Rotamer-Free Protein Sequence Design Based on Deep Learning and Self-Consistency

Yufeng Liu
University of Science and Technology of China
Lu Zhang
University of Science and Technology of China
Weilun Wang
University of Science and Technology of China
Min Zhu
University of Science and Technology of China
Chenchen Wang
University of Science and Technology of China
Fudong Li
University of Science and Technology of China
Jiahai Zhang
University of Science and Technology of China
Houqiang Li
University of Science and Technology of China
Quan Chen
University of Science and Technology of China https://orcid.org/0000-0002-3301-3065
Haiyan Liu (hyliu@ustc.edu.cn)
University of Science and Technology of China

Article

Keywords:

Posted Date: February 1st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1209166/v1

Version of Record: A version of this preprint was published at Nature Computational Science on July 21st, 2022. See the published version at https://doi.org/10.1038/s43588-022-00273-6.

Rotamer-Free Protein Sequence Design Based on Deep Learning and Self-Consistency

Yufeng Liu[1,†], Lu Zhang[1,†], Weilun Wang[2,†], Min Zhu[1], Chenchen Wang[1], Fudong Li[1,3], Jiahai Zhang[1,3], Houqiang Li[2,*], Quan Chen[1,3,*], and Haiyan Liu[1,3,4,*]

[1]MOE Key Laboratory for Membraneless Organelles and Cellular Dynamics, School of Life Sciences, Division of Life Sciences and Medicine, University of Science and Technology of China, Hefei, Anhui 230027, China.

[2]CAS Key Laboratory of GIPAS, School of Information Science and Technology, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, Anhui 230027, China.

[3]Biomedical Sciences and Health Laboratory of Anhui Province, University of Science and Technology of China, Hefei, Anhui 230027, China.

[4]School of Data Science, University of Science and Technology of China, Hefei, Anhui 230027, China.

[†]These authors contributed equally to the work.

[*]To whom correspondence should be addressed; e-mails:lihq@ustc.edu.cn, chenquan@ustc.edu.cn, hyliu@ustc.edu.cn.

Abstract

We present ABACUS-R, a method based on deep learning for designing amino acid sequences that autonomously fold into a given target backbone. This method predicts the sidechain type of a central residue from its 3D local environment by using an encoder-decoder network trained with a multi-task learning strategy. The environmental features encoded by the network include the types but not the conformations of the sidechains of surrounding residues. This eliminates the needs for reconstructing and optimizing sidechain structures, and drastically simplifies the sequence design process. Thus iteratively applying the encoder-decoder to different central residues is able to produce self-consistent overall sequences for a target backbone. Extensive results of wet experiments, including five structures solved by X-ray crystallography, show that ABACUS-R outperforms state-of-the-art energy function-based *de novo* sequence design methods by significant margins in success rate and design precision. ABACUS-R constitutes a robust tool for wide applications in protein design and protein engineering.

1 Introduction

Computational protein design brings the capability of inventing *de novo* proteins [1, 2] to fulfill various structural and functional needs from therapeutics [3, 4] to bio-catalysis [5, 6]. One of the general problems to be solved in computational protein design is inverse protein folding, which is to select amino acid sequences that autonomously fold into a given backbone target. Although a few existing methods to solve this problem have been repeatedly verified with experimentally solved structures of designed proteins [7, 8, 9, 10], these methods still suffer from deficiencies including low success rates [11, 12], high sensitivity to target structures [13, 14], and overly monotonous designed sequences that lack the diversity and variability of natural amino acid sequences [10, 15]. To overcome these limitations calls for further method innovations [16, 17].

Conventional methods for inverse protein folding are based on optimizing empirical energy functions with respect to the sidechain types (and the sidechain conformations [18]). The energy functions are either physics-based (*e.g.* RosettaDesign [8], Proteus [19] and EvoEF2 [20]) or statistically learned from data (*e.g.* ABACUS [9, 10] and TERM [16]). Invariably, these energy functions employ the approximation of treating complicated, many-body molecular interactions as linear combinations of one-body and two-body terms [8, 9, 10, 18]. This approximation has been a fundamental accuracy-limiting factor despite continuous improvements [21] of conventional energy functions.

Previously, we developed a statistical energy model named ABACUS [9, 10] (a backbone-based amino acid usage survey). Proteins designed with ABACUS have been verified with experimentally-solved structures in a number of studies [9, 10, 22, 23, 24]. Although the energy terms in ABACUS were devised to consider high-order coupling be-

tween various physical factors, the non-linear integration of coupled effects was restricted to the single-residue and the residue-pair-wise levels, beyond which separately learned energy terms were combined only in a linear way. Replacing this linear combination with an approach that can integrate non-linear coupling at higher orders may significantly increase the robustness and accuracy of this data-driven method.

A suitable approach that does not depend on the linear combination approximation is deep learning [25]. Deep learning has already tremendously advanced protein structure prediction (*i.e.*, forward folding) [26, 27, 28]. Its application in structure-based sequence design (*i.e.*, inverse folding) were explored in several recent studies [17, 29, 30, 31, 32, 33, 34]. Although promising methods have been demonstrated to outperform conventional energy function-based approaches in computational tests, in tests by wet experiments, deep learning methods have not yet exhibited performances comparable to established energy function-based methods (for a recent review, see ref [35]). Only until recently, a study using a deep learning method has reported experimentally-solved structures for two sequences designed for an ideal TIM-barrel backbone [17]. Thus inverse protein folding by deep learning still needs significant improvements to have real impacts on computational protein design.

Nevertheless, we anticipate a well-developed deep learning method to significantly surpass conventional statistical energy models such as ABACUS in both *in silico* tests and wet experiments, because a deep learning-based method can retain features that have been proven to work while eliminating known problems. In the current study, we have moved along this direction by developing ABACUS-R, a model that uses deep learning (without using the linear combination approximation) to establish a vector representation that encodes the 3D local environment of a focused central residue. From this representation, a range of attributes of the central residue, including its sidechain type, can be decoded. To design an overall sequence for a given target backbone, we apply the pre-trained encoder-decoder to different residues of the backbone iteratively to obtain closed, self-consistent, decoding solutions. We have evaluated ABACUS-R by using both *in silico* metrics and wet experiments.

2 Results

2.1 Overview of the model

The encoder part of the encoder-decoder network of ABACUS-R (see Figure 1A) is a Transformer [36] whose input comprises the sidechain types and 3D backbone structure information of all the structurally neighboring residues of a central residue. We emphasize that the sidechain type of the central residue is not used as input. Neither are the sidechain conformations of the neighboring residues. The encoder is invariant with respect to both the translation and rotation in 3D and the permutation of the order of the neighboring residues. The encoder's output constitutes the desired vector representation that is to be decoded into various attributes of the central residue. This encoder-decoder

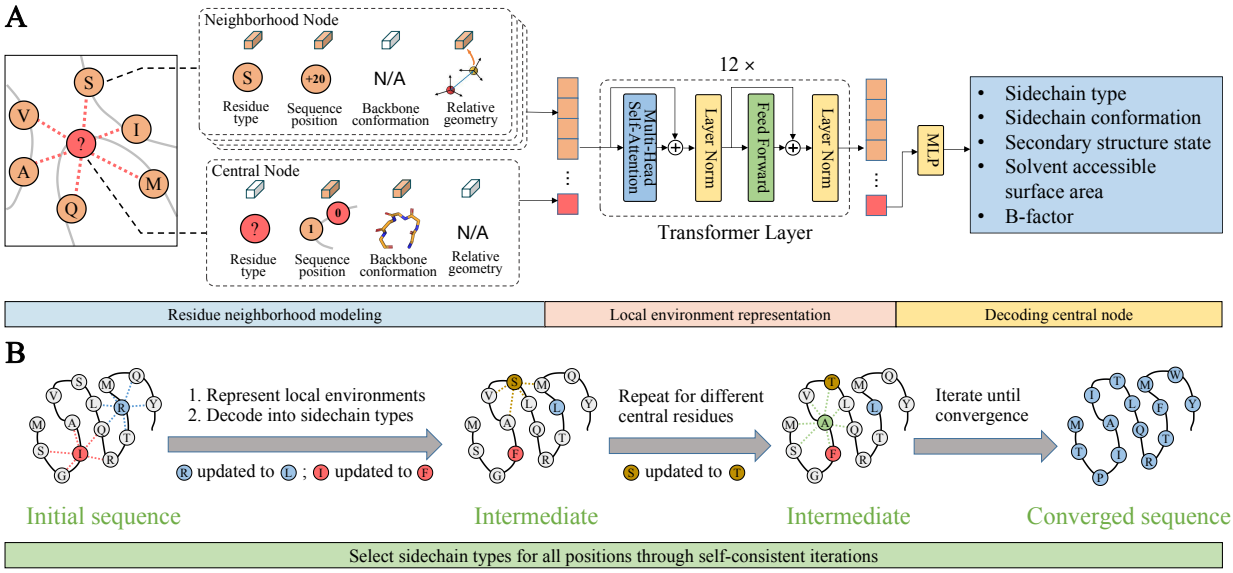

Figure 1. An overview of the ABACUS-R method. **A**, The framework of the encoder-decoder model. The input of the encoder includes backbone-only structure features and the sidechain types of surrounding residues (Residue neighborhood modeling). The encoder is a Transformer of 12 blocks, the output of which constitutes a general, integrated representation of the input features (Local environment representation). To benefit from multi-task learning, this representation is decoded into several different attributes (including the sidechain type) of the central residue (Decoding central node). **B**, The iterative approach for designing self-consistent overall sequences. In each iteration, the encoder-decoder network is applied to each residue in a randomly chosen subset of residues to update its sidechain type according to its current, sequence-dependent local environment.

network has been trained on a selected set of PDB [37] structures.

The self-consistent iterations for designing the overall sequences for a given target backbone are illustrated in Figure 1B. The process starts from an initial sequence which can be chosen randomly; then for one or more randomly chosen central residues, their sidechain types are updated by applying the pre-trained encoder-decoder network to their local environments (which depend on the sidechain types of the surrounding residues in the current sequence); this operation is repeated for different central residues in successive iterations until the sidechain types no longer change for as many residues as possible (*i.e.*, have converged).

2.2 Accuracy of the encoder-decoder network

A set of non-redundant PDB structures have been used to train the encoder-decoder network (see Method). We have learned two groups of network parameters by splitting the selected PDB structures into training and test sets in two different ways. The first group of network parameters ($\text{Model}_{\text{eval}}$) have been learnt by using about 95% of the protein structures for training and the remaining 5% for testing, with the structures for testing belonging to single-domain topology classes (according to the CATH 4.2 classification of protein structures [38]) in the dataset of the selected structures. With this choice of test proteins, none of the test structures belonged to the same CATH topology as a training structure. Thus $\text{Model}_{\text{eval}}$ can be used for unbiased computational evaluations. The second group of network

A

Settings					Results						
AAtype	SS3+SS8	SASA	B-factor	χ_1+ χ_2	Accuracy(%)	SS3(%)	SS8(%)	SASA	B-factor	χ_1	χ_2
√					43.0	-	-	-	-	-	-
√	√				47.4	97.0	94.7	-	-	-	-
√	√	√			48.8	97.3	95.2	0.899	-	-	-
√	√	√	√		48.9	97.3	95.2	0.899	0.541	-	-
√	√	√	√	√	**49.3**	**97.3**	**95.2**	**0.900**	**0.540**	**0.754**	**0.586**

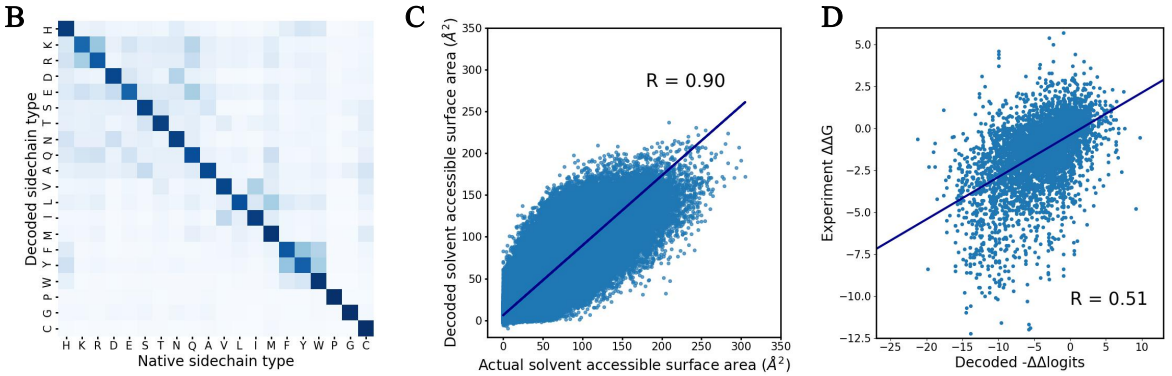

Figure 2. Performance of $\text{Model}_{\text{eval}}$ in computational tests. **A**, Accuracy of the encoder-decoder. The "✓" symbols in the "Settings" columns mark which attributes of the central residue have been included as the decoding objectives in multi-task learning; the "Results" columns give the decoding accuracies for various attributes of the central residue ("Accuracy" for sidechain type; "SS3" and "SS8" for secondary structure state categorized according to the three-state scheme and the eight-state scheme, respectively; and χ_1 and χ_2 for sidechain torsional angles with the accuracies computed by tolerating decoding errors of less than 40°). **B**, The confusion matrix of decoding the sidechain type of the central residue. Darker cells indicate that higher fractions of residues of the corresponding native sidechain types (the horizontal axis) have been decoded into the corresponding decoded sidechain types (the vertical axis). **C**, Scattering plot of the actual versus the decoded solvent accessible surface areas of the central residues. **D**, Scattering plot of experimentally measured $\Delta\Delta$G versus $-\Delta\Delta$logits from the decoder for the dataset of protein mutants. R is the Pearson correlation coefficient in B and C.

parameters ($\text{Model}_{\text{final}}$) have been learnt by randomly splitting the selected protein structures into roughly 95% for training and 5% for testing, disregarding their CATH structure classification. $\text{Model}_{\text{final}}$ has been used to design the experimentally examined sequences.

The encoder has been trained by using a multi-task learning strategy, in which multiple attributes of the central residue have been simultaneously considered as decoding objectives, including the sidechain type, the secondary structure state (the three-state and the eight-state schemes of categorizing the secondary structure type [39] have been considered simultaneously), the solvent accessible surface area (SASA), the sidechain torsional angles (*i.e.* χ_1 and χ_2), and the crystallographic B factor of the C_α atom.

The decoding accuracies of $\text{Model}_{\text{eval}}$ for the different attributes are reported in Figures 2A-C and Supplementary Figure 1. Similar results for $\text{Model}_{\text{final}}$ are reported in Supplementary Table 1. Briefly, both $\text{Model}_{\text{eval}}$ and $\text{Model}_{\text{final}}$ are able to recover the sidechain type of the central residue with an accuracy of around 50%. Moreover, for a significant

fraction of the decoding results that did not recover the exact native sidechain types, the decoder produces sidechain types physicochemically similar to the native types (*e.g.*, lysine for arginine, phenylalanine for tyrosine, *etc.*, see Figure 2B). The original and the decoded SASAs are compared in Figure 2C; similar comparisons for the other attributes are shown in Supplementary Figure 1. It is interesting to note that by adding the decoding accuracies for attributes other than the sidechain type to the overall loss in network training, the accuracies for decoding the sidechain type have been significantly improved (Figure 2A, from 45.3% to 49.3% for $\text{Model}_{\text{eval}}$, and Supplementary Table 1, from 47.2% to 53.0% for $\text{Model}_{\text{final}}$).

For the categorical sidechain type attribute, the actual output of the decoder is a vector of "$logits$" values which are transformed into (normalized) "probabilities" with the Softmax function (see Method). To examine if the negatives of the $logits$ values can be interpreted as effective "energies" of different residue types, we chose a dataset of protein mutants with experimentally measured changes of protein stability [40], and examined the Pearson correlation coefficient between the stability changes (as measured by $\Delta\Delta$G) of different mutants with the changes in $-\Delta\Delta logits$ (see Method). The resulting value of 0.51 (see Figure 2D) is comparable to that reported for a deep learning network specifically trained to reproduce the effects of mutations on protein stability [40]. This observation supports the use of the "$-logits$" value as a surrogate of the effective "energy" of a sidechain type in a specific local 3D environment.

2.3 Convergence of the sequence design iterations

We have applied $\text{Model}_{\text{eval}}$ and self-consistent iterations to design overall sequences for 100 target structures taken from the test set of $\text{Model}_{\text{eval}}$. These target structures cover three main CATH classes (Supplementary Table 2). For each target structure, 10 sequences have been designed using 10 different runs, each run starting from a different random initial sequence. As the iterative approach is effectively a greedy algorithm to maximize the (predicted) probabilities of the sidechain types, we monitored the evolution of the negative logorithms of the probabilities of the designed sidechain types (the $-logP$ values) during the iterative runs. Figure 3A shows how the averaged per residue $-logP$ value has decreased and converged to a plateau value. In the meanwhile, the sidechain types of most residues were converging towards the corresponding types in the final sequences (Figure 3B). For all target structures, the iterative runs can produce self-consistent sequences that are converged either completely or up to having only a very small number of fluctuating positions (see the inset of Figure 3B and Supplementary Figure 2A).

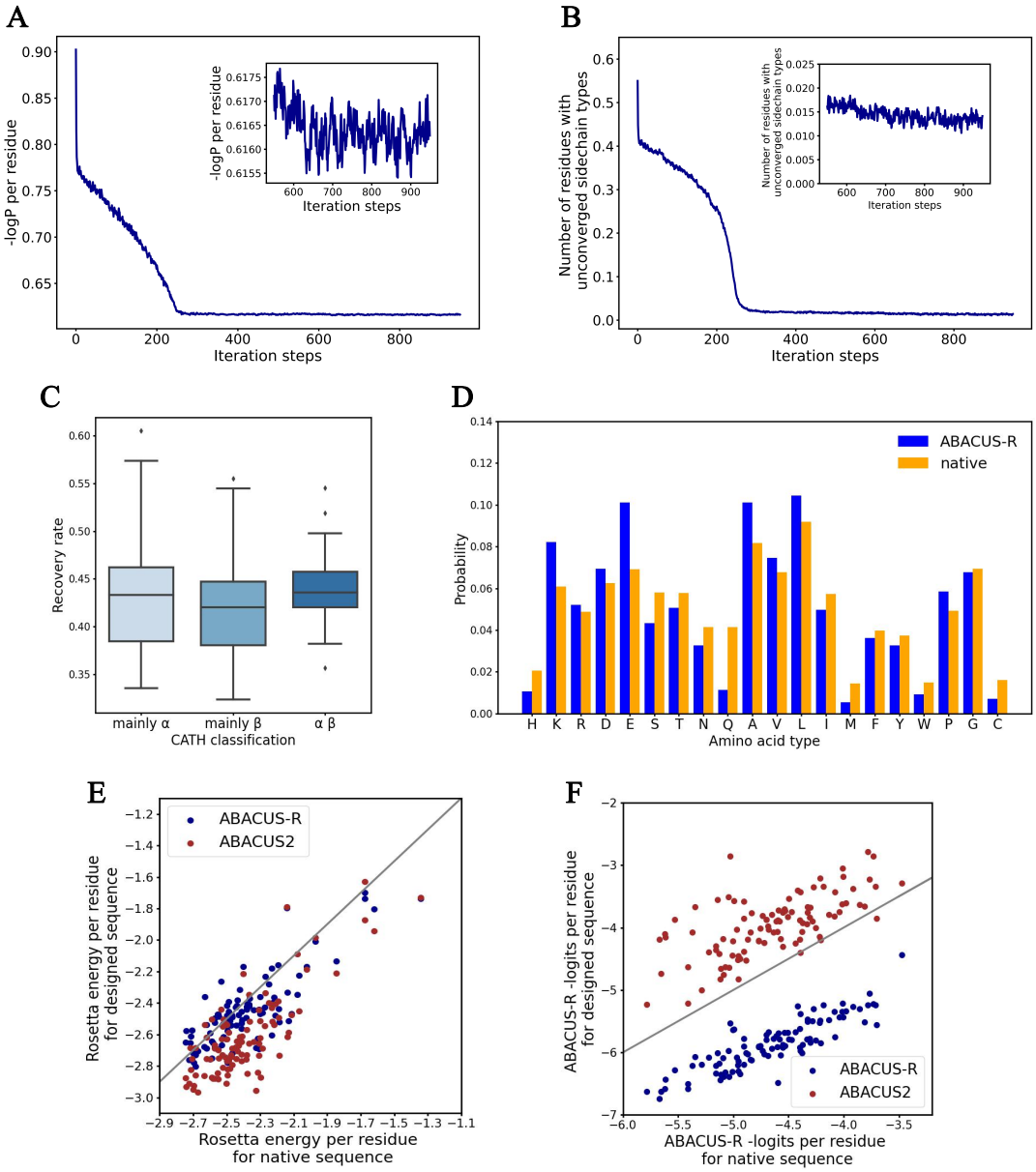

Figure 3. Results of overall sequence design for natural backbones. The results have been obtained with $\text{Model}_{\text{eval}}$ and are averages over 100 target backbones. **A**, Evolution of $-logP$ per residue during the self-consistent iterations. **B**, The evolution of the number of residues with unconverged sidechain types (*i.e.*, the sidechain types are different from those in the final sequences) during the self-consistent iterations. **C**, The rates of recovering the native sidechain types for backbone targets in three CATH architecture classes. **D**, The distributions of sidechain types in the designed sequences and in the native sequences. **E**, Scattering plot of the Rosetta energies per residue of the designed sequences versus the Rosetta energies per residue of corresponding native sequences. Blue (red) points correspond to sequences designed by ABACUS-R (ABACUS2). A grey line is drawn along the diagonal. **F**, Scattering plot of the ABACUS-R $-logits$ values per residue of the designed sequences versus the ABACUS-R $-logits$ values of corresponding native sequences. Blue (red) points are for sequences designed by ABACUS-R (ABACUS2). A grey line is drawn along the diagonal.

If not all positions can fully converge in a sequence design run, the sequence of the lowest total $-logP$ recorded during the run is taken as the final design result. This choice is for convenience and not critical, because both the number of unconverged positions and the fluctuations of $-logP$ are very small between the sequences recorded during the plateau stage (see the inset of Figures 3A and 3B), suggesting all these sequences to be equally acceptable. Nevertheless, we inspected the unconverged residues in several target structures (results not shown), and found that they were usually groups of two to three residues interacting with each other with few interactions outside of the corresponding groups. In the final iterations, the decoded sidechain type combinations of such a group oscillated between a few possibilities instead of converging to a single combination, indicating that optimum sidechain types cannot be selected simultaneously for all the residues in such a group.

For the same target structure, the runs starting from different random initial sequences usually lead to not exactly the same but still highly similar sequences. For the 100 targets considered here, the sequence identities between the (almost completely converged) self-consistent sequences from different runs ranged from 0.76 to 0.89, with the per residue $-logits$ values varied only within a narrow range (*i.e.*, ± 0.50, which is smaller than the differences between the $-logits$ values of the corresponding designed and native sequences, see Method). Thus the sequences designed by different runs can be considered as equally plausible.

2.4 Designed sequences compared with native sequences

For the 100 target structures considered, the average identity between the designed sequences and corresponding native sequences is $43.1\% \pm 5.4\%$ (for different targets, this identity varied from 32% to 61%). While the native sidechain type recovery rate does not exhibit strong dependency on the CATH class of the overall structure (Figure 3C), this recovery rate has the expected tendency of decreasing with increasing solvent accessibility (see Supplementary Figure 2B). The amino acid compositions of the designed and the native sequences are highly similar: the Pearson correlation coefficient between the set of frequencies of the 20 sidechain types in the designed sequences and those in the native sequences is 0.93 (Figure 3D). Nevertheless, some sidechain types including glutamic acid, alanine, and lysine have been used more frequently in the designed than in the native sequences, while the sidechain types of obviously reduced usages in the designed sequences include glutamine, histidine, and methionine (Figure 3D).

The designed sequences and the native sequences have been further compared in their Rosetta per residue energies computed after backbone relaxation by Rosetta [8, 41]. For the same target backbone structure, the Rosetta per residue energies of the designed and the native sequences are similar (Figure 3E). Thus the designed sequences are as suitable as the native sequences for the corresponding target structures according to the independent metric of Rosetta energy.

Notwithstanding such similar Rosetta energies, the designed sequences are of substantially lower per residue $-logits$ values in comparison with the native sequences (Figure 3F). This is anticipated because the $-logits$ values

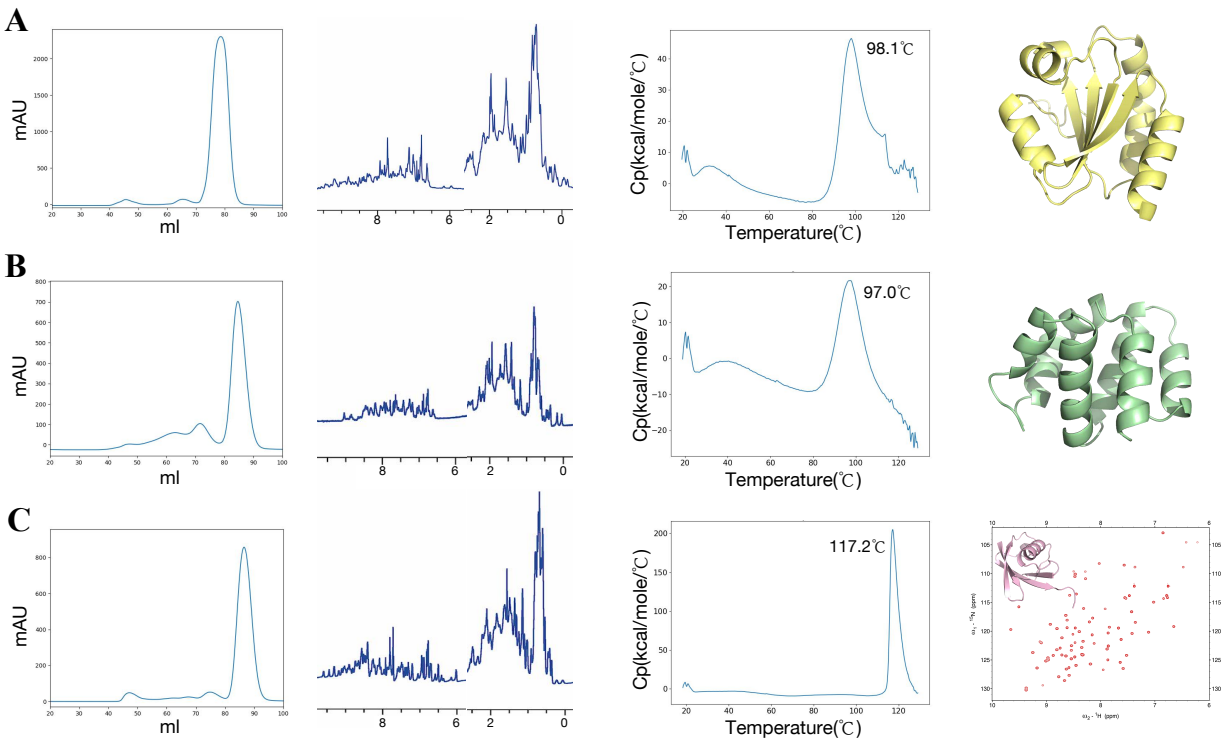

Figure 4. Results of experimental analysis of designed proteins. For each of the three backbone targets considered, the results of one designed protein are shown (**A**, 1r26-A3; **B**, 1cy5-A7; and **C**, 1ubq-A4). From left to right: Results of size exclusion chromatography experiments; ^{1}H NMR spectra; thermal capacity versus temperature curves from differential scanning calorimetry experiments; The experimentally solved structures (1r26-A3, 1cy5-A7) or NMR HSQC (1ubq-A4). The C_α RMSDs of the structures from the corresponding backbones are 0.57 Å for 1r26-A3 and 0.88 Å for 1cy5-A7. For 1ubq-A4, the target backbone is shown together with the HSQC spectrum.

of the designed sequences have been effectively minimized through the self-consistent iterative process. The results in Figures 3E and 3F together indicate that the $-logits$ metric has constituents that are orthogonal to the Rosetta energy. The partial orthogonality between the two metrics is also exemplified by the moderate correlations between the per residue $-logits$ values and the Rosetta energies of the designed and the native sequences (the respective Pearson correlation coefficients are 0.24 and 0.17; see Supplementary Figure 2C.).

2.5 Experimental analysis of designed sequences

We experimentally examined the sequences designed by ABACUS-R for three natural backbones (the PDB IDs of the targets are 1r26, 1cy5, and 1ubq). These backbones have been chosen because they had been used to evaluate the ABACUS model by wet experiments [9, 10, 22, 23]. To assess the method in an unbiased way, the automatically generated sequences have been used as is for experimental examination without any post-design selection or adjustment.

Two batches of wet experiments have been carried out to examine proteins designed using two different protocols.

Table 1. Overview of experiments on designed sequences.

Target[a]	-logits[b]	Identity(%)[c]	Identity(%)[d]	Examined/ Expressed/ Soluble	Crystallization attempted/ Crystal obtained	Structure solved	HSQC spectra measured
1r26	-6.76±0.21	53.4±2.9	83.9±5.3	10/10/10	5/4	3	0
	-6.50±0.17	47.1±3.9	57.4±2.2	10/8/7	1/1	1	2
1cy5	-6.75±0.19	37.0±1.4	80.8±2.6	10/10/10	7/2	1	0
	-6.49±0.18	40.9±2.4	55.1±2.3	10/7/6	2/0	0	1
1ubq	-6.30±0.22	45.1±2.7	78.9±4.8	7/6/6	6/1	0	1
	-6.12±0.20	43.2±4.1	61.4±2.2	10/10/10	1/0	0	2
Total number of experimental examined designs: 57. Total number of designs that are soluble and well-folded based on at lease on ^{1}H-NMR: 49.					Total number of designs with HSQC spectrum measured[e]: 6. Total number of designs with solved crystal structures[e]: 5. (RMSD[f]: 0.51 $\sim$ 0.88 Å)		

[a] For each target, the first and the second rows correspond to designs examined in the first batch and the second batches of experiments, respectively.
[b] Average $-logits$ per residue.
[c] Average identity between native sequence and designed sequences.
[d] Average identity with other designed sequences in the same batch.
[e] These two groups of designs do not overlap.
[f] The main C_α atoms RMSD of designs from targets.

The first protocol employs the self-consistent iterations described above to maximally converge the sidechain types of all residues. While this protocol effectively minimizes the $-logits$ values, the designed sequences are highly similar to each other (the identities between different sequences designed for the same backbone are around 80%, see Table 1 and Supplementary Table 3). In the second protocol, the sidechain types are not chosen to be the ones of the highest probabilities as predicted by the decoders. Instead, they are sampled from distributions derived from the output of the decoders (see Method). The second protocol leads to more variable sequences from different runs, albeit with somewhat higher $-logits$ values(see Table 1 and Supplementary Table 4).

Among the 27 sequences examined in the first batch, 26 have led to successful protein expression in $E.coli$. All the expressed proteins could be readily purified in soluble forms. The purified proteins have been subjected to a range of experimental analysis including size exclusion chromatography, ^{1}H NMR spectrum measurement, NMR heteronuclear single quantum coherence (HSQC) spectrum measurement, differential scanning calorimetry (DSC), and protein crystallization and X-ray structure determination (Table 1) . Complete data from these experiments are reported in Supplementary Figures 3-5. Representative results for three designed sequences (one sequence for each backbone target) are shown in Figures 4A-C as examples.

The size exclusion chromatography results and the ^{1}H NMR spectra have covered all the purified proteins in the first batch (Supplementary Figures 3-5 and Supplementary Table 5). They indicate that all these proteins are monomers and likely to fold into well-defined 3D structures. Five proteins covering the three target backbones have been measured by DSC experiments, which showed that all these proteins unfold cooperatively at high temperatures from 97 to 117 °C (Supplementary Figure 6). High resolution structures of four proteins, including three designed

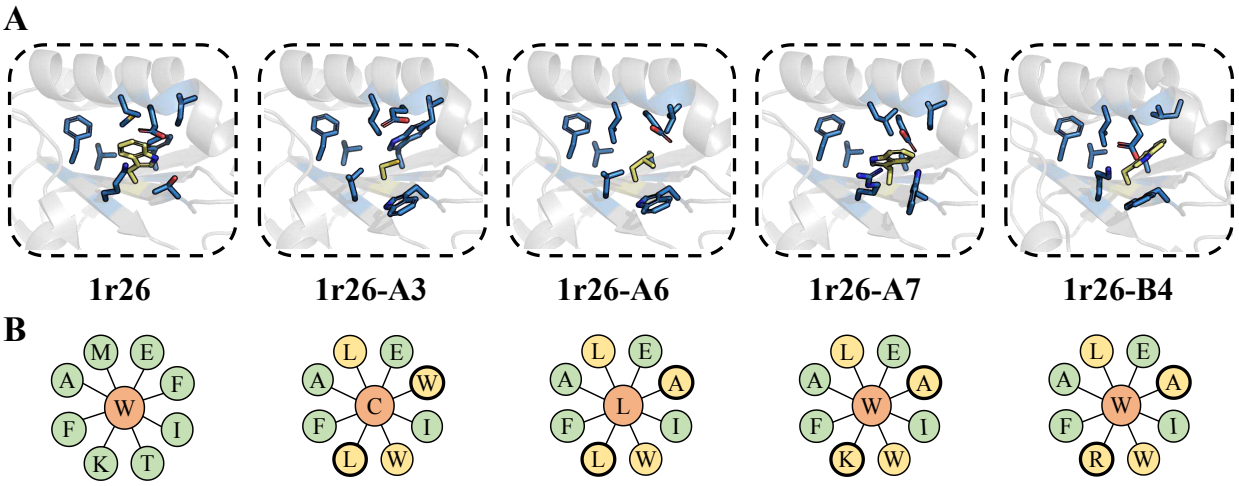

Figure 5. Variable sidechain types and sidechain packing have been designed around a core residue (residue 24) in the 1r26 backbone. **A**, Crystal structures of the different proteins showing (as sticks) the sidechains of residue 24 surrounded by residues (22, 26, 36, 37, 40, 52, 54, and 75. 1r26 is the native sequence. 1r26-A3, 1r26-A6, 1r26-A7 and 1r26-B4 are ABACUS-R designed sequences). **B**, Diagrams showing the sidechain types of residue 24 and its surrounding residues in the different proteins. In each diagram, the orange circle indicates the central residue; the green circles indicate residues that are of the native sidechain types; the yellow circles indicate residues for which non-native sidechain types have been designed; the circles with bold borders indicate residues whose sidechain types did not completely converge in the self-consistent iterations.

for the 1r26 backbone and one designed for the 1cy5 backbone, have been solved with X-ray crystallography. All the solved structures agree with the corresponding design targets with high precision (the root mean square deviations or RMSDs of C_α atoms ranged from 0.51 to 0.88 Å. See Supplementary Figures 3-4). For proteins designed for the 1ubq backbone, we only obtained crystals for one protein (1ubq-A4), which were twinned and no X-ray structure had been solved. However, the NMR HSQC spectrum and the DSC results of 1ubq-A4 (Figure 4C) prove that this protein folds into a well-defined 3D structure, which we expect to agree well with the 1ubq target backbone because the sequence identity between 1ubq-4A and 1ubq is 46.1%.

In the second batch of experiments, we examined proteins obtained by using the modified protocol that designs more diverse sequences. Table 1 shows that the identities between the different sequences designed for the same target are around 58%. The averaged $-logits$ values listed in the same table are slightly higher than those of the sequences examined in the first batch. Among the 30 designed proteins analyzed, 25 led to successful protein expression in $E.coli$, and 23 could be purified in soluble form. The purified proteins have been subjected to the same types of experimental analysis as the first batch proteins (Table 1 and Supplementary Table 6). The results are reported in Supplementary Figures 6-10. Again, the size exclusion chromatography and NMR data (Supplementary Figure 7-9) and also Supplementary Figure 10 showing HSQC spectra suggest that all these proteins are monomeric and fold into well-defined 3D structures. The DSC curves for five proteins covering the three target backbones indicate that

these proteins unfold cooperatively at high temperatures from 85 to 118 °C (see Supplementary Figure 6). One high resolution structure for a second batch protein has been solved (1r26-B4), whose C_α RMSD from the 1r26 target backbone is 0.67 Å (Supplementary Figure 7). Thus the modified protocol leads to somewhat decreased but still acceptable success rate. Moreover, those successfully designed proteins are still very stable, although their unfolding temperatures appear to be somewhat lower than the proteins designed using the maximum convergence protocol for the same targets (see the DSC curves of the first and second batch proteins in Supplementary Figure 6. Note that there are exceptions to this observation).

3 Discussion

3.1 The computational model of ABACUS-R compared with related models

A number of previous studies have investigated deep learning-based approaches to inverse protein folding. Some of them did not consider any sidechain type-dependent coupling between different residues and thus only aimed at predicting sequence profiles [31, 32, 33, 34]. The other studies considered this coupling and had attempted designing overall sequences from scratch [17, 29, 30]. Among the latter type of methods, the ProteinSolver method treated sequence design as a constraint satisfaction problem and tried to solve it using deep graph neural networks [30]. The other methods [17, 29] used deep learning networks to predict the sidechain type of a central residue based its local environment. This is conceptually similar to the encoder-decoder network of ABACUS-R. However, there are large variations between different models in the network architectures as well as in what information about the local environment has been used, and how.

All methods have used information about backbone structures, with some methods having used somewhat incomplete information (*e.g.*, only inter-residue distances [30]) and other methods having used more complete information by considering both distances and orientations [29] or by using a "voxel" grid to map the types and locations of all atoms [17]. Information about sidechains were not used in methods that only predicted sequence profiles. The sequence generator proposed by Ingraham *et al.* [29] also did not use any sidechain information in its network for encoding the local environment. Instead, this model autoregressively considered the influences of the sidechain types of upstream residues on the downstream residues while sequentially sampling the sidechain types of individual residues along the sequence. We note that a drawback of this approach based on autoregression is that the effects of any improperly chosen upstream residue will be accumulated and propagated downstream, which may lead to an exponentially decaying probability of generating truly foldable overall sequences with increasing sequence length. The 3D convolution network (3DCNN) model [17, 33] has used both sidechain type and sidechain conformation information by mapping the atom types and locations of all atoms, including sidechain atoms, to a voxel grid. A major difficulty

for this approach is that accurate coordinates of sidechain atoms are not available in sequence redesign, and these coordinates can only be approximately constructed by using rotamers [18], which may lead to significant degradation of overall accuracy. Besides this, the solution space of sequence design by 3DCNN was spanned by not only the dimensions of the sidechain types but also the dimensions of the sidechain atomic coordinates, which may make the overall optimization problem too complicated to be solved by a simple approach such as self-consistent iteration.

In its modeling of the 3D local environment, ABACUS-R differs from previous methods in an aspect that is fundamentally important for sequence design: the input of the encoder of ABACUS-R has included, alongside with those backbone-only structural features, the sidechain types but not the sidechain conformations of the neighboring residues. On the one hand, the use of sidechain type information accounts for the sidehcain type-dependent coupling between the central residue and its neighbors. Thus not just sequence profiles but actual overall sequences can be designed through the self-consistent iteration approach, which does not suffer from the error accumulation and propagation problem of the autoregressive approach. On the other hand, by leaving out sidechain conformations from the input of the encoder of ABACUS-R, only the sidechain types need to be updated during the self-consistent iterations. This 1) eliminates the use of approximate rotamers and 2) makes sequence design an easy optimization problem. Thus it is the distinctive way in which ABACUS-R models the 3D local environments of individual residues that has provided the basis for ABACUS-R to overcome some of the most serious problems of previous models.

3.2 The performance of ABACUS-R compared with previous deep learning methods

The performances of sequence design methods were most commonly evaluated by carrying out sequence design for natural backbones and examining the rates of recovering the native sidechain types. Although the native sidechain type recovery rates for methods that only predict sequence profiles have been reported to range from 36% to 56% [31, 32, 34], these methods alone cannot be used to design physically plausible overall sequences (to do so they need to be integrated with other models that can treat sidechain type-dependent inter-residue interactions). As for other deep learning models, an average recovery rate of 39.2% was reported for the autoregressive sequence generator applied to 40 targets of natural backbones [29], while the recovery rate ranges for different target backbones were reported to be from 25% to 45% for the 3DCNN model [17] and from 37% to 44% for the ProteinSolver model [30]. Compared with these values, the average recovery rate by ABACUS-R is significantly higher. The same conclusion can be drawn by comparing the design results of different methods for the same specific target backbones: for the same 40 target backbones used to benchmark the auoregressive model, the average native recovery rate of ABACUS-R is 44.6%, which is to be compared with the averages of 39.2% reported in ref [29] (for a fair comparison, the ABACUS-R network parameters used in this particular benchmark have been learned using a set of training structures that excluded structures in the same CATH topologies as the targets in ref [29]); for four target backbones for which the average

native sidechain type recovery rates of 3DCNN were reported to be 30.0% to 35.4%, the average native recovery rates of ABACUS-R is 33.7% to 44.1% (Supplementary Table 7).

The most critical evaluation of a sequence design method should be based on examining *de novo* designed sequences using wet experiments. For the 3DCNN model [17], a number of selected designs for several natural backbones have been experimentally assessed to have desired secondary structure contents and to fold cooperatively according to circular dichroism (CD) signatures. The same types of results were reported for several designs generated by ProteinSolver [30]. So far, the only experimentally solved atomic structures for *de novo* sequences designed using a deep learning method were the X-ray crystal structures of two sequences designed by the 3DCNN model [17]. The target backbone of these sequences was a *de novo* TIM-barrel of four-folded symmetry, which comprised structure elements that were likely to be far more regular or ideal than those in natural backbones. Thus when it comes to *de novo* sequence design for (natural) backbones that are abundant with diverse, none-ideal structure elements, we do not know about previous experimental validations of a deep learning method by atomic structures solved using crystallography or NMR.

Given these backgrounds, the experimental results reported here for ABACUS-R represent significant advances: the success rates of ABACUS-R are high (49 of 57 experimentally examined proteins are soluble monomers that appear to be well-folded, with these proteins covering three distinctive target backbones and having been examined "as is" without considering any post-design filtering); data from NMR HSQC measurements, DSC measurements, and crystallographic analysis are all consistent with that the respective proteins are of well-defined 3D structures, of high unfolding temperatures, and of structures that precisely match the corresponding target backbones at the atomic level.

3.3 ABACUS-R compared with previous energy function-based methods

For the three target backbones considered here, ABACUS-R has exhibited far higher success rates than our previous ABACUS model. Almost all designs (26 of 27 in the first batch) by ABACUS-R could be expressed and purified as soluble, monomeric, and well-folded proteins. This is contrast to that more than half of the original ABACUS designs on these three targets could not be be expressed or purified; only after introducing mutations through directed evolution, structures of ABACUS designs for the 1r26 and 1cy5 targets had been successfully solved [9, 22]. The solved structures of the ABACUS-R sequences have much smaller RMSDs (all below 1 Å) from the corresponding targets than the structures of the ABACUS sequences. The success rates found here for ABACUS-R are also significantly higher than those reported for the RosettaDesign method [8]. Recently, Marin *et al.* [14] have investigated the sensitivity of RosettaDesign to the structure of the target backbone by experimentally examining proteins designed for eight different backbones of thioredoxin proteins (the 1r26 target used here is also a thioredoxin). They reported that in the most favorable case (*i.e.*, by considering the target backbone for which RosettaDesign exhibited the highest success

rate), only three out of eight experimentally examined designed proteins could be purified as monomers. In an earlier study of these authors, the crystal structure of one protein designed by Rosetta for a thioredoxin backbone had been determined, with the various models in an asymmetric unit having RMSDs of 1.8 to 2.0 Å from the target backbone. These RMSDs are obviously larger than the corresponding RMSDs exhibited by the ABACUS-R designed proteins. Thus the ABACUS-R redesigned sequences can much more precisely recover the none-ideal backbone features in the targets than sequences designed by ABACUS2 [10] or RosettaDesign [8].

In Figure 3E and F, we compared the Rosetta energies of the ABACUS-R-designed sequences, the ABACUS2-designed sequences, and the native sequences for 100 test backbone targets. Interestingly, while the ABACUS-R sequences are of comparable Rosetta energies to the native sequences, the ABACUS2 sequences are of systematically lower Rosetta energies than both the native and the ABACUS-R sequences. This indicates that the Rosetta energy function has not captured factors that contributed to the higher success rate of folding of the ABACUS-R sequences relative to the ABACUS2 sequences. Nor has it captured factors that contributed to the higher thermal stability (as suggested by the high unfolding temperatures of the ABACUS-R designs for 1r26, 1cy5 and 1ubq) of the ABACUS-R sequences relative to the native sequences.

Besides success rate and structure precision, ABACUS-R can also be more robust than ABACUS or other energy function-based methods for the purpose of designing more diverse sequences. For the energy function-based methods, an important factor that limits their ability to design diverse sequences is that they may not be able to extensively explore the different ways of sidechain packing inside the core regions of a given backbone, because these models intrinsically suffer from the incompatibility between the inaccurately reconstructed positions of sidechain atoms using rotamers and the high sensitivity of some of the energy terms (*e.g.*, the van der Waals repulsion terms) to atomic positions. As ABACUS-R does not explicitly use sidechain structures, it does not suffer from this limitation. This may facilitate ABACUS-R to design varied sequences that use alternative sidechain combinations to form equally well-packed cores. As an example to illustrate this, Figure 5 shows how a group of core residues of different combinations of sidechain types are packed in the native 1r26 structure and in the crystal structures of four ABACUS-R designed proteins for this backbone. Besides the variations of the core residues, the diversity of the overall sequences designed by ABACUS-R can be further increased by applying the sidechain type sampling protocol as described in Method. The results of our second batch experiments show that by applying this protocol, the diversity of the designed sequences can be increased to the levels of less than 60% sequence identity with some (acceptable) compromises in success rate.

4 Conclusions

The ABACUS-R method is based on deep learning and self-consistency. It uses a Transformer to encode the 3D local environment of a central residue. The output of this encoder constitutes a general representation, from which the sidechain type and other attributes of the central residue can be decoded. In computational tests of redesigning the overall sequences for natural protein backbones, ABACUS-R reaches significantly higher native sidechain type recovery rates than existing deep learning methods. In evaluations based on wet experiments, ABACUS-R outperforms state-of-the-art energy function-based methods by large margins, achieving stronger robustness (namely, significantly higher success rates and more diverse designed sequences) and higher structural precision (RMSDs between the structures of the designed proteins and the target backbones below 1 Å) at much lower computational costs. To our knowledge, this is the first time that a deep learning-based method has been proven by experiments to outperform conventional methods of *de novo* protein sequence design by large margins. Thus ABACUS-R represents a significant advance both in the application of deep learning in the study of proteins and in the method development of protein design. This method can find wide applications in studies that use protein design and protein engineering.

5 Method

5.1 Modeling the 3D local environment

The set of residues that form the 3D local environment of a central residue is defined to be its k-nearest neighbor residues according to the $C_\alpha - C_\alpha$ distances. As input of the encoder, we have considered the following three types of features for each neighboring or surrounding residue: its location and orientation relative to the central residue ($\mathbf{X}_{\text{SPA}}$), its sequence position relative to the central residue ($\mathbf{X}_{\text{RSP}}$), and its sidechain type ($\mathbf{X}_{\text{AA}}$). The raw input for the different types of information are defined as below.

The components of $\mathbf{X}_{\text{SPA}}$ are defined using 1) the $C_\alpha - C_\alpha$ distance from the environment residue to the central residue; 2) the Cartesian coordinates of the environment residue's C_α in a local coordinate frame defined using the backbone N, C_α and C atoms of the central residue (in this frame, C_α is at the origin, the $C_\alpha - N$ direction is along the $\mathbf{x}$ axis, and the $N - C_\alpha - C$ plane corresponds to the $\mathbf{x} - \mathbf{y}$ plane); and 3) the rigid body rotation that aligns the orientations of the two local coordinate frames of the central and the environment residues. The $C_\alpha - C_\alpha$ distance is mapped to a vector of 16 components using a set of Gaussian radial basis functions centered around 16 distance values ranged from 0 to 20 Å; the Cartesian coordinates are used as is; the rigid body rotation is mapped to a 3D vector whose direction corresponds to the axis of the rotation and whose length corresponds to the magnitude of the rotated angle.

After these mappings, $\mathbf{X}_{\text{SPA}}$ for each residue is a vector of 22 real-valued components.

The components of $\mathbf{X}_{\text{RSP}}$ are defined based on the difference between the position indices of the neighboring residues and the central residue, *i.e.*, $\Delta i = i - i_{\text{central}}$. The $\mathbf{X}_{\text{RSP}}$ is a one-hot vector that encodes 129 states, with each integer value of Δi from -64 to +63 corresponding to one of 128 states, and the remaining one state encoding whether the corresponding neighboring residue is not on the same peptide chain as the central residue.

The $\mathbf{X}_{\text{AA}}$ is a one-hot vector that encodes the 20 sidechain types plus a special 'MASKED' type (for the central residue, see below).

Features of the central residue is mapped to vectors of the same dimensions as the neighboring residues, only that all components of its $\mathbf{X}_{\text{SPA}}$ vector are zeros, and its $\mathbf{X}_{\text{AA}}$ value is mapped to the 'MASKED' type. Besides these, the backbone conformation centered around the central residue ($\mathbf{X}_{\text{BB}}$) is also considered. The components of $\mathbf{X}_{\text{BB}}$ are the 15 backbone torsional angles ϕ_{i-2}, ψ_{i-2}, ω_{i-2} $\cdots$ ϕ_{i+2}, ψ_{i+2}, ω_{i+2}, in which i is the position index of the central residue. The angles are divided by π to obtain values in $[-1, +1)$. To maintain the same input dimensions for vectors encoding the central and the neighboring residues, the $\mathbf{X}_{\text{BB}}$ vectors are also formally considered for all the neighboring residues, but the values of all their components are zero.

To produce input for the Transformer encoder from the above raw input feature vectors, the one-hot-encoded feature vectors $\mathbf{X}_{\text{AA}}$ and $\mathbf{X}_{\text{RSP}}$ are linearly transformed, while the feature vectors $\mathbf{X}_{\text{SPA}}$ and $\mathbf{X}_{\text{BB}}$ which are of real-valued components are transformed using a linear layer. The resulting intermediate vectors are respectively noted as $\mathbf{E}_{\text{AA}}$, $\mathbf{E}_{\text{RSP}}$, $\mathbf{E}_{\text{SPA}}$, and $\mathbf{E}_{\text{BB}}$, which are simply concatenated together to form the overall input vector $\mathbf{E}$, namely,

$$\begin{aligned}
\mathbf{E}_{\text{AA}} &= \mathbf{W}_{\text{AA}}\mathbf{X}_{\text{AA}}, \\
\mathbf{E}_{\text{RSP}} &= \mathbf{W}_{\text{RSP}}\mathbf{X}_{\text{RSP}}, \\
\mathbf{E}_{\text{SPA}} &= \mathbf{W}_{\text{SPA}}\mathbf{X}_{\text{SPA}} + \mathbf{b}_{\text{SPA}}, \\
\mathbf{E}_{\text{BB}} &= \mathbf{W}_{\text{BB}}\mathbf{X}_{\text{BB}} + \mathbf{b}_{\text{BB}}, \\
\mathbf{E} &= [\mathbf{E}_{\text{AA}} \;\; \mathbf{E}_{\text{RSP}} \;\; \mathbf{E}_{\text{SPA}} \;\; \mathbf{E}_{\text{BB}}],
\end{aligned} \tag{1}$$

where the matrices $\mathbf{W}_{\text{AA}}$, $\mathbf{W}_{\text{RSP}}$, $\mathbf{W}_{\text{SPA}}$, $\mathbf{W}_{\text{BB}}$ and the biasing vectors $\mathbf{b}_{\text{SPA}}$ and $\mathbf{b}_{\text{BB}}$ are comprised of trainable elements, which are shared between all residues.

5.2 Architecture of the Transformer-based encoder

We note the set of $\mathbf{E}$ vectors for all residues as $\{\mathbf{E}_n; n = 0, 1, 2, ..., k\}$, in which $\mathbf{E}_0$ is the vector for the central residue. These set of vectors are fed to a Transformer of 12 blocks, as shown in Figure 1. Each block of the Transformer is composed sequentially of a multi-head self-attention module [36], a layer normalization transformation [42], a feed-

forward module, and a second layer normalization transformation, as formulated below,

$$\begin{aligned}
\mathbf{F}_n^0 &= \mathbf{E}_n, \\
\mathbf{Q}_n^i = \mathbf{W}_i^Q \mathbf{F}_n^i,\ \ \mathbf{K}_n^i &= \mathbf{W}_i^K \mathbf{F}_n^i,\ \ \mathbf{V}_n^i = \mathbf{W}_i^V \mathbf{F}_n^i, \\
\mathbf{F}_n'^i &= \mathrm{LN}(\mathbf{F}_n^i + \mathrm{MHA}(\mathbf{Q}_n^i, \{\mathbf{K}_m^i, \mathbf{V}_m^i; m = 0, 1, 2, ..., k\})), \\
\mathbf{F}_n^{i+1} &= \mathrm{LN}(\mathbf{F}_n'^i + \mathrm{FFN}(\mathbf{F}_n'^i)),
\end{aligned} \tag{2}$$

where the residue index n is from 0 to k, the block index i is from 1 to 12, the $\mathrm{MHA}(\cdot,\ \cdot)$, $\mathrm{LN}(\cdot)$ and $\mathrm{FFN}(\cdot)$ operations refer to the multi-head self-attention module, the feed-forward module, and the layer normalization transformation, respectively. From the output of the above Transformer, the vector $\mathbf{F}_0^{12}$ is taken as a general representation of the local environment of the central residue.

5.3 The decoders and training losses

Different attributes of the central residues are decoded by passing $\mathbf{F}_0^{12}$ through different perceptron networks (decoders), each decoder network having a single hidden layer.

For the sidechain type decoder, its output layer contains twenty perceptron nodes, the output values (the $logits$) are transformed using Softmax to yield normalized probabilities predicted for the 20 sidechain types. Namely, for sidechain type a,

$$p_a = e^{logits_a} / \sum_{b=1}^{20} e^{logits_b}. \tag{3}$$

The cross-entropy loss function is used to measure the decoding loss, namely, for each training item,

$$\mathcal{L}_{\mathrm{AA}} = -\sum_{a=1}^{20} y_a log(p_a), \tag{4}$$

where a is sidechain type index, $\mathbf{y} = (y_0, y_1, \cdots, y_{20})$ is the one-hot vector produced by the ground-truth label (*i.e.*, the native sidechain type), and $\mathbf{p} = (p_0, p_1, \cdots, p_{20})$ is the vector of the predicted probabilities.

The decoders and losses for the secondary structure (SS) state are defined in the same way as that for the sidechain type. Both the three-state and the eight-state categorization schemes are used at the same time. The cross-entropy losses are respectively noted as $\mathcal{L}_{\mathrm{SS3}}$ and $\mathcal{L}_{\mathrm{SS8}}$.

For the numerical attributes including the solvent accessible surface area (SASA) and the X-ray B factor, we first normalized the attributes using the means and standard variations computed from the training data. The corresponding decoders regress the normalized attributes. The losses $\mathcal{L}_{\mathrm{SASA}}$ and $\mathcal{L}_{\mathrm{Bfactor}}$ are defined as the L-1 errors of the regressions.

The total loss for the multi-task learning task is given by

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\text{AA}} + \lambda_1(\mathcal{L}_{\text{SS3}} + \mathcal{L}_{\text{SS8}}) + \lambda_2\mathcal{L}_{\text{SASA}} \\ + \lambda_3\mathcal{L}_{\text{Bfactor}} + \lambda_4(\mathcal{L}_{\chi_1} + \mathcal{L}_{\chi_2}),\end{aligned} \tag{5}$$

where λ_1, λ_2, λ_3 and λ_4 are the trade-off parameters to balance the different losses.

5.4 The self-consistent iterations for overall sequence design

For a given target backbone, we start from an initial sequence with randomly chosen sidechain types for all residues. In each iteration, we randomly select a number of residues, consider each of them as the central residue, apply the pre-trained encoder-decoder network to its current 3D local environment, and update the sidechain type of the central residue according to the output of the (sidechain type) decoder. Within one iteration, the calculations for all the selected residues are executed in parallel to take advantage of parallel computation. The initial number of residues selected for parallel sidechain type updating is set to be 80% of the sequence length, and this number is gradually reduced in subsequent iterations until the final iterations in which the residues are considered one by one.

During the iterations, we monitor the total $-logP$ value of the evolving sequence, which is defined as

$$-logP_{\text{total}} = \sum_{i=1}^{L} -logp^i_{a_i}, \tag{6}$$

where L is the length of the sequence, i is the residue index, a_i is the index of the sidechain type at position i in the current sequence, and $p^i_{a_i}$ is the probability of the corresponding sidechain type predicted by the encoder-decoder network according to the current local environment of residue i (which depends on the current sidechain types of i's neighboring residues).

In generating the sequences for examination by wet experiments, two different protocols have been applied to update the sidechain type of a central residue during the iterations. In the first "normal" protocol, the sidechain type of the maximum probability (as predicted by the corresponding decoder) is always chosen. This protocol is used to generate maximally self-consistent overall sequences examined by the first batch of experiments. In the second protocol, the sidechain type is sampled according to a probability distribution derived from the output of the encoder. To control the diversity of the sampled sidechain type, we mapped the original distribution predicted by the decoder into a new distribution using the following formula,

$$p'_a = (p_a)^\alpha / \sum_{b=1}^{20} (p_b)^\alpha. \tag{7}$$

The value of the parameter α controls the diversity of the sampled sidechain type: for a very large positive value of α, the first and the second protocol become equivalent; decreasing the value of α increases the diversity of the designed sequences, but it will also cause the $-logits$ metric to increase (Supplementary Figure 11). To design the sequences (which have inter-sequence identity of around 60% for sequences designed for the same target) for examination in the second batch of experiments, we have used the value of 1.5 for α, which still lead to $-logits$ values below those of the native sequences (Supplementary Figure 11).

5.5 Other implementation and computation details

The ABACUS-R model has been implemented in Python. The encoder-decoder network has been implemented using the PyTorch package. Hyperparameters that have been explored during the optimization of the model include the number of nearest neighbor residues (*i.e.*, the value of k) to be considered, the number of blocks of the Transformer encoder, the dimensions of all the internal layers or modules, and the weights of the different losses. Some of the finally used values are: 20 for k; 12 for the number of blocks of the Transformer; 0.2, 0.2, 0.2 and 0.5 for λ_1, λ_2, λ_3 and λ_4. The networks have been trained by using the Adam optimizer [43] as implemented in PyTorch with the parameters $\beta_1 = 0.9$, $\beta_2 = 0.99$ and with a gradually decreasing learning rate from 10^{-4} to 10^{-6}. The PDB structures for training and testing are 25234 X-ray structures selected by using the PISCES [44] server with a resolution cutoff of 2.5 Å and a sequence-identity cutoff of 50%.

The sequence design runs for the target proteins considered here usually take around 350 to 400 iterations to reach convergence (see also Figures 3A and B). For the sake of simplicity, we used 1000 iterations for all the targets. For a target of about 100 residues, 1000 iterations take about 30 seconds on a computer with a GeForce RTX 3090 GPU.

The Rosetta energies have been computed using the RosettaDesign program [8] version 3.12. As ABACUS-R does not explicitly reconstruct sidechain conformations, all sidechains of all the evaluated sequences (*i.e.,* the native sequences and the designed sequences) have been repacked by using the FixBB option of Rosetta, and then the complete structures have been relaxed using Rosetta RelaxBB to determine the minimized energies.

5.6 Experimental methods

DNA sequences of designed proteins were constructed into the NdeI and XhoI sites of pET-22b(+) by General Biotech and TsingKe Biotech. Plasmids encoding the designs were transformed into $Escherichia\ coli$ BL21(DE3) cells. Protein expression was induced at OD_{600} = 0.7 with 1 mM IPTG for 20 hours at 16°C. For ^{1}H-^{15}N HSQC NMR study, uniformly ^{15}N-labeled proteins were prepared by growing the bacteria in inorganic medium (24 g/L KH_2PO_4, 5 g/L NaOH, 0.5 g/L NH_4Cl, 2.2 mM $MgSO_4$, 0.1 mM $CaCl_2$ and 2.5 g/L glucose) using $^{15}NH_4Cl$ as isotope source. Cells were harvested and sonicated in buffer containing 20 mM Tris and 500 mM NaCl at pH 7.8. The soluble

supernatant was purified by Ni^{2+} affinity chromatography and concentrated in buffer containing 20 mM Tris, 300 mM NaCl and 1 mM EDTA, pH 7.8 and then subjected to size exclusion chromatography in a Superdex 75 column with the ÄKTA purifier system (GE Healthcare).

The thermal stability of designed proteins were evaluated by nano differential scanning calorimetry (nanoDSC) (TA Instruments, DE). The protein samples were prepared in buffer containing 300 mM NaCl, 20 mM Tris, 1 mM EDTA, pH8.0 with a protein concentration of 3-6 mg/mL. The protein melting temperature were monitored in a 0.33 mL cell from 25 to 125 °C at a heating rate of 1 °C/min. Responses of proteins were measured by reheating and scanning for three times. The nanoDSC scans were background-corrected and analyzed with Launch NanoAnalyze software.

All NMR data were collected at 298 K on a Bruker DMRX600 spectrometer equipped with triple resonances, self-shielded z-axis gradient probes. Two-dimensional HSQC-NMR samples typically contained 0.35-0.5 mM ^{15}N-labeled proteins, 25 mM NaH_2PO_4, 150 mM NaCl, 2 mM EDTA (pH 6.9) and 10% (v/v) D_2O. Data were processed using the programs NMRDraw/NMRPipe [45] and SPARKY [46].

Crystallization screening was conducted at 289 K by the hanging-drop vapor diffusion method using various screening kits. Purified and concentrated proteins (15-20 mg/mL) were used for crystallization. The crystals for data collection were obtained in 24-48 hours in PEG 1500 for 1r26-A3, 2.5 M Ammonium sulfate and 0.1 M BIS-TRIS propane, pH7.0 for 1r26-A6, 2% v/v 1.4-Dioxane, 0.1 M BICINE pH9.0, 10% w/v PEG 20000 for 1r26-A7, 27 % w/v PEG 2000 MME and 0.1 M Sodium cacodylate, pH6.5 for 1r26-B4. Crystals of 1cy5-A7 appeared in 7 days in buffer containing 12% w/v Polyethylene glycol 3,350 and 4% v/v Tacsimate, pH7.0. All crystals were shortly soaked in reservoir solution supplemented with 30% glycerol (v/v) and then flash frozen in liquid nitrogen. The diffraction data were collected on BL19U1 [47] and BL02U1 beamlines at 0.9785 to 0.9791 Å. Data sets were processed by using HKL2000 program [48] for 1r26-A3, 1r26-A6, 1r26-A7 and XDS program [49] for 1cy5-A7 and 1r26-B4. The designed structures served as search model for molecular replacement with MOLREP [50]. The final structures were refined by PHENIX [51]. The statistics for data collection and structural refinement are summarized in Supplementary Table 8.

6 Acknowledgements

This work was supported by the National Key R&D Program of China (2018YFA0900703 to HY Liu. and 2018YFA0901600 to Q Chen), National Natural Science Foundation of China (21773220 to HY Liu., 31971175 and 32171411 to Q Chen) and Youth Innovation Promotion Association, Chinese Academy of Sciences (2017494 to Q Chen). We thank the staffs from BL19U1 and BL02U1 beamlines of National Facility for Protein Science in Shanghai (NFPS) and of Shanghai

Synchrotron Radiation Facility for assistance during crystallographic data collection. We thank Mengqi Lv and Yuehui Yun for assistance with X-ray diffraction data collection and processing.

7 Author contributions

HY Liu, HQ Li, YF Liu and WL Wang conceived the computational framework. Q Chen, L Zhang and YF Liu designed the experimental study. YF Liu and WL Wang wrote the computer programs and performed the calculations under the supervision of HY Liu and HQ Li. L Zhang performed experimental analyses under the supervision of Q Chen and HY Liu. M Zhu, CC Wang and FD Li analyzed crystallographic data. JH Zhang collected and processed NMR data. YF Liu, WL Wang and HY Liu wrote the paper with input from all other authors.

8 Data availability

Complete lists of the training and testing proteins, the amino acid sequences designed for the 100 targets by $\text{Model}_{\text{eval}}$, and the amino acid sequences and DNA sequences of the experimentally examined proteins can be downloaded from https://github.com/liuyf020419/ABACUS-R. The experimentally solved protein structures have been deposited in the Protein Data Bank with accession codes 7VQL (1r26-A3), 7VQV (1r26-A6), 7VQW (1r26-A7), 7VTY (1cy5-A7) and 7VU4 (1r26-B4).

9 Code availability

The codes for training and applying ABACUS-R can be downloaded from https://github.com/liuyf020419/ABACUS-R.

References

[1] Brian Kuhlman and Philip Bradley. Advances in protein structure prediction and design. *Nature reviews molecular cell biology*, 20:681–697, 2019.

[2] Po-Ssu Huang, Scott E Boyken, and David Baker. The coming of age of de novo protein design. *Nature*, 537:320–327, 2016.

[3] Daniel-Adriano Silva, Shawn Yu, Umut Y Ulge, Jamie B Spangler, Kevin M Jude, Carlos Labão-Almeida, Lestat R Ali, Alfredo Quijano-Rubio, Mikel Ruterbusch, Isabel Leung, et al. De novo design of potent and selective mimics of il-2 and il-15. *Nature*, 565:186–191, 2019.

[4] Longxing Cao, Inna Goreshnik, Brian Coventry, James Brett Case, Lauren Miller, Lisa Kozodoy, Rita E Chen, Lauren Carter, Alexandra C Walls, Young-Jun Park, et al. De novo design of picomolar sars-cov-2 miniprotein inhibitors. *Science*, 370: 426–431, 2020.

[5] Justin B Siegel, Alexandre Zanghellini, Helena M Lovick, Gert Kiss, Abigail R Lambert, Jennifer L St Clair, Jasmine L Gallaher, Donald Hilvert, Michael H Gelb, Barry L Stoddard, et al. Computational design of an enzyme catalyst for a stereoselective bimolecular diels-alder reaction. *Science*, 329:309–313, 2010.

[6] Yinglu Cui, Yinghui Wang, Wenya Tian, Yifan Bu, Tao Li, Xuexian Cui, Tong Zhu, Ruifeng Li, and Bian Wu. Development of a versatile and efficient c–n lyase platform for asymmetric hydroamination via computational enzyme redesign. *Nature catalysis*, 4:364–373, 2021.

[7] 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*, 302:1364–1368, 2003.

[8] Andrew Leaver-Fay, Michael Tyka, Steven M Lewis, Oliver F Lange, James Thompson, Ron Jacak, Kristian W Kaufman, P Douglas Renfrew, Colin A Smith, Will Sheffler, et al. Rosetta3: an object-oriented software suite for the simulation and design of macromolecules. *Methods in enzymology*, 487:545–574, 2011.

[9] Peng Xiong, Meng Wang, Xiaoqun Zhou, Tongchuan Zhang, Jiahai Zhang, Quan Chen, and Haiyan Liu. Protein design with a comprehensive statistical energy function and boosted by experimental selection for foldability. *Nature communications*, 5: 1–9, 2014.

[10] Peng Xiong, Xiuhong Hu, Bin Huang, Jiahai Zhang, Quan Chen, and Haiyan Liu. Increasing the efficiency and accuracy of the abacus protein sequence design method. *Bioinformatics*, 36:136–144, 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*, 357:168–175, 2017.

[12] Aaron Chevalier, Daniel-Adriano Silva, Gabriel J Rocklin, Derrick R Hicks, Renan Vergara, Patience Murapa, Steffen M Bernard, Lu Zhang, Kwok-Ho Lam, Guorui Yao, et al. Massively parallel de novo protein design for targeted therapeutics. *Nature*, 550:74–79, 2017.

[13] Kristoffer E Johansson, Nicolai Tidemand Johansen, Signe Christensen, Scott Horowitz, James CA Bardwell, Johan G Olsen, Martin Willemoës, Kresten Lindorff-Larsen, Jesper Ferkinghoff-Borg, Thomas Hamelryck, et al. Computational redesign of thioredoxin is hypersensitive toward minor conformational changes in the backbone template. *Journal of molecular biology*, 428:4361–4377, 2016.

[14] Frederikke Isa Marin, Kristoffer Enøe Johansson, Charlotte O'Shea, Kresten Lindorff-Larsen, and Jakob Rahr Winther. Computational and experimental assessment of backbone templates for computational redesign of the thioredoxin fold. *The journal of physical chemistry B*, 125:11141–11149, 2021.

[15] Grant S Murphy, Jeffrey L Mills, Michael J Miley, Mischa Machius, Thomas Szyperski, and Brian Kuhlman. Increasing sequence diversity with flexible backbone protein design: the complete redesign of a protein hydrophobic core. *Structure*, 20: 1086–1096, 2012.

[16] 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 U.S.A*, 117: 1059–1068, 2020.

[17] Namrata Anand-Achim, Raphael R. Eguchi, Alexander Derry, Russ B. Altman, and Po-Ssu Huang. Protein sequence design with a learned potential. *bioRxiv*. doi: 10.1101/2020.01.06.895466.

[18] Bassil I Dahiyat and Stephen L Mayo. De novo protein design: fully automated sequence selection. *Science*, 278:82–87, 1997.

[19] Thomas Simonson, Thomas Gaillard, David Mignon, Marcel Schmidt am Busch, Anne Lopes, Najette Amara, Savvas Polydorides, Audrey Sedano, Karen Druart, and Georgios Archontis. Computational protein design: the proteus software and selected applications. *Journal of computational chemistry*, 34:2472–2484, 2013.

[20] Xiaoqiang Huang, Robin Pearce, and Yang Zhang. Evoef2: accurate and fast energy function for computational protein design. *Bioinformatics*, 36:1135–1142, 2020.

[21] Shide Liang, Zhixiu Li, Jian Zhan, and Yaoqi Zhou. De novo protein design by an energy function based on series expansion in distance and orientation dependence. *Bioinformatics*, 2021. doi: 10.1093/bioinformatics/btab598.

[22] Xiaoqun Zhou, Peng Xiong, Meng Wang, Rongsheng Ma, Jiahai Zhang, Quan Chen, and Haiyan Liu. Proteins of well-defined structures can be designed without backbone readjustment by a statistical model. *Journal of structural biology*, 196:350–357, 2016.

[23] Mingjie Han, Sanhui Liao, Xiong Peng, Xiaoqun Zhou, Quan Chen, and Haiyan Liu. Selection and analyses of variants of a designed protein suggest importance of hydrophobicity of partially buried sidechains for protein stability at high temperatures. *Protein science*, 28:1437–1447, 2019.

[24] Ruicun Liu, Jichao Wang, Peng Xiong, Quan Chen, and Haiyan Liu. De novo sequence redesign of a functional ras-binding domain globally inverted the surface charge distribution and led to extreme thermostability. *Biotechnology and bioengineering*, 118:2031–2042, 2021.

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[26] Sheng Wang, Wei Li, Shiwang Liu, and Jinbo Xu. Raptorx-property: a web server for protein structure property prediction. *Nucleic acids research*, 44:W430–W435, 2016.

[27] 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 U.S.A*, 117: 1496–1503, 2020.

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

[29] John Ingraham, Vikas K Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. In *Advances in neural information processing systems*, 2019.

[30] Alexey Strokach, David Becerra, Carles Corbi-Verge, Albert Perez-Riba, and Philip M Kim. Fast and flexible protein design using deep graph neural networks. *Cell systems*, 11:402–411, 2020.

[31] 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*, 60:1245–1252, 2020.

[32] 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*, 88:819–829, 2020.

[33] Wen Torng and Russ B Altman. 3d deep convolutional neural networks for amino acid environment similarity analysis. *BMC bioinformatics*, 18:1–23, 2017.

[34] 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*, 60:391–399, 2019.

[35] Sergey Ovchinnikov and Po-Ssu Huang. Structure-based protein design with deep learning. *Current opinion in chemical biology*, 65:136–144, 2021.

[36] 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*, 2017.

[37] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28:235–242, 2000.

[38] Ian Sillitoe, Nicola Bordin, Natalie Dawson, Vaishali P Waman, Paul Ashford, Harry M Scholes, Camilla SM Pang, Laurel Woodridge, Clemens Rauer, Neeladri Sen, et al. Cath: increased structural coverage of functional space. *Nucleic acids research*, 49:D266–D273, 2021.

[39] Dmitrij Frishman and Patrick Argos. Knowledge-based protein secondary structure assignment. *Proteins: structure, function, and bioinformatics*, 23:566–579, 1995.

[40] Huali Cao, Jingxue Wang, Liping He, Yifei Qi, and John Z Zhang. Deepddg: predicting the stability change of protein point mutations using neural networks. *Journal of chemical information and modeling*, 59:1508–1514, 2019.

[41] Patrick Conway, Michael D Tyka, Frank DiMaio, David E Konerding, and David Baker. Relaxation of backbone bond geometry improves protein energy landscape modeling. *Protein science*, 23:47–55, 2014.

[42] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[43] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2017.

[44] Guoli Wang and Roland L Dunbrack Jr. Pisces: a protein sequence culling server. *Bioinformatics*, 19:1589–1591, 2003.

[45] Frank Delaglio, Stephan Grzesiek, Geerten W Vuister, Guang Zhu, John Pfeifer, and AD Bax. Nmrpipe: a multidimensional spectral processing system based on unix pipes. *Journal of biomolecular NMR*, 6:277–293, 1995.

[46] Woonghee Lee, William M Westler, Arash Bahrami, Hamid R Eghbalnia, and John L Markley. Pine-sparky: graphical interface for evaluating automated probabilistic peak assignments in protein nmr spectroscopy. *Bioinformatics*, 25:2085–2087, 2009.

[47] Wei-Zhe Zhang, Jian-Chao Tang, Si-Sheng Wang, Zhi-Jun Wang, Wen-Ming Qin, and Jian-Hua He. The protein complex crystallography beamline (bl19u1) at the shanghai synchrotron radiation facility. *Nuclear science and techniques*, 30:1–11, 2019.

[48] Zbyszek Otwinowski and Wladek Minor. [20] processing of x-ray diffraction data collected in oscillation mode. *Methods in enzymology*, 276:307–326, 1997.

[49] Wolfgang Kabsch. Integration, scaling, space-group assignment and post-refinement. *Acta crystallographica section D: biological crystallography*, 66:133–144, 2010.

[50] Alexei Vagin and Alexei Teplyakov. Molecular replacement with molrep. *Acta crystallographica section D: biological crystallography*, 66:22–25, 2010.

[51] Paul D Adams, Ralf W Grosse-Kunstleve, L-W Hung, Thomas R Ioerger, Airlie J McCoy, Nigel W Moriarty, Randy J Read, James C Sacchettini, Nicholas K Sauter, and Thomas C Terwilliger. Phenix: building new software for automated crystallographic structure determination. *Acta crystallographica section D: biological crystallography*, 58:1948–1954, 2002.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- RotamerFreeProteinSequenceDesignBasedonDeepLearningandSelfConsistencysupp.pdf
- HLiuEPCflat.pdf
- 7VQLvalreportfullP1.pdf
- 7VQWvalreportfullP1.pdf
- 7VQVvalreportfullP1.pdf
- 7VTYvalreportfullP1.pdf
- 7VU4valreportfullP1.pdf