISSN 14712105. doi: 10.1186/s12859-019-3019-7. URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3019-7. Publisher: BioMed Central Ltd.

[194] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*, 2019.

[195] Milot Mirdita, Lars von den Driesch, Clovis Galiez, Maria J Martin, Johannes Söding, and Martin Steinegger. Uniclust databases of clustered and deeply annotated protein sequences and alignments. *Nucleic acids research*, 45(D1):D170–D176, January 2017. ISSN 13624962. doi: 10.1093/nar/gkw1081. Publisher: Oxford University Press.

[196] S. D. Dunn, L. M. Wahl, and G. B. Gloor. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. *Bioinformatics*, 24(3):333–340, February 2008. ISSN 13674803. doi: 10.1093/bioinformatics/btm604. Publisher: Oxford Academic.

[197] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. ISSN 1533-7928. URL http://scikit-learn.sourceforge.net.

[198] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding, October 2021. URL http://arxiv.org/abs/2104.09864. arXiv:2104.09864 [cs] version: 2.

[199] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large Batch Optimization for Deep Learning: Training BERT in 76 Minutes. page 38, 2020.

[200] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. ZeRO: Memory optimizations Toward Training Trillion Parameter Models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16, November 2020. doi: 10.1109/SC41405.2020.00024.

[201] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial Attention in Multidimensional Transformers. *arXiv*, December 2019. URL http://arxiv.org/abs/1912.12180. Publisher: arXiv.

[202] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):15, February 2018. ISSN 1474-760X. doi: 10.1186/s13059-017-1382-0. URL https://doi.org/10.1186/s13059-017-1382-0.

[203] Isaac Virshup, Sergei Rybakov, Fabian J. Theis, Philipp Angerer, and F. Alexander Wolf. anndata: Annotated data, December 2021. URL https://www.biorxiv.org/content/10.1101/2021.12.16.473007v1. Pages: 2021.12.16.473007 Section: New Results.

[204] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, September 2020. URL http://arxiv.org/abs/1802.03426. arXiv:1802.03426 [cs, stat].

[205] A Shrake and J A Rupley. Environment and exposure to solvent of protein atoms. lysozyme and insulin. *J. Mol. Biol.*, 79(2):351–371, September 1973.

[206] G Labesse, N Colloc'h, J Pothier, and J P Mornon. P-SEA: a new efficient assignment of secondary structure from C alpha trace of proteins. *Comput. Appl. Biosci.*, 13(3): 291–295, June 1997.

[207] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, 1976. doi: https://doi.org/10.1107/S0567739476001873. URL https://onlinelibrary.wiley.com/doi/abs/10.1107/S0567739476001873.

[208] Sung Il Yoon, Brandi C Jones, Naomi J Logsdon, and Mark R Walter. Same structure, different function crystal structure of the Epstein-Barr virus IL-10 bound to the soluble IL-10R1 chain. *Structure*, 13(4):551–564, April 2005.

[209] G Szakonyi, J M Guthridge, D Li, K Young, V M Holers, and X S Chen. Structure of complement receptor 2 in complex with its C3d ligand. *Science*, 292(5522):1725–1728, June 2001.

[210] Nicole L Kallewaard, Davide Corti, Patrick J Collins, Ursula Neu, Josephine M McAuliffe, Ebony Benjamin, Leslie Wachter-Rosati, Frances J Palmer-Hill, Andy Q Yuan, Philip A Walker, Matthias K Vorlaender, Siro Bianchi, Barbara Guarino, Anna De Marco, Fabrizia Vanzetta, Gloria Agatic, Mathilde Foglierini, Debora Pinna, Blanca Fernandez-Rodriguez, Alexander Fruehwirth, Chiara Silacci, Roksana W Ogrodowicz, Stephen R Martin, Federica Sallusto, Joann A Suzich, Antonio Lanzavecchia, Qing Zhu, Steven J Gamblin, and John J Skehel. Structure and function analysis of an antibody recognizing all influenza a subtypes. *Cell*, 166(3):596–608, July 2016.

[211] Kathryn Westendorf, Stefanie Žentelis, Lingshu Wang, Denisa Foster, Peter Vaillancourt, Matthew Wiggin, Erica Lovett, Robin van der Lee, Jörg Hendle, Anna Pustilnik, J Michael Sauder, Lucas Kraft, Yuri Hwang, Robert W Siegel, Jinbiao Chen, Beverly A Heinz, Richard E Higgs, Nicole L Kallewaard, Kevin Jepson, Rodrigo Goya, Maia A Smith, David W Collins, Davide Pellacani, Ping Xiang, Valentine de Puyraimond, Marketa Ricicova, Lindsay Devorkin, Caitlin Pritchard, Aoise O'Neill, Kush Dalal, Pankaj Panwar, Harveer Dhupar, Fabian A Garces, Courtney A Cohen, John M Dye, Kathleen E Huie, Catherine V Badger, Darwyn Kobasa, Jonathan Audet, Joshua J Freitas, Saleema Hassanali, Ina Hughes, Luis Munoz, Holly C Palma, Bharathi Ramamurthy, Robert W Cross, Thomas W Geisbert, Vineet Menachery, Kumari Lokugamage, Viktoriya Borisevich, Iliana Lanz, Lisa Anderson, Payal Sipahimalani, Kizzmekia S Corbett, Eun Sung Yang, Yi Zhang, Wei Shi, Tongqing Zhou, Misook Choe, John Misasi, Peter D Kwong, Nancy J Sullivan, Barney S Graham, Tara L Fernandez, Carl L Hansen, Ester Falconer, John R Mascola, Bryan E Jones, and Bryan C Barnhart. LY-CoV1404 (bebtelovimab) potently neutralizes SARS-CoV-2 variants. *Cell Rep.*, 39 (7):110812, May 2022.